5.1.2 La representación gráfica en los interfaces de nodos, tramos y líneas de bus

En los interfaces de nodos, tramos y líneas de bus se usa el interfaz de representación gráfica pero con algunas modificaciones. Las principales diferencias con el interfaz de representación gráfica expuesto en la sección anterior son, en primer lugar, que aparecen dos imágenes en lugar de una, y de tamaño diferente. Estas dos imágenes representan vistas distintas de la red vial. La segunda diferencia es que al seleccionar un nodo o tramo sobre una de estas imágenes se ejecuta la función adecuada según el interfaz del que se trate (nodos, tramos o líneas de bus).

La representación gráfica en estos casos se consigue mediante una clase llamada *Informacion_pantalla*.

Este clase es muy importante porque se usa en todos los interfaces. Esta clase se encarga de crear las dos imágenes que aparecen en los interfaces en la parte derecha del mismo. La imagen superior contiene una visión general de la red vial y la inferior representa la porción de red vial seleccionada en ese instante. Asímismo, en la imagen superior aparece representada por un rectángulo de color y con la impresión de la red en color inverso, la porción de red vial seleccionada y que se está representando en la imagen inferior.

Una vez las imágenes han sido creadas adecuadamente, el control sobre ellas recae sobre el objeto de la clase *Contexto* de la que son una propiedad, es decir, *contexto* o *contexto_giro*.

Esto es en principio, ya que ambas imágenes disponen de controles que permiten variar su contenido. En concreto cada imagen dispone de un botón para realizar operaciones de zoom y otro para realizar operaciones de windowing. También disponen de un botón "restaurar". La pulsación de este botón hace que se represente en la imagen la red vial al completo.

Además la imagen inferior dispone de un botón "seleccionar" que ejecuta diferentes funciones según el interfaz en el que se encuentre el usuario. Así, si el usuario se encuentra se encuentra en el interfaz de nodos, podrá seleccionar un nodo sobre dicha imagen y se ejecutarán los métodos adecuados ante ese evento.

Esta es la clase *Informacion_pantalla*:

```
class Informacion_pantalla
{
      TWinControl *padre;
      TBevel *caja1,*caja2;
      int An,Al,Orx,Ory;
public:
      void Pinta_mapa(Contexto *context,TWinControl *formulario,
                        void __fastcall (__closure * funcion)(TObject *Sender));
      void __fastcall Evento_repintar (TObject *Sender);
      void Desactivacion_sel_nodo(Contexto *context,TWinControl *padre,bool
invisible);
};
```

El objeto *padre* de la clase *TWinControl* se usa para almacenar el objeto gráfico sobre el que se crearán las dos imágenes. Es el padre de las imágenes.

Los dos objetos *caja1* y *caja2* de la clase *TBevel* se usan para crear un pequeño pedestal sobre el que se dibujan las imágenes.

Las propiedades *An, Al, Orx* y *Ory* se usan para almacenar las propiedades geométricas de la imagen antes de llamar al método para luego poder devolverla a su estado inicial. Esto hace que el proceso sea totalmente transparente al usuario.

El método más importante es *Pinta_mapa*. Este método se encarga de representar los dos pedestales que contendrán las imágenes en el formulario que se le pasa como parámetro. Por supuesto lo hace usando el método de división del formulario en distintos porcentajes.

A continuación llama a los métodos *Crear_imagen_principal* y *Crear_imagen_auxiliar* del objeto *contexto_giro*. Estos métodos sirven cada uno para crear un objeto de la clase *TImage*, si bien el primero de ellos asigna la dirección de la imagen a la propiedad *imagen* del objeto *contexto_giro* y el segundo asigna la dirección de la imagen creada a la propiedad *imagen_aux* del mismo objeto. La imagen *imagen_aux* no se ve en pantalla. La propiedad *imagen* del objeto *contexto_giro* es la imagen superior de las dos.

Como se verá, lo que hace este método es usar la propiedad *imagen* del objeto *contexto* para representar la imagen inferior. Esta propiedad era la que contenía anteriormente la imagen donde se representaba toda la red vial.

También llama a los métodos adecuados de la propiedad mapa del objeto *contexto_giro* para calcular las variables de entorno para las nuevas imágenes.

A continuación hace que el objeto padre de la propiedad *imagen* del objeto *contexto* sea el formulario del interfaz en el que está actualmente el usuario (interfaz de nodos, de tramos o de líneas de bus). Seguidamente, modifica las propiedades geométricas de esta imagen para que se represente en el lugar que aparece en el formulario, es siempre la inferior. Esto quiere decir que la imagen inferior es la propiedad *imagen* del objeto *contexto.* Esto se consigue llamando al método *Actualizacion_imagen* de la clase *Contexto.*

A continuación recalcula las variables de entorno para el objeto *contexto* usando las dimensiones de la nueva imagen.

A continuación se representa la red vial en la imagen *imagen* del objeto *contexto_giro*. Esto hace que se muestre toda la red vial en la imagen superior.

Sin embargo, en la imagen inferior se usa el objeto *contexto* que hasta ahora había estado representando la red vial cargada. Eso quiere decir que los límites de la imagen inferior son los mismos que los que tenía la red vial antes de cargar el interfaz correspondiente. Por eso la imagen inferior representa la porción de red vial que el usuario tenía seleccionada previamente.

Finalmente, el área que se representa en la imagen inferior se traslada a coordenadas locales en la superior y se dibuja un rectángulo sobre esa zona usando al dibujar la constante *pmMergePenNot*. Esto implica que se dibuja el rectángulo con su color, tal cual, y lo que había bajo él se representa pero en color inverso, es este caso, blanco.

Para realizar estas operaciones de dibujo se usan varios métodos de la clase *Contexto*, aplicados al objeto *contexto_giro*.

Finalmente se actualiza la propiedad *padre* al valor adecuado.

El método *Evento_repintar* se usa para borrar la imagen inferior y representar en ella de nuevo la red vial al completo. Aunque, por supuesto, sin variar el tamaño de dicha imagen.

Finalmente, el evento *Desactivacion_sel_nodo* devuelve la propiedad *imagen* del objeto *contexto* a su estado original (usando de nuevo *Actualizacion_imagen*) y le asigna a de nuevo a todos los objetos gráficos el objeto padre adecuado.