UNIDAD I

INTRODUCCIÓN A LA GRÁFICA POR COMPUTADORA

En esta unidad se presenta una introducción general a la gráfica por computadora. Después de dar una reseña histórica, se describen las características básicas del dispositivo gráfico y se termina por mencionar algunas áreas importantes de aplicación de la gráfica por computadora.

I.1 LA GRAFICA POR COMPUTADORA

La **gráfica por computadora** es la creación, almacenamiento, manipulación y despliegue de imágenes con la asistencia de una computadora. La imagen puede ser un fin en si, tal como la portada de una revista. Sin embargo la imagen sirve a menudo para comunicar información sobre un objeto, proceso, sistema u otra cosa y entonces la imagen no es un fin, sino un medio de comunicar información. En este caso, la imagen es la representación visual de un modelo que está almacenado en la computadora. Un **modelo** se define como un conjunto de datos que representa las características de un objeto más importantes para una aplicación determinada. Una definición más general de la gráfica por computadora es la creación, almacenamiento y manipulación de modelos de objetos y sus imágenes por medio de una computadora.

Un ramo relacionado a la gráfica por computadora es el procesamiento de imágenes. Se trata de tomar imágenes, normalmente fotografías o videos digitalizados, e intentar reconocer objetos o patrones en ellas, o modificar la imagen para enfatizar ciertas cualidades. Entonces tenemos el proceso inverso: en lugar de construir la imagen en base de los datos, se empieza con la imagen y se trata de obtener información, o a veces, modificar dicha imagen.

A continuación se presentan algunos ejemplos de las aplicaciones de procesamiento de imágenes. En la cartografía, se analizan fotografías aéreas o de satélite para producir mapas de bosques, cultivos, y otros. En la radiografía, se intenta reconocer huesos, tumores y otras estructuras del cuerpo. En la automatización industrial se analizan imágenes grabadas por una cámara de video para detectar productos defectuosos, y para controlar las acciones de ciertas máquinas.

Un concepto importante en la gráfica por computadora es el de la interacción. Un programa que solamente grafica la función **seno(x)** es un ejemplo de gráfica no interactiva, o pasiva. En cambio, un programa de diseño de circuitos, o un juego de video, serían ejemplos de gráficas interactivas. La **gráfica interactiva** es la técnica de utilizar comunicación bi-direccional entre la computadora y el usuario en la que la computadora se comunica con imágenes gráficas. La computadora, al

recibir señales del usuario, modifica la imagen. Y el usuario, viendo la imagen, puede tomar acciones para modificarla.

Un ejemplo de la gráfica interactiva es un programa de diseño de circuitos, en el que el diseñador puede agregar un nuevo componente y conectarlo con otras partes del circuito, o borrar componentes y conexiones. Otro ejemplo sería un juego de video, al ver un proyectil acercándose a su nave, el jugador puede conducir la nave para evitar un impacto. La razón principal para la efectividad de la gráfica interactiva en estas aplicaciones es la rapidez con la cual el usuario puede asimilar la información desplegada.

I.2 TECNOLOGIAS GRAFICAS

I.2A UNA BREVE HISTORIA DEL DISPOSITIVO DE DESPLIEGUE

A continuación se presentan algunos de los criterios que han impulsado y dirigido el desarrollo de la tecnología de despliegue.

- *__Rapidez de Despliegue__: El tiempo que se requiere para dibujar. También se toma en cuenta la carga en el CPU para controlar el despliegue.
- *__Modificación Selectiva__: la facilidad con que se puede modificar una parte de la imagen sin afectar el resto.
- *__Realismo y Calidad de la Imagen__:
  - -__Alta Resolución__: La **resolución** se define como el número de puntos distinguibles por pulgada que se pueden desplegar, pero comúnmente es usada para significar el número total de puntos direccionables en una pantalla. La resolución está muy relacionada con la precisión de la representación de una imagen.
  - -__Relleno__: la calidad en la representación de objeto sólidos.
  - -__Color__: el rango de colores, tonos e intensidades que se pueden desplegar.
  - -__Estabilidad__: La imagen no es intermitente ni mareada; y al ser borrada, no deja "fantasmas" en la pantalla.
- *__Precio__: mientras más barato sea, más útil y utilizado será.

A inicios de 1950 se utilizó un **Tubo de Rayo Catódico** (CRT) para exhibir la salida en la computadora "Whirlwind" de MIT. A mediados de los años cincuenta nació la gráfica interactiva cuando se agregó un "lapicero de luz" al CRT a un sistema de defensa aérea, permitiendo al operador tocar la pantalla con el **lapicero de luz** para señalar la posición de un blanco.

El primer avance después del CRT básico sucedió a mediados de los años sesenta, cuando fue desarrollado el **tubo de vectores**. Este dispositivo tiene un CRT, un procesador y una memoria conocida como "buffer de despliegue". El buffer de despliegue guarda una lista de comandos para dibujar líneas y mover el rayo, entre otras cosas. El procesador del tubo de vectores lee secuencialmente los comandos en la lista y dirige el rayo catódico sobre la pantalla, dejándolo encendido para dibujar una línea o apagándolo para moverse a otra posición. Al llegar al fin de la lista, el procesador vuelve al principio de la lista y empieza de nuevo. Hay que redibujar o "refrescar" la imagen

continuamente ya que el fósforo de la pantalla brilla solamente por unos milisegundos. Cuando hay muchas líneas en la lista, empiezan a apagarse antes de ser refrescadas, creando un efecto molesto de intermitencia. Uno de los fuertes del tubo de vectores es su alta resolución. Una limitante que tiene es que, debido al dibujo basado en líneas, no puede dibujar figuras rellenas ni usar colores.

Para resolver estos problemas fue desarrollado a finales de los sesenta el **tubo de almacenamiento**, dispositivo en el que la imagen se guarda sin refrescar. Los puntos son pedazos de fósforo ubicados en las intersecciones de una cuadrícula de alambres que está detrás de la pantalla. El rayo, que tiene posicionamiento aleatorio, se mueve sobre un punto y aumenta su voltaje brevemente para encenderlo. El punto se mantiene encendido hasta que se blanquea la pantalla entera. Algunos de estos equipos dibujan líneas por medio de mover el rayo encendido entre dos puntos en la pantalla, encendiendo todos los puntos intermedios. El tubo de almacenamiento tiene menor resolución que el tubo de vectores, no tiene colores, pero permite la fácil representación de formas rellenas. No permite la animación o la modificación selectiva de la imagen.

El tubo de vectores se encontraba solamente en unas cuantas empresas grandes e instituciones porque la memoria y el procesador que requiere lo hizo un aparato bien caro. El tubo de almacenamiento, con un precio inicial de $15,000, fue una ganga comparado con el costo aproximado a los $100,000 de los tubos de vectores, propiciando por primera vez la amplia difusión de la gráfica por computadora.

El último salto hacia adelante en la tecnología de despliegue ocurrió a mediados de los años setenta con la introducción del **despliegue raster** o **despliegue de barrido**, basado en la tecnología del televisor. Este dispositivo utiliza una memoria **bitmap** (mapeo de bits) para guardar una matriz de puntos que representan la imagen. Los puntos en la memoria se convierten en un señal para impulsar el rayo que barre la pantalla repetidamente en un patrón fijo. Es una tecnología barata y actualmente la resolución de los bitmaps está alcanzando la de los tubos de vectores, con las ventajas de que se pueden dibujar objetos sólidos y usar colores.

Hay varias otras tecnologías de despliegue en uso hoy día, tales como impresoras de puntos y láser, plotters, y grabadores de película y video. Pero el dispositivo gráfico más importante es la pantalla porque permite la gráfica interactiva. Aunque hay varios tipos de pantalla, tales como "el tubo de almacenamiento" y el "tubo de vectores", el más común es el bitmap, que normalmente usa un tubo (CRT) de barrido. Las nuevas computadoras portátiles de clase "maletín" también usan un bitmap, pero con pantallas LCD (diodo de cristal de litio) y plasma. Casi no hay diferencia en la programación de estos dispositivos, entonces este folleto se enfoca en la tecnología dominante: el bitmap con pantalla CRT de barrido.

La historia no sería completa sin mencionar algunos avances en el software gráfico. El suceso más importante fue el desarrollo, en 1963, del sistema de dibujo interactivo "Sketchpad" (block de esbozo) como proyecto de doctorado de Ivan Sutherland, el padre de la gráfica por

computadora. En Sketchpad el usuario usa un lápiz de luz para poner primitivos gráficos tales como líneas, arcos y caracteres en la pantalla. Además permite la creación de objetos compuestos de múltiples primitivos, y la transformación interactiva de la imagen.

Después de Sketchpad se desarrollaron varios sistemas primitivos de diseño asistido por computadora (CAD) usando tubos de vectores. Estos fueron especializados para industrias específicas tales como las de automóviles y aviones. Pero debido a esta falta de generalidad y de portabilidad su uso fue muy limitado. Una de las aplicaciones que más impulsó el desarrollo del realismo en las gráficas por computadora fue la simulación de vuelos para capacitar pilotos. La simulación no funcionaba bien con las imágenes de "marcos de alambres" de los tubos de vectores, entonces empezaron a investigar la tecnología de los televisores y los algoritmos para crear objetos sólidos. Ahora la industria de entretenimiento está en la vanguardia de la síntesis de imágenes realistas, y allí se pueden usar técnicas más complejas porque no se tiene el requerimiento de animar las imágenes en tiempo real que tiene la simulación de vuelos.

Otro avance que vale destacar fue la introducción de la interfaz gráfica al usuario. Mucho del trabajo inicial para esta se hizo en el centro de investigaciones Xerox PARC en los principios de los años setenta. Alan Kay encabezó un equipo que desarrolló una computadora personal llamada "Alto", basada en dos ideas que impactaron la gráfica por computadora: primero, que la interfaz al usuario debería ser principalmente gráfica, y segundo, que el dispositivo gráfico debería ser un bitmap. Unas de las primeras y las más exitosas microcomputadoras comerciales para acatar estos avances fueron los modelos Lisa y el MacIntosh de Apple.

I.2B EL CONCEPTO DEL BITMAP

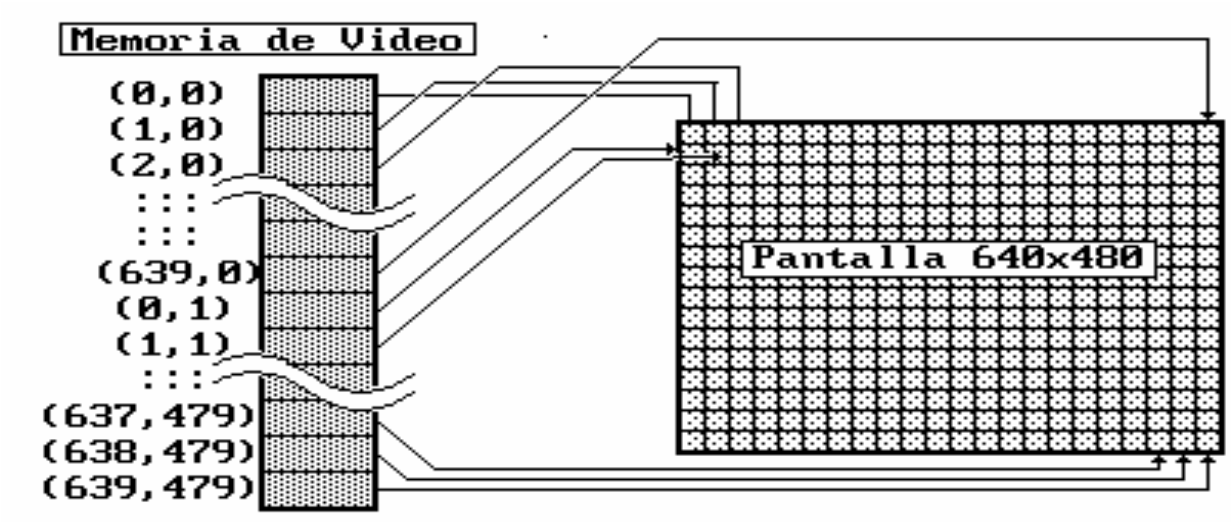

Fig. I.1 Mapeo desde la Memoria de Video hasta la Pantalla en Modo Gráfico. El origen de la pantalla está en la izquierda superior debido al patrón de barrido del CRT.

El **bitmap** es un dispositivo gráfico que utiliza una pantalla CRT (tubo de rayo catódico) de barrido (como un televisor) y una memoria que guarda una representación de la imagen. La imagen está organizada en una matriz de puntos, cada uno de los cuales corresponde a una celda en la memoria. Un punto se llama **pixel**, del inglés "picture cell", y la memoria se llama **memoria de video** o **memoria gráfica**. El color o intensidad de un pixel es controlado por el valor almacenado en la celda correspondiente en la memoria de video. El circuito del bitmap va leyendo el contenido de la memoria de video, convirtiéndolo en una señal para controlar la intensidad (o color) del rayo mientras este barre repetidamente sobre la pantalla entera, de izquierda a derecha, línea por línea.

Entonces, cada pixel en la pantalla es individualmente controlable desde el programa. En efecto, hay una correspondencia 1 a 1 entre las celdas de la memoria gráfica y los pixeles en la pantalla. Por eso, el dispositivo se llama "mapa de bits", o **bitmap**. Otros nombres comunes son "frame buffer" (buffer de marco), "raster graphics" (gráficas por rastreo), "interfaz de video", "tarjeta gráfica" y "adaptador gráfico". Este último es el nombre usado para los bitmaps de la IBM PC.

La resolución de un bitmap es el número de columnas de pixeles por el número de filas en el despliegue. A menudo se incluye en la resolución el número de bits en la celda que corresponde a un pixel, normalmente expresado como "bits por pixel". La resolución varía de aproximadamente 200 x 200 hasta 4000 x 4000, y el número de bits en cada celda varía entre 1 y 64, según la clase de dispositivo.

I.2B.1 CONTROL DE COLOR DEL BITMAP

Por ahora, consideraremos el tipo más sencillo de control de color: control directo, en el que el valor del pixel determina directamente el color desplegado. Aunque no son muchos los bitmaps que usan control directo de color, para los propósitos de explicar el control de color el principio es el mismo.

Control directo implica que el valor del pixel determina directamente el color del pixel, o sea, que el hardware del bitmap convierte cada valor de pixel a un color predefinido. Por ejemplo, valor de pixel 0 puede ser siempre negro, valor de pixel 1 puede ser siempre rojo, etc., sin la posibilidad de redefinir esta correspondencia.

El número de colores disponibles en un bitmap es igual al número de posibles valores que puede tener un pixel. Este se calcula en 2 elevado al número de bits/pixel. La tabla a continuación da algunos ejemplos de este cálculo.

Bits/Pixel	Colores Disponibles
1	2
2	4
4	16
8	256
16	65,536

En un sistema monocromo (negro y blanco), el valor de pixel 0 corresponde a negro y el valor máximo de pixel corresponde a blanco. Los valores intermedios corresponden a graduaciones de gris. Esto se llama una escala gris. Por ejemplo, si el dispositivo de escala gris tiene 2 bits/pixel, los 4 colores serán los siguientes:

Valor de Pixel	Color
0	negro
1	gris oscuro
2	gris claro
3	blanco

I.2B.2 EL MODO ALFANUMERICO

Los bitmaps a menudo tienen un modo de operación en que despliegan solamente caracteres, llamado **modo alfanumérico** o **modo de caracteres.** En este modo los pixeles de la pantalla no son individualmente programables. El modo de operación presentado en la sección anterior, en el que cada pixel es individualmente controlable, se llama el **modo gráfico.** Algunos adaptadores disponen modos de ambos tipos, incluso a veces una selección de modos de cada tipo, mientras otros bitmaps disponen de solamente uno u otro. En la PC, el modo alfanumérico es el modo de operación por defecto, y toda máquina de esa clase lo dispone. Con el desarrollo del diseño de la PC, IBM ha ido agregando nuevos modos gráficos, pero estos son aprovechados solamente por aplicaciones especializadas. La computadora MacIntosh, en cambio, no dispone de modo alfanumérico y trabaja siempre en modo gráfico.

En el modo alfanumérico, la pantalla está organizada como una matriz de caracteres (veáse figura I.2). Cada posición de caracter en la pantalla corresponde a una celda en una memoria que almacena un código ASCII. Para desplegar un caracter en la pantalla, hay que almacenar el código del caracter en la celda de la memoria que corresponde a la posición deseada. La memoria que guarda los códigos ASCII se llama la **memoria de video** o la **memoria de caracteres.** Las celdas de la memoria de

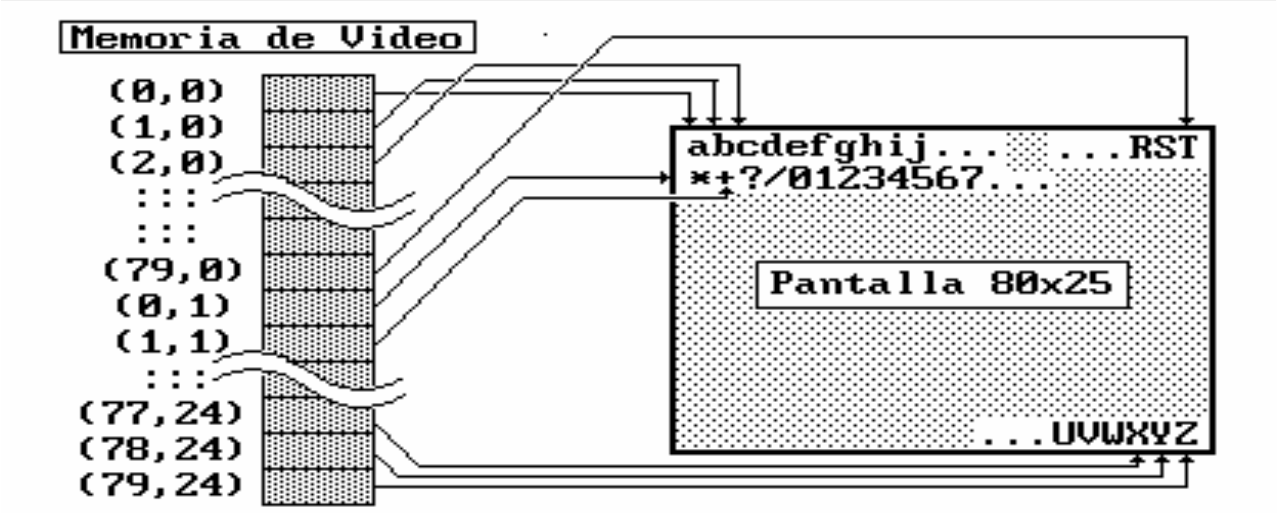

Fig. I.2 Mapeo desde la Memoria de Video hasta la Pantalla en el Modo Alfanumérico

caracteres normalmente son de 8 bits de ancho, y así son posibles de guardar 256 distintos códigos de caracter. El bitmap también guarda un conjunto de patrones de los caracteres, y se encarga de traducir los códigos ASCII a su patrón correspondiente para ser desplegado en la pantalla.

El bitmap en modo de caracteres tiene el único propósito de almacenar los códigos ASCII en su memoria de video y convertirlos a caracteres visibles en la pantalla. El bitmap no implementa las acciones correspondientes a los caracteres de control tales como "retorno", "retroceder" y "nueva línea" (códigos 13, 9 y 10 respectivamente). Estas acciones de control tienen que ser implementadas a un nivel de control más alto, normalmente en un manejador o en un programa de aplicación. De hecho, en los bitmaps de la PC se puede almacenar los códigos de los caracteres de control y ver un caracter visible en la posición correspondiente en la pantalla. Por ejemplo, almacenar el código de <Retorno> en la memoria de video hace desplegar una nota musical en la pantalla.

I.2B.3 SELECCION DE MODO

Los adaptadores de la PC proporcionan una amplia selección de modos de caracteres y de gráficas. El modo por defecto de la PC es un modo de caracteres. Para hacer gráficas, un programa debe poner primero el adaptador en el modo gráfico. Al terminar con las gráficas, es recomendable regresar el adaptador a su modo original. Al nivel bajo, cambiar modos significa escribir numerosos valores a los registros de control del adaptador. El cambio de modo de operación del bitmap se puede realizar por medio de un servicio del BIOS, o una función de Turbo C. En ambos casos, el programa proporciona solamente el índice del modo que desea, y el BIOS o Turbo C hace la inicialización de todos los registros del bitmap.

La tabla a continuación muestra los modos disponibles en los 5 adaptadores más comunes en la PC (MDA, CGA, EGA, VGA) y su sucesor, la PS-2 (EGA, MCGA). La identificación más común para los modos es por medio de los índices que usa el BIOS para identificarlos. Las columnas a la derecha de la tabla, encabezadas con los tipos de adaptadores, tienen un "x" cuando el tipo de adaptador proporciona el modo correspondiente.

BIOS Modo	Gráfico/ Caracter	Resolución (H x V x bits/pixel)	MDA	CGA	EGA	MCGA	VGA
0,1	C	40x25, 16 colores		x	x	x	x
2,3	C	80x25, 16 colores		x	x	x	x
4,5	G	320x200x2		x	x	x	x
6	G	640x200x1		x	x	x	x
7	C	80x25, monocromo x		x		x	
13	G	320x200x4		x		x	
14	G	640x200x4			x		x
15	G	640x350x1 (mono)			x		x
16	G	640x350x4			x		x
17	G	640x480x1			x	x	
18	G	640x480x4				x	
19	G	320x200x8				x	x

Notas:

1) Los modos 0, 2 y 4 son iguales a los modos 1, 3 y 5 en todos los adaptadores excepto el CGA, donde la señal compuesta del monitor es monocromo en los modos pares y de colores en los impares.

2) El adaptador Hércules dispone un modo de caracteres (80x25, monocromo) y uno de gráficas (720x350x1). El Hércules no es soportado por IBM y por lo tanto no tiene soporte completo en el BIOS. Sin embargo, si es soportado por muchos paquetes de software tales como Turbo C y Lotus 1-2-3.

Vale tocar el tema de la evolución de los adaptadores de las micros de IBM. La primera PC de IBM en 1982 tenía como adaptador estándar el MDA, que proporciona texto monocromo sin gráficas. Otras empresas ya ofrecían microcomputadoras con gráficas a un precio parecido. Para aumentar sus ventas, IBM sacó pronto el adaptador CGA que proporciona gráficas en colores, pero es de baja resolución y de pocos colores, su diseño hace muy inconveniente la programación y es incompatible con el MDA. De hecho, la calidad de los caracteres se empeoró, reduciendo el tamaño del bloque de caracter desde 9x14 pixeles en el MDA a solamente 8x8 pixeles en la CGA.

Otros fabricantes de las micros avanzaban rápidamente en la gráfica, e IBM respondió en serio por primera vez con el anuncio del adaptador EGA. El EGA mantuvo compatibilidad hacia abajo con todos los modos del MDA y del CGA, e introdujo varios nuevos modos gráficos con mayor resolución y más colores. Además, reparó la degradación de los caracteres en los modos alfanuméricos, aumentando el bloque de caracter a 8x14 pixeles. La nueva microcomputadora de IBM, la PS/2, estrenó con el adaptador MCGA que es compatible hacia abajo con el MDA y el CGA. El más reciente adaptador para el PC, en salir de IBM es el VGA, compatible por abajo con todos los adaptadores anteriormente mencionados y proporcionando algunos nuevos modos gráficos que por primera vez permiten el desarrollo de aplicaciones gráficas profesionales.

Parece extraño que el adaptador CGA ofrece un modo de texto de resolución 40x25 cuando hay también uno de 80x25. ¿Por qué se usaría el primer modo cuando el segundo exhibe el doble del número de caracteres? Existe el mismo fenómeno entre los modos gráficos del CGA de resolución 320x200 y de 640x200. La razón es que los monitores de baja calidad, como son los televisores de casa, no pueden exhibir con claridad una línea de 80 caracteres o de 640 pixeles de ancho. Los modos de menor resolución son más adecuados a esos monitores baratos.

I.3 EL PAQUETE GRAFICO

Un **sistema gráfico** es una combinación de hardware y software para facilitar el uso de entrada y salida gráfica en los programas. Uno de los componentes de software en el sistema gráfico es el **paquete gráfico**, una colección de rutinas que dibujan y realizan otras operaciones de manejo de datos gráficos y del dispositivo gráfico. El paquete gráfico también se llama una "biblioteca gráfica" o a veces, una "caja de herramientas gráficas". El propósito del paquete gráfico es proporcionar al programador de alto nivel una interfaz al dispositivo gráfico para liberarlo de los detalles de su manejo. Los programas de aplicación

gráfica normalmente usan un paquete gráfico para manejar el dispositivo gráfico.

I.3A EL MODELO DEL PROGRAMADOR

El paquete gráfico proporciona al programador de aplicación una interfaz al dispositivo más entendible y conveniente que el propio dispositivo. Esta interfaz, llamada el "modelo del programador", crea un dispositivo virtual para el programador al esconder muchos de los detalles del manejo del dispositivo real. Las dos áreas más importantes en este sentido son el sistema de coordenadas y el esquema de colores.

Los paquetes frecuentemente proporcionan una clase de coordenadas más flexible que el sistema de coordenadas del hardware, llamadas **coordenadas físicas** (CF). El tratamiento más conveniente es permitir que el usuario defina su propio sistema de coordenadas, llamadas **coordenadas mundiales** (CM), y el paquete maneje de manera transparente (es decir, el programador de aplicación no tiene que saberlo) el mapeo de la imagen a la pantalla. Esto da independencia de las coordenadas físicas, que no solamente tiene mal ubicado el origen y usa unidades que no son las más adecuadas, sino que varía entre distintos dispositivos y modos de operación. Sin embargo, esta flexibilidad se consigue a costo del control sobre la imagen final. Por ejemplo, es difícil anticipar si dos puntos cercanos en coordenadas mundiales serán mapeados a un solo pixel, a dos pixeles contiguos o a dos pixeles separados, ya que esto depende de la conversión de coordenadas.

Otra técnica de hacer más flexible el uso de coordenadas en algunos paquetes es el uso de **coordenadas relativas**, donde se interpretan las coordenadas como distancias relativas a una posición determinada, llamada la posición actual o puntero gráfico, y a veces marcada con un cursor visible. Las coordenadas relativas se distinguen de las **coordenadas absolutas** en que estas últimas son distancias del origen. Tales paquetes tiene dos grupos de funciones, uno para dibujar los primitivos en coordenadas absolutas, y otro para hacerlo en coordenadas relativas. Las coordenadas relativas permiten la definición de figuras independientes de la posición. Una figura así definida puede desplegarse en cualquier parte de la pantalla, usando el puntero gráfico para controlar su posición. Una figura definida en coordenadas absolutas, en cambio, siempre se dibuja en la misma posición, a menos que sus coordenadas se trasladen explícitamente, lo que sale más caro que el uso de coordenadas relativas y está disponible solamente en los paquetes avanzados.

La otra área donde el paquete gráfico puede simplificar la programación de aplicaciones mediante la creación de un modelo, es en la selección de colores. Un dispositivo de color normalmente tiene un mapa de color: una tabla en el hardware para guardar las definiciones de colores que corresponden a los distintos valores de pixel. Cada color en esta tabla se define como combinación de tres intensidades de rojo, verde y azul. Es bastante difícil programar el mapa de color para lograr el efecto deseado. Por eso, algunos paquetes proveen un modelo alternativo de color, tal como un conjunto de colores predefinidos, o el sistema TIS, donde se especifican tinte, iluminación y saturación en lugar de mezclas de tres colores primarios. El modelo más sencillo que un paquete puede ofrecer es un conjunto de colores predefinidos, tales

como "celeste" y "rosado", que el paquete traduce a valores rojo-verde-azul para cargar en el mapa de color. Así el programador de aplicación no tiene que preocuparse con la composición rojo-verde-azul de un color, pero a costos de perder mucha flexibilidad en el uso de color. El sistema TIS es más complejo para manejar, pero más flexible. Estos temas se tratan en detalle en la unidad VII.

I.3B CLASES DE FUNCIONES EN EL PAQUETE GRAFICO

Para profundizar el concepto del paquete gráfico, a continuación se presentan siete clases de funciones que pueden encontrarse en un paquete gráfico. Muchos paquetes tiene un subconjunto de estas. Turbo C, por ejemplo, tiene solamente funciones de las categorías (1), (2) y (7). Si el programador en Turbo C requiere funciones de las otras clases, tendrá que escribirlas.

1. Salida de Primitivos Gráficos: Dibujan figuras tales como puntos, líneas y círculos; además pueden incluir texto, relleno con patrones y otros objetos.

2. Atributos: Fijan las características de los primitivos tales como el color, patrón de líneas, fuente y tamaño de texto, etc.

3. Control de Segmentos: Un segmento es uno o más primitivos gráficos agrupados para ser manipulados como una unidad. Operaciones típicas en segmentos son: abrir y cerrar un segmento; suprimir; trasladar; modificar atributos; y visualizar u ocultar.

4. Operación de Visualización: Permite la definición de un sistema de coordenadas por el usuario, y un mapeo de una parte de este sistema a una parte de la pantalla.

5. Transformaciones: Realizan operaciones de traslación, rotación y escalación u otras transformaciones de coordenadas en los primitivos gráficos.

6. Entrada: Manejan la interacción entre el usuario y el programa de aplicación, por ejemplo, la selección de una posición o un objeto en la pantalla, uso de menúes, etc.

7. Control y Utilidad: Manejan un rango de operaciones tales como inicialización y control del dispositivo, manejo de errores, la interrogación sobre atributos y características del dispositivo, y lectura de partes de la imagen.

I.3C EL DISEÑO DEL PAQUETE GRAFICO

A continuación se presentan algunas de las cualidades deseables en el diseño de un paquete gráfico. El alumno debería de mantenerlas en la mente cuando lea la próxima unidad sobre el paquete gráfico de Turbo C.

Simplicidad

Opciones que son demasiado complejas para ser entendidas por el programador de aplicación no serán usadas. A veces el diseñador del paquete no puede reconocer de antemano las partes que serán difíciles de entender. Una solución es escribir primero el manual del usuario antes

de desarrollar el sistema. Cualquier cosa que es difícil de explicar, probablemente será también difícil de entender. Cabe señalar que el propósito del paquete no es dar acceso a toda función del dispositivo gráfico, sino crear un "dispositivo virtual", una interfaz en software al hardware gráfico que presente un modelo más sencillo. Entonces, las funciones más complejas no tienen que ser desechadas si pueden ser escondidas detrás de una interfaz más comprensible.

Funcionalidad Completa

La funcionalidad tiene que ver con el rango de operaciones que el paquete realiza. La meta es proporcionar un conjunto de funciones que satisfaga la mayoría de las aplicaciones, permitiendo la fácil realización de aplicaciones gráficas sin tener que escribir rutinas para operaciones gráficas básicas o programar directamente el hardware. La selección de funciones debe incluir aquellas de uso general, evitando la trampa de tratar de responder a todos los casos posibles.

Ortogonalidad

La ortogonalidad es la capacidad de hacer un amplio rango de operaciones con un mínimo de funciones. La separación de los atributos (color, patrón etc.) de los primitivos gráficos (puntos, líneas etc.) es un ejemplo. Vale entender bien este concepto ya que es válido para cualquier paquete, gráfico o no gráfico.

Considere un primitivo gráfico como una línea, que puede tener atributos de color, grosor y estilo (quebrada, punteada etc.) además de su forma geométrica. Un paquete no muy ortogonal podría tener una función para fijar el color, otra para fijar el grosor y estilo, y otra para dibujar una línea entre dos puntos usando el color, estilo y grosor actual. Puede haber tres funciones, como estas de Turbo C:

```
void setcolor (int color);
void setlinestyle (int estilo, int grosor);
void line (int x1, int y1, int x2, int y2);
```

La ortogonalidad toma en cuenta la forma usual en que se van a usar las funciones. Por ejemplo, normalmente no se va a cambiar el estilo y el grosor cada vez que se dibuja una línea. Entonces es bueno separar la función para fijar el estilo y el grosor. El color de dibujo frecuentemente se usa no solamente para líneas, sino también para otros objetos que se dibujan a la vez. Por lo tanto, tiene sentido hacer una sola función de color para todo objeto gráfico en lugar de uno para líneas, uno para puntos, uno para círculos etc.

Un paquete gráfico que combina en una sola función los atributos y la geometría de una figura sería menos ortogonal. Un caso aún menos ortogonal sería un paquete que tuviera una función para cada posible combinación de atributos, y quizá para distintas geometrías. ¡Imagínese si hubiera centenares de funciones como

```
void linea_horizontal_verde_ancha_punteada (x, y, longitud);
```

Es cierto que es un caso exagerado, pero la idea es clara. Ortogonalidad sugiere que las clases de las funciones, como los vectores en el álgebra lineal, representen distintas dimensiones, tal que dos o tres funciones

sencillas puedan definir una clase de objetos gráficos que llenen un espacio respectivamente de dos o tres dimensiones.

Consistencia

La consistencia en un paquete se refiere al hecho de que se deben adoptar y obedecer algunas convenciones acerca de cosas tales como nombres y parámetros de las funciones. Por ejemplo, si la función para dibujar un círculo relleno es:

```
void circ_rell (int x, int y, int rad, int patron);
```

la lógica de consistencia sugiere que la función para dibujar un rectángulo relleno debería ser algo como:

```
void rect_rell (int x1, int y1, int x2, int y2, int patron);
```

El nombre tiene la misma forma, y los parámetros usan el mismo orden. Pero usar lo siguiente, confundirá al programador:

```
void rellenar_rect (int patron, int y1, int y2, int x1, int x2);
```

Otra parte de la consistencia tiene que ver con el comportamiento de las funciones. Por ejemplo, deberían tratar de la misma manera casos de excepción o de error. Si el paquete es consistente, se entiende mejor.

Robustez

Los programadores de aplicación son capaces de todo tipo de mal uso de un sistema gráfico, por equivocación o por curiosidad. El sistema debería soportar tal tratamiento con un mínimo de queja. Los errores triviales deberían ser corregidos sin aviso del sistema. Cuando el error es grave, el sistema debería de avisar en la manera más útil posible. Solamente en casos extremos el sistema debe terminar la ejecución ya que causaría la pérdida de resultados valiosos.

Rapidez

La rapidez de un sistema gráfico está frecuentemente limitada por factores tales como el sistema operativo o el dispositivo gráfico. Si el sistema va a implementarse en distintos ambientes, el diseñador puede minimizar el impacto de estos factores evitando funciones que requieren una respuesta muy rápida o hardware muy especializado. Debe de normalizar la rapidez para que las aplicaciones tengan un tiempo de respuesta consistente. Una aplicación debe correr con una velocidad consistente sin recurrir a trucos especiales que son dependientes en el hardware.

Economía

Un sistema gráfico que es demasiado grande o lento, será difícil de usar. El sistema gráfico debe mantenerse pequeño y rápido para que la adición de gráficas a cualquier aplicación tenga un costo mínimo.

Generalidad

El paquete debería fomentar la portabilidad de las aplicaciones gráficas. Entonces tiene que proporcionar las funciones que representen las operaciones gráficas más generales sin hacerlas depender de las características idiosincráticas de un dispositivo específico. Implicando así que se sacrificaría una parte de la funcionalidad inherente de un

dispositivo gráfico para ganar una mayor portabilidad. Además, el programador de aplicación no debería ser obligado a entender muchos de los detalles del hardware para usar el paquete.

I.3D LOS ESTANDARES GRAFICOS

Lograr la independencia del hardware para tener software portable es un reto constante en la programación. Para muchas aplicaciones, esto se logra mediante la estandarización de lenguajes de programación de alto nivel. El SQL viene dando portabilidad a las aplicaciones de bases de datos. El UNIX estandariza el sistema operativo en docenas de máquinas. La independencia del hardware se logra escondiendo los detalles del hardware, así que el usuario o el programador de aplicación no nota las diferencias. Una interfaz en software independiente de la máquina se puede llamar una "máquina virtual", que aparenta a las aplicaciones una máquina idealizada. El manejo de tal interfaz resta eficiencia de corrida, pero esto puede ser minimizado mediante la incorporación de soporte en el hardware para los estándares.

El software gráfico está poco estandarizado por varias razones. Primero, es por que la gráfica por computadora es una tecnología relativamente joven. El rápido desarrollo tecnológico y la variedad de necesidades en la gráfica ha producido gran diversidad de características funcionales en los dispositivos gráficos que dificulta el trabajo de definir el "dispositivo gráfico virtual" necesario para la estandarización. Segundo, dado que los algoritmos gráficos son complejos y lentos, el software gráfico frecuentemente trabaja muy cerca del hardware para ganar eficiencia, creando una dependencia en el hardware que previene la creación de una interfaz estándar.

Hay muchos problemas de portabilidad de las aplicaciones gráficas debido a las distintas características de los bitmaps tales como el número de colores desplegados y la resolución en pixeles. Si una aplicación despliega una imagen de 16 colores, por ejemplo, es difícil que esta sea desplegada en un dispositivo monocromo sin que salga distorcionada hasta ser irreconocible.

La situación con la resolución no es mucho mejor. Si una imagen desarrollada para un despliegue de resolución 1000 x 1000 pixeles tiene que desplegarse en un dispositivo de resolución de 200 x 400, inevitablemente se pierde mucho detalle. Una solución al problema de las diferencias en resoluciones es dibujar en un sistema de coordenadas independiente del dispositivo (véase la unidad IV), y luego transformar estas a las coordenadas del dispositivo. Para las figuras geométricas, como líneas y círculos, que no son pequeños, funciona bien en muchos casos.

Los problemas relacionados a la resolución surgen en la **discretización** de la imagen, que es la representación aproximada de una figura teórica por un conjunto de pixeles reales en la pantalla, porque en este proceso las diferencias en la resolución de pixeles producen grandes diferencias en la calidad de la imagen. La figura I.3 muestra el problema de transformar una imagen desde un dispositivo a otro con la mitad de la resolución. La figura I.3b muestra la letra "a" en la pantalla de alta resolución. Si se quiere mantener la escala de la

imagen en el dispositivo de baja resolución, se pierde mucho detalle (figura I.3 c). Pero si se quiere mantener el nivel de detalle, hay que usar una área mayor de la pantalla con el resultado de que se mira sólo una parte de la imagen (figura I.3d). La transformación de coordenadas además aumenta el tiempo de ejecución, haciéndola una solución no factible para muchas aplicaciones de gráficas dinámicas.

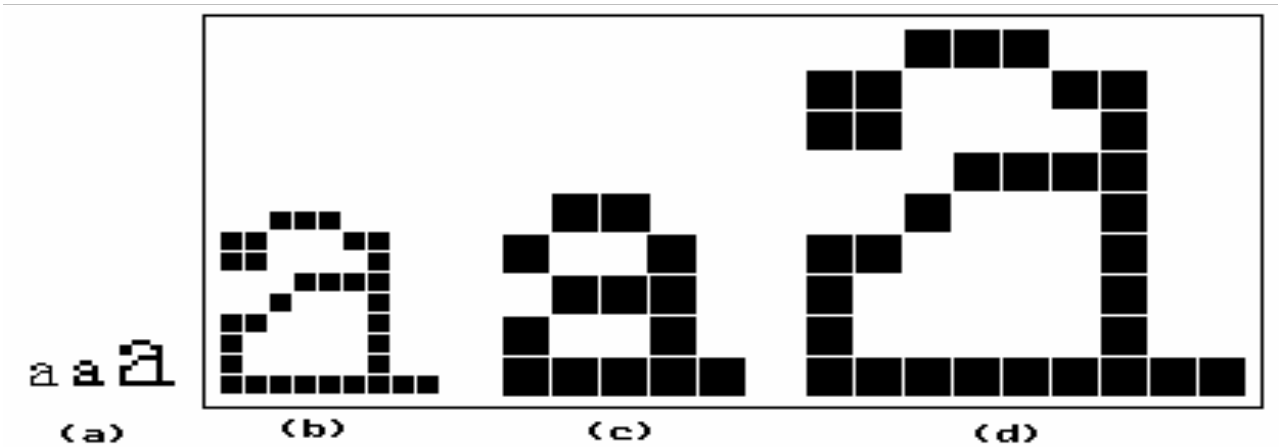

Fig. I.3 El Problema de Portabilidad entre Adaptadores de Distintas Resoluciones

Entonces el ideal de estandarizar la interfaz al dispositivo gráfico está lejos todavía y el software gráfico continuará padeciendo de una baja portabilidad. Una alternativa para tratar las diferencias entre dispositivos es hacer que el programa modifique su comportamiento de acuerdo al tipo de adaptador, pero es menos preferible ya que la carga de tratar las diferencias entre los dispositivos cae en el programador de aplicación. Además, tales aplicaciones son complejas y son portables solamente entre los tipos de dispositivos que son explícitamente reconocidos.

En los años sesenta, cuando surgieron los primeros dispositivos gráficos comerciales, los programas gráficos fueron creados "desde la tierra por arriba". Eran bien costosos de desarrollar porque toda la programación gráfica tenía que ser hecha de nuevo cada vez. El primer avance fue el desarrollo del paquete gráfico para facilitar el desarrollo de aplicaciones gráficas en un ambiente específico para un dispositivo específico. El paquete es un conjunto de rutinas para realizar operaciones básicas tal como el dibujo de líneas para que no haya que escribir las mismas rutinas básicas nuevamente en cada aplicación. Sin embargo, estos estaban restringidos a funcionar en un único tipo de computador con un unico tipo de dispositivo. No se podía transportar la aplicación a otra computadora u otro dispositivo sin reescribir una gran parte del código.

Surgió en la industria de la gráfica por computador un estándar de hecho basado en las capacidades del dispositivo gráfico más popular de aquel entonces, el Tektronix 4014. Es un dispositivo que recibe, a través de una línea serial, comandos gráficos para dibujar primitivos tales como líneas o caracteres, así como para realizar operaciones de control tales como limpiar la pantalla. Muchos de los competidores publicaron sus productos como "compatible con el Tektronix 4014", o con "emulación en software del Tektronix 4014". El software de emulación del

4014 recibe comandos gráficos del 4014 provenientes de un programa de aplicación o de un archivo y los ejecuta en otro tipo de dispositivo. El aislamiento del dispositivo gráfico desde la computadora con la línea serial produjo una definición de una interfaz clara y sencilla que facilitó la emulación. Los bitmaps de hoy día, en cambio, están enchufados directamente en el bus del sistema y la interfaz al sistema, que incluye la memoria de video y un sinnúmero de registros de control y atributos, es demasiado complejo para emular.

En los años sesenta se percibió la necesidad de estandarizar los paquetes gráficos para dar portabilidad a las aplicaciones gráficas. Un **estándar gráfico** define un conjunto de rutinas o funciones y su efecto sobre el dispositivo gráfico. Para cada lenguaje de programación que soporta, se define una **ligadura al lenguaje** en la que se especifican los nombres (y tipos) de las rutinas o funciones, y el número, el tipo y el orden de los parámetros de cada rutina.

La razón de la estandarización de los paquetes es para que una aplicación pueda correr en cualquier ambiente donde esté instalada una versión del paquete. Esto es lo que se conoce como un **paquete independiente del dispositivo**, un paquete que proporciona la misma interfaz a la aplicación independiente del dispositivo gráfico que usa. No significa que el código del mismo paquete es portable entre dispositivos o computadoras. Por supuesto esto tendría que modificarse para correr en distintos ambientes. Pero, modificando sólo el paquete se puede ganar la portabilidad de un sinnúmero de aplicaciones.

La ACM (Asociación de Maquinaria de Computación) desarrolló el estándar gráfico propuesto "CORE", finalizado en 1977, que fue ampliamente acatado por la industria de la gráfica por computadora. Basándose en el CORE, la ISO (Organización Internacional de Estándares) sacó en 1982 el "GKS" (Sistema de Núcleo Gráfico), un estándar más comprensivo y flexible que el CORE y actualmente el más acatado. GKS define cuatro niveles de compatibilidad de salida (1,2,3 y 4) y tres de entrada (A, B y C) tal que cada nivel sucesivo es un superconjunto del nivel anterior. Entonces permite que un paquete GKS-compatible sea implementado al nivel más adecuado para los objetivos y las capacidades de un sistema gráfico. Existe en GKS una portabilidad hacia arriba de aplicaciones tal que una aplicación puede correr en cualquier ambiente con una implementación de GKS con niveles de entrada y salida igual o mayor. Por ejemplo, una aplicación que requiere nivel 2B de GKS puede correr en cualquier otro sistema con GKS de nivel 2B, 3B, 4B, 2C, 3C ó 4C, pero no puede correr en sistemas con 1A, 2A, 3A ó 4A.

Hay mucho interés en desarrollar los estándares gráficos porque permitirán la "importación" de software gráfico entre sistemas gráficos. Sin embargo, las exigencias de eficiencia obligan a la mayoría de aplicaciones gráficas complejas a desconocer los estándares gráficos. Una tendencia esperanzadora que se nota en la industria de dispositivos gráficos es el diseño de dispositivos que toman en cuenta los requisitos de los estándares, facilitando la implementación de dichos estándares y hasta se están implementando en hardware o firmware algunas de las funciones de los estándares.

I.3E EL ESTANDAR "X WINDOWS"

El **Sistema X Windows** es un estándar gráfico independiente del hardware y del sistema operativo; diseñado para controlar un dispositivo gráfico de clase bitmap a través de una red o dentro de una sola máquina. En lugar de usar llamadas a procedimientos o al sistema como la mayoría de estándares gráficos, utiliza un protocolo asíncrono de redes. Fue desarrollado en el Massachusetts Institute of Technology en 1984 y ha llegado a ser un estándar de la industria de computadoras. Está implementado en casi todos los sistemas UNIX, además de varias otras máquinas. Debido al aumento de las redes y el procesamiento distribuido, X Windows va a cobrar mucha importancia en las aplicaciones gráficas del futuro. Las ventajas de X Windows pueden ser resumidas así:

*Es independiente del hardware, SO, lenguaje de programación y protocolo de red.
*Funciona independientemente del tipo de conexión, sea esta local o a través de una red, y hace a la red transparente al usuario y al programador de aplicación. Sólo requiere un flujo de bytes.
*En la mayoría de aplicaciones, no hay pérdida de velocidad ya que con una red como Ethernet el dibujo de las gráficas tiende a ser más lento que la trasmisión de comandos a través del protocolo.

Arquitectura Cliente-Servidor

La arquitectura de X Windows usa un modelo cliente-servidor parecido a las bases de datos distribuidas, pero con los papeles cambiados tal como en la figura I.4. Un **Servidor X** es un programa en el Sistema X Windows que controla la pantalla, reemplazando el software normal del sistema que maneja la pantalla. El servidor despliega texto, líneas, círculos, imágenes y otros elementos gráficos, según los pedidos (mensajes) recibidos de una aplicación. Un Servidor X puede manejar la salida gráfica de varias aplicaciones simultáneas. Típicamente, cada aplicación despliega información en una o más de sus propias ventanas en la pantalla.

Un programa de aplicación en el Sistema X Windows (tal como una hoja de cálculo o un programa de dibujo interactivo) que se comunica con un Servidor X se llama un **Cliente X**. Los Clientes X normalmente crean una o más ventanas en la pantalla para desplegar su salida. El servidor X puede recibir mensajes desde múltiples clientes X. Los clientes X son programados usando algunas bibliotecas para realizar la interfaz al Protocolo X, las que están actualmente disponibles para los lenguajes C y Lisp.

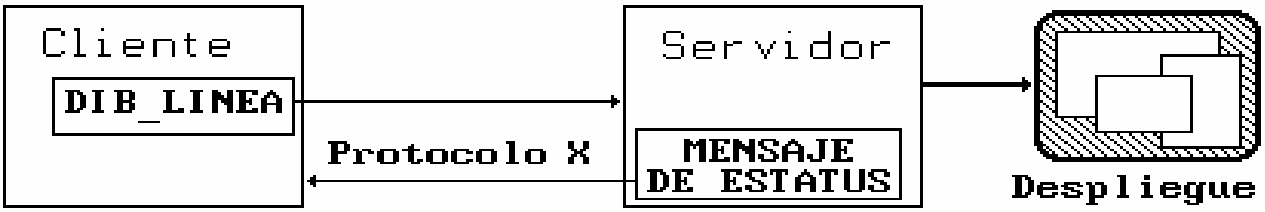

Fig. I.4 Comunicación en el Sistema X Windows: el cliente envia un comando tal como "dib_linea", el servidor regresa un mensaje de estado.

Una de las decisiones críticas de diseño en X Windows fue la división de capacidades entre el cliente y el servidor ya que este determina lo que tiene que ser pasado entre ellos. Se basó en

consideraciones de portabilidad de clientes, facilidad de programación de clientes y eficiencia. El servidor es responsable de todo dibujo y de la interfaz con el ratón y el teclado. También maneja fuera de la pantalla, bloques de memoria, fuentes de caracteres, ventanas, cursores y mapas de color. Hay un servidor prototipo suministrado por MIT que tiene una parte independiente del hardware y otra parte dependiente que tiene que ser adecuada para cada nuevo tipo de dispositivo gráfico.

Otra decisión crítica fue el modelo del dispositivo que se presentaría al cliente. Si se hubieran tratado de esconder, en el diseño del servidor, las diferencias entre los dispositivos, tales como la resolución y el número de colores, se habría hecho más fácil la programación de aplicaciones que requieren un tamaño determinado de pantalla, pero no se hizo así por que también habría dificultado operaciones tales como la manipulación de pixeles y del mapa de colores. Además, habría complicado el código del servidor. Entonces se optó por hacer más complicada la programación de clientes para hacerlo más poderoso.

Protocolo X para la Comunicación Cliente-Servidor

Las dos partes, cliente y servidor, están ligadas por medio de un sistema bien definido de mensajes llamado el **Protocolo X**. Este sistema de mensajes puede estar implementado a través de una red o dentro de una sola máquina. Cuando el cliente quiere producir salida gráfica en la pantalla, envía un mensaje al servidor, que describe la tarea a ejecutar. Hay mensajes para abrir una ventana, dibujar un círculo, imprimir una cadena de texto, etc. Al enviar un mensaje, el control regresa de inmediato al cliente sin esperar que la tarea su cumpla.

El servidor puede regresar mensajes al cliente para señalar un error o un evento (una entrada). También el cliente puede enviar un mensaje para interrogar al servidor sobre el estado o las características del dispositivo, a lo que el servidor responde con otro mensaje. Los mensajes del servidor también son parte del Protocolo X.

El hecho de que se comuniquen con mensajes permite que el Servidor X y el Cliente X estén en distintas máquinas en una red. Este es el propósito original de X Windows: ser un sistema gráfico de redes con independencia del hardware y del sistema operativo. Por ejemplo, teniendo una PC bajo MS-DOS con un servidor X, se puede tener una ventana que muestra la salida de un cliente corriendo en una máquina VAX bajo VMS y otra ventana que muestra la salida de un cliente que corre en una máquina Next bajo su propio sistema, mientras el manejador de ventanas corre en una máquina Sun bajo Unix. Y si la transmisión de la red es suficientemente rápida, ¡no te das cuenta que todo esto no está dentro de tu máquina! En este sentido X Windows es un gran avance en la portabilidad de aplicaciones gráficas y la independencia del hardware de despliegue.

La figura I.5 muestra un diagrama de una red con cuatro máquinas. La máquina A ejecuta un cliente que se despliega en la pantalla de la máquina C (usando el servidor de C). La máquina C tiene un cliente, pero este se despliega en la pantalla de la máquina B, que también despliega otro cliente que corre en la misma máquina. El segundo cliente de A se despliega en la pantalla en la máquina denominada Terminal X. Una Terminal X es una máquina que corre un Servidor X y nada más, siendo

dedicada al despliegue de gráficas generadas por Clientes X. Note que no es necesario que las máquinas sean compatibles.

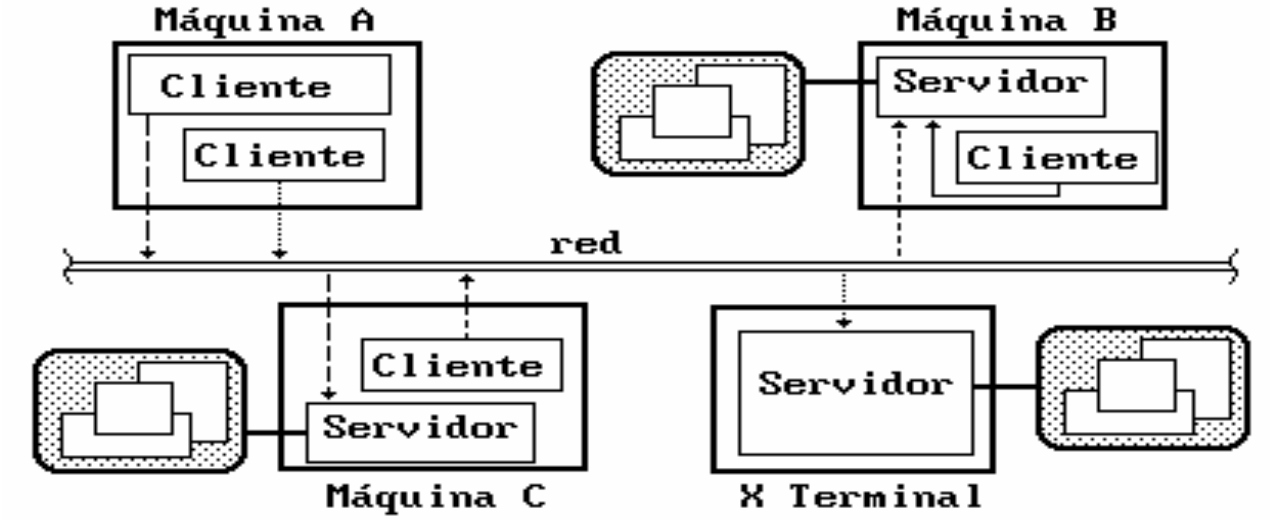

Fig. I.5 X Windows Corriendo en un Sistema Distribuido. El patrón de línea indica el flujo de los mensajes.

El Manejador de Ventanas

El servidor atiende solamente las requisas de Protocolo X, pero no proporciona funciones para permitir al usuario modificar el tamaño, posición u orden de apilamiento de las ventanas. Estas funciones podrían implementarse en cada cliente, pero resultaría mucha programación redundante. Podrían integrarse también en los servidores, pero X Windows opta por un diseño más flexible donde cada servidor tiene un cliente especial denominado el **manejador de ventanas** para cumplir estas funciones.

El manejador de ventanas, que se puede ejecutar local o externamente, tiene privilegios especiales que le permiten supervisar todas las ventanas desplegadas por el servidor. El manejador de ventanas despliega varios adornos en el marco de la ventana incluso el título y varios botones de control para modificar el tamaño, posición, etc. Son estos adornos los que distinguen a los distintos manejadores de ventanas. El contenido de la ventana está bajo el control del cliente, y no tiene que ver con el manejador de ventanas. Este permite que el usuario seleccione el manejador de ventanas que quiera independientemente del servidor y clientes que corre. La decisión de no especificar la interfaz al usuario en el estándar fue un factor importante en la aceptación de dicho estándar por fabricantes que ya tenían una interfaz propia que no hubieran querido cambiar.

I.4 LA APLICACION GRAFICA

Una **aplicación gráfica** es un programa completo que sirve para cumplir una tarea especifica mediante el uso de la gráfica por computadora. Algunos ejemplos de aplicaciones gráficas son los del diseño de un edificio, la preparación de un afiche o la exhibición de un esqueleto basada en datos de un rayo-X. Este folleto se ocupa con el desarrollo de aplicaciones gráficas mediante el uso de un paquete

gráfico pero no pretende enseñar el uso de las aplicaciones gráficas comerciales tales como el AutoCAD y el Harvard Graphics.

I.4A EL PROGRAMA DE APLICACION GRAFICA

La aplicación gráfica típica tiene tres componentes de software. Primero, está el componente del sistema gráfico que maneja el despliegue de la imagen, normalmente un paquete gráfico, pero a veces esto es realizado por un dispositivo gráfico inteligente. El segundo es la estructura de datos de la aplicación que guarda un modelo del objeto de la aplicación. El tercero es el programa de aplicación que usa el modelo para generar comandos al sistema gráfico.

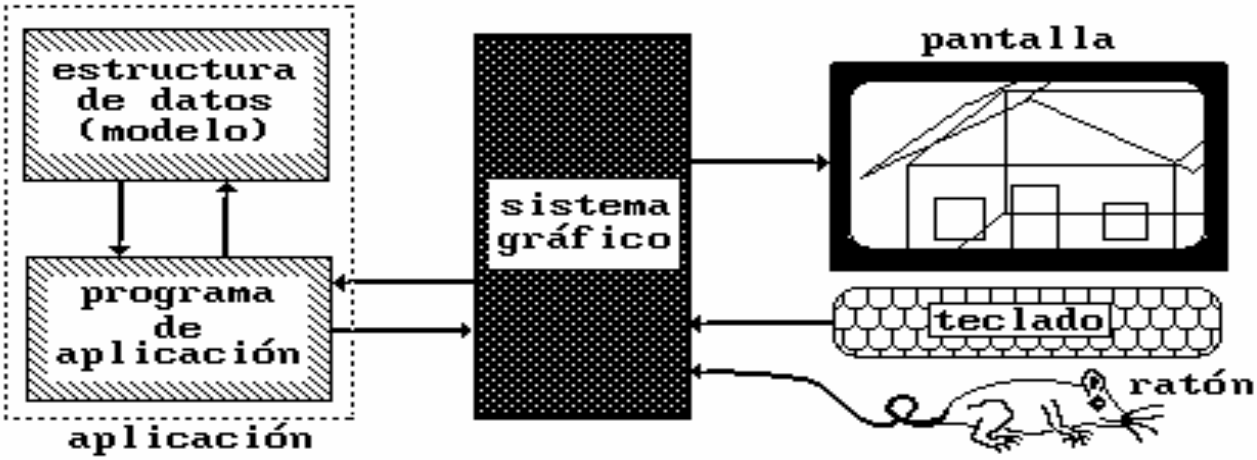

Fig. I.4 Modelo del Programador de La Gráfica Interactiva

La estructura de datos contiene un modelo de un objeto, que es una representación de los aspectos del objeto que son de interés a la aplicación. La descripción del objeto no es una descripción de una imagen, sino que describe muchas características geométricas y no geométricas. Por ejemplo, en una aplicación de diseño arquitectónico la estructura podría contener una lista de las paredes de una casa con la posición y tamaño de cada una. Además para cada pared habría una descripción de los postes estructurales, puertas y ventanas, todos con dimensión y posición. Además habría información sobre la materia y costo de cada elemento básico. Los tamaños y posiciones son datos geométricos. La materia y el costo son datos no geométricos, que a veces afectan la imagen del objeto, pero a veces son usados para obtener otra información tal como una cotización de las materias de la construcción.

El programa de aplicación usa el modelo para crear una vista del objeto que se expresa al sistema gráfico en términos de primitivos gráficos. Una **vista** significa que se genera una imagen que muestra una parte del modelo desde un punto de vista que destaca ciertas características del modelo. Se puede usar color y otros atributos para mostrar propiedades no geométricas. El programa de aplicación puede variar la visualización que genera, con la inclusión opcional de propiedades no geométricas. Lo importante es que la estructura de datos no contiene directamente los primitivos gráficos, sino una descripción de un objeto. El programa de aplicación usa esta descripción para generar los primitivos que representan visualmente ciertas partes del objeto.

Regresando al ejemplo de la aplicación arquitectónica, el programa puede exhibir el plan de un edificio desde distintos puntos de vista. Puede usar color para indicar el tipo de materia, por ejemplo si es de madera, concreto, hierro, etc. Las aristas que estarían ocultas pueden ser dibujadas con líneas puntadas. Puede exhibir solamente las paredes como sólidas, o puede exhibir los componentes estructurales dentro de las paredes. Así el programa iría obteniendo información de la estructura de datos y procesándola para generar primitivos gráficos que se envían al sistema gráfico.

El programa de aplicación se ocupa con tres actividades fundamentales. Primero, como fue antes señalado, está el trabajo de transformar la descripción del objeto a una imagen del objeto. Segundo, manejar la estructura de datos de la aplicación. Esto incluye la construcción y modificación del modelo que contiene. La estructura de datos puede ser un arreglo, una lista ligada, un archivo, una base de datos u otra cosa. Lo que es de interés aquí no es la forma de almacenamiento del modelo, sino su contenido. Tercero, tiene que manejar la interacción con el usuario. Por medio de un dispositivo de entrada, el usuario puede modificar la visualización del modelo, por ejemplo, escogiendo otra escala, punto de vista o la exhibición de otros detalles. También puede modificar el modelo.

I.4B EJEMPLOS DE APLICACIONES GRAFICAS

La Graficación de Datos Numéricos

Los conjuntos de datos numéricos y las funciones que se ocupan en las ciencias y la administración son más comprensibles presentados en la forma de una gráfica de barras o sectores, gráficas de producción o de inventario, gráfica cartesiana, o de otro tipo. Viendo una gráfica así se pueden deducir rápidamente los patrones o características de los datos.

La Cartografía

La gráfica por computadora es usada para preparar mapas de alta precisión sobre la geografía u otros fenómenos naturales. Además de los geográficos, hay mapas de tránsito, de vegetación, demográficos, meteorológicos, topográficos y otros.

El Diseño y el Dibujo Asistido por Computadora (CAD)

Sistemas de CAD son usados para diseñar componentes y sistemas de dispositivos mecánicos, eléctricos, y electrónicos. Tales sistemas incluyen edificios, plantas eléctricas y químicas, motores, automóviles, aviones, sistemas ópticos, redes telefónicas y computadoras. Algunos sistemas de CAD se enfocan solamente en la producción de dibujos para un sistema. Otros se enfocan en la realización de pruebas o simulaciones sobre el diseño, por ejemplo: las propiedades eléctricas, térmicas, estructurales o químicas. A veces "post-procesan" el diseño para sacar listas de componentes o materiales.

La Simulación y la Animación

La gráfica por computadora puede ser usado para estudiar el comportamiento de sistemas reales o teóricos. Permite al científico examinar los cambios sobre el tiempo de funciones matemáticas, y los modelos matemáticos de sistemas tales como flujos hidráulicos, reacciones químicas, distorsión de materiales bajo presión, y los

órganos humanos. También se pueden simular vuelos para entrenar pilotos, poniendo efectos como distintos tipos de terreno, otras aeronaves, nubes, viento, y las luces del aeropuerto de noche. La animación por computadora es también utilizada para hacer caricaturas, juegos de video, publicidades y videos informativos.

El Control de Procesos

Mientras la simulación permite al usuario interactuar con un mundo sintético, otras aplicaciones le permiten interactuar con el mundo real para monitorear y controlar procesos. Despliegues de estado para plantas químicas y eléctricas, refinerías, aeropuertos y redes de computadoras exhiben valores de datos registrados en partes críticas del sistema. Por ejemplo, en una planta química el despliegue puede mostrar cuán lleno es un tanque y usar color para mostrar su temperatura. Se pueden dibujar medidores para mostrar presión y volumen de flujo, y desplegar con intermitencia un mensaje cuando sucede una condición crítica. El supervisor del proceso puede ajustar válvulas o controles de temperatura que son controlados por la misma computadora. Otro ejemplo puede ser un aeropuerto, donde el despliegue de control de tráfico puede apuntar a la par de cada nave su número de vuelo, altura y velocidad, ayudando al controlador a manejar el tráfico. La mayor comprensibilidad de la información gráfica minimizaría los choques.

La Tipografía y la Automatización de Oficinas

Se puede preparar tipografía de documentos para la publicación en una pantalla gráfica, exhibiendo texto con distintas fuentes y tamaños de caracteres, integrando al texto dibujos sintéticos como las gráficas de barras o los dibujos arquitectónicos, e imágenes fotográficas grabadas con una cámara de video o un "scanner". Para la automatización de una oficina, se pueden preparar y difundir documentos electrónicos que incluyen elementos gráficos tales como un organigrama o gráfico de barras.

El Arte y la Publicidad

Se pueden preparar imágenes para comunicar un mensaje usando las técnicas de dibujo y pintura por computadora. Estos permiten la generación del arte de la publicidad con mayor rapidez y menor costo que los métodos manuales. También se usa la gráfica por computadora para preparar diapositivas para presentaciones científicas y educativas. El arte por computadora, a pesar del rechazo por algunos "puristas", está ganando reconocimiento como un medio legítimo de la expresión artística. En todos estos campos de la gráfica por computadora se nota que se están creando con mayor frecuencia imágenes que, en vez de transferirlas a otros medios tales como papel y película, son exhibidas en la misma pantalla de la computadora.