

Definición del formato de gráfico de Aerotri. Abril de 2013, versión 04. El hecho de que el número de versión comience por un 0 indica que no es definitiva. No obstante la versión 04 ya es estable.

Se describe el formato de tres tipos de fichero: El fichero de gráfico en sí, que suele tener extensión .gra; el fichero de configuración, en el que se define la representación de los elementos así como el significado de los mismos, que suele tener extensión .cfg, y el fichero de formas, con extensión .fdf.

La mayor parte de las posibilidades de este formato todavía no están implementadas en el visor de gráficos de Aerotri, pero cualquier persona puede programar un visor/editor más potente.

FICHERO DE GRAFICO

Representación de datos

La unidad básica es el 4-byte, o sea, 4 bytes. Dentro de este, *b0*, *b1*, *b2*, *b3* indican los bytes que lo componen, comenzando en el más bajo, y *b01* y *b23* los 2-bytes bajo y alto. Dentro de un fichero deben aparecer siempre en el orden *b0,b1,b2,b3*, e.d., Little-Endian.

El fichero almacena tres tipos de datos : números enteros, números “reales” y cadenas de texto.

Los números enteros son siempre positivos y ocupan por lo general un 4-byte. En muchas ocasiones se empaqueta varia información en un 4-byte, en cuyo caso los enteros en él incluidos ocuparán un número de bits que en cada momento se especifica. En alguno de estos campos de bits la información es puramente cualitativa. Un uso muy común de los números enteros es para almacenar la posición de alguna información respecto a una posición base implícita (por ejemplo, el comienzo del fichero). Cuando se da el valor de un número entero en letra de máquina de escribir, como FFFF 0000, se está dando en base hexadecimal, cada carácter representando 4 bits.

Los llamados números reales pueden ocupar un 4-byte o bien dos. En el primer caso se almacenan según el tipo del mismo tamaño de la especificación IEC 60559, salvo por que no se permiten infinitos ni valores que no representen un número, y los números denormalizados pueden tratarse como si fuesen cero. Si ocupan dos 4-bytes se almacenan según el tipo de este tamaño de la referida especificación. De aquí en adelante se hará referencia a estos dos tipos de datos como *float* y *double* respectivamente.

Todos los bits a 1 en una coordenada (que son números float o double) indica que no existe. Solamente es posible en aquéllas coordenadas que indican incrementos respecto a otra coordenada. Los valores en los que esto es posible coinciden exactamente con los que son float o double según lo indicado en el bit de doble precisión (que se explica al describir la estructura de los elementos gráficos). A la hora de representarse se hará igual que si fuese 0 salvo que el visor sepa que para esos elementos debe aplicar una representación especial. Suele emplearse para información que es intrínsecamente bidimensional, almacenando este valor especial para la coordenada Z.

Las cadenas de texto comienzan siempre en el bit más bajo de un 4-byte y ocupan un número exacto de 4-bytes, para lo que se rellenará con bytes 0 si es necesario. Además, los 2 bytes más altos del último 4-byte deben ser 0, de manera que el final de una cadena de texto se pueda encontrar fácilmente con independencia de la codificación, para lo cual se añadirá un 4-byte entero a 0 si fuese necesario. A continuación se muestran varios ejemplos

de la longitudes de cadenas de texto, sin incluir ningún byte 0 de terminación, y el número de bytes que ocupa en el fichero, indicando entre paréntesis el número de 4-bytes :

5 → 8 (2)

6 → 8 (2)

7 → 12 (3)

8 → 12 (3)

Codificación de las cadenas de texto

Un código de codificación ocupa 1 byte, que será el *b3* del 4-byte que antecede al texto.

Salvo los textos de los elementos texto, que son elementos gráficos, las cadenas de texto no se representan o bien, para los nombres de elementos, se representan pero en el espacio píxel y sin permitir opciones complicadas como en el caso de los elementos texto, y su significado es más importante que su representación. Por ello la codificación tiene que dar cuenta, además de la transformación de los bytes de entrada a números, qué significa cada número. Se permiten los siguientes valores :

FF : Codificación de 8-bits por defecto de la máquina, sin transformación. Para trabajar dentro de un entorno en el que se emplea siempre la codificación por defecto del sistema operativo, despreocupándose de asuntos de codificación. Cuando el fichero se lleve a otra máquina la interpretación de los textos puede no ser correcta.

FE : Unicode, 2 bytes, sin transformación

FD : Unicode, UTF-16

FC : Unicode, UTF-8

01 : Windows occidental (cp 1252)

02 : Windows, Europa central (cp 1250)

03 : Macintosh estándar

04 : Macintosh, Europa central

Las posiciones siguientes están aún sin definir pero los candidatos son Windows cirílico, turco y báltico e ISO latin1, latin 2, latin 5 y latin 9. La serie de codificaciones que comienza en 01 son valores informativos. Un visor/editor puede no implementarlas y tratarlas como la codificación FF. Por tanto las únicas codificaciones que aseguran independencia respecto a la máquina y el visor son FE, FD y FC.

En los textos de los elementos texto se permite solamente las codificaciones FF, FE, FD y FC. Además algunos caracteres no se representan a sí mismos sino que actúan como códigos de control. Se explica al describir estos elementos.

Estructura del fichero

El fichero consta de una cabecera a la que sigue un índice que indica la posición de

ciertos elementos clave: otros índices y los diversos bloques de información. Todo el contenido del fichero tras la cabecera se estructura en elementos. La cabecera del fichero es como sigue:

4-byte 0:

b0: xxxx0000

xxxx: Número de decimales con que se deben dar las coordenadas, en la unidad de la coordenada Z (X,Y pueden tener un factor de escala que modifique este valor). 1111 (15) indica 15 o mayor.

b1: Libre

b2: 04 Versión

b3: BA (Si no es así se supone que no es un fichero de gráfico de Aerotri)

4-byte 1: Libre

4-bytes 2-13: Xmin, Xmax, Ymin, Ymax, Zmin, Zmax.

Los elementos se dividen en elementos gráficos: tipos 0--FFFFA, y elementos CCL: tipos FFFFB--FFFFE. El tipo FFFFF no se usa. Los elementos gráficos han de formar un bloque único, sin elementos de otro tipo intercalados: el bloque gráfico. Los otros dos bloques que hay definidos son el bloque de compuestos y el bloque de textos.

Los dos primeros 4-bytes de cada elemento son comunes a todos los tipos:

4-byte 0: Tipo y subtipo.

Los 20 bits más altos son el tipo; los 12 bits bajos, el subtipo

4-byte 1: 4-byte que ocupa este elemento

La posición de cada elemento dentro del fichero viene indicada en unos índices, como en seguida se verá, de manera que no es necesario que los elementos se sucedan inmediatamente uno tras otro. De hecho, es imposible acceder a los elementos secuencialmente ignorando los índices. Como excepción a esto, inmediatamente tras la cabecera ha de aparecer el elemento FFFFD-0, esto es, tipo FFFFD, subtipo 0.

Elementos CCL

Son los elementos cuyo tipo es FFFFE, FFFFD, FFFFC o FFFFB. De ahora en adelante se referirá a ellos como elemento CCL 14, CCL 13, CCL 12 o CCL 11.

El elemento CCL 11 (FFFB) es el elemento CCL de la aplicación. Nunca será definido. Las combinaciones de tipo y subtipo para los elementos CCL 12--CCL 14 que no se encuentren definidas a continuación están reservadas para futuras versiones y no deberán aparecer nunca en un fichero. En estos momentos se encuentran definidas las siguientes:

CCL 14: Sistema de coordenadas, con varios subtipos definidos

CCL 13:

Subtipo 0: Índice general. Indica la posición en el fichero de los elementos CCL

Subtipo 1 : Pares clave/valor
Subtipo 2 : Fichero de configuración insertado
Subtipo 3 : Tabla de elementos
Subtipo 4 : Tabla de compuestos
Subtipo 5 : Tabla de cadenas de texto
Subtipo 10 : Orden de los elementos
Subtipo 11 : Bloque de compuestos
Subtipo 12 : Bloque de textos
CCL 12 : Aún no hay ningún subtipo definido para este tipo

Elemento CCL 11

Elemento CCL de la aplicación. Este elemento nunca será definido y será ignorado siempre por el visor de AeroTri, pero está sujeto a las mismas restricciones que los otros elementos CCL; es decir, solamente pueden aparecer al principio o al final del fichero y tienen que aparecer indexados en el elemento CCL 13-0.

Elemento CCL 13-0: Índice general

Este elemento ha de ser obligatoriamente el primero tras la cabecera. Consta de un cierto número de pares de 4-bytes elemento/posición. el 4-byte *elemento* guarda el tipo y subtipo del elemento en cuestión, el 4-byte *posición* su posición en 4-bytes a contar desde el comienzo del fichero :

4-bytes 2-3 : Tipo y subtipo; posición
4-bytes 4-5 : Tipo y subtipo; posición
...
4-byte x -- x+1 : 0000 0000; Libre

Indica la posición de todos o algunos de los elementos CCL del fichero. Si la posición de alguno de ellos no está indicada aquí, entonces algún elemento CCL cuya posición si está indicada será a su vez un índice que recogerá la posición de aquél, o de otro que recogerá la posición de aquél, etc. Además puede, opcionalmente, haber una entrada con tipo-subtipo igual a FFFFF FFF. En esta entrada la posición indica el final lógico del fichero (el fichero puede incluir más bytes por cuestiones de alineación). Un elemento 0000 0000 indica el final de los pares y ha de aparecer obligatoriamente al final. Un elemento distinto de 0000 0000 cuya posición sea FFFF FFFF indica una entrada del índice borrada y es ignorado.

Elemento CCL 13-5: Tabla de cadenas de texto (*Obligatorio*)

4-byte 2 : Posición de la cadena 1
4-byte 3 : Posición de la cadena 2

...

4-byte n+1: Posición de la cadena n

4-byte n+2: 0000 0000

Cada vez que en esta definición del formato del fichero se diga que el contenido de un 4-byte es una «Referencia a texto» o «Referencia al nombre», dicho 4-byte contiene un entero k que quiere decir que el texto en cuestión es la cadena k -ésima de este elemento; es decir, que el texto se encuentra en la dirección indicada por «Posición de la cadena k ».

El bit más alto de cada 4-byte indica si se trata de un texto congelado o no (se explica enseguida). El resto de bits indican la posición en 4-bytes del texto desde el comienzo del bloque de textos (elemento CCL 13-12). Un 4-byte a FFFF FFFF indica una entrada eliminada. Ninguna referencia a texto podrá apuntar a esa entrada de la tabla.

Dos entradas distintas de esta tabla no deben apuntar al mismo texto (pero sí pueden apuntar a textos iguales).

Deben apuntar a una misma entrada de esta tabla referencias de textos que sean lógicamente el mismo, de manera que si se cambia el texto ese cambio afecte a todos los textos que comparten referencia, salvo que se trate de un texto congelado. Los textos congelados no se modifican, o más exactamente, su modificación está reservada al programa editor (p.e., por un cambio de idioma) y si se quiere cambiar un texto que apunta a un texto congelado deberá cambiarse la referencia. El texto congelado más común es la cadena vacía.

Una entrada a la que no apunte ninguna referencia y que no se trate de un texto congelado puede eliminarse. Por lo tanto ninguna aplicación puede guardar un fichero con ciertos textos preparados, sin congelar, y sin que todavía ninguna referencia apunte a ellos y suponer que otras aplicaciones lo van a respetar.

Elemento CCL 13-1: Pares clave/valor (*Opcional*)

Tanto la clave como el valor ocupan un número variable de 4-bytes. Las claves son cadenas de texto. El primer par clave/valor empieza en el *4-byte 2*. Cada bloque clave/valor se estructura de la siguiente manera:

4-byte 0: Número de 4-bytes que siguen = n

4-byte 1: Referencia al nombre de la clave

4-byte 2:

b0: Tipo de dato almacenado

0: enteros

1: floats

2: doubles

3: referencia a texto

b12: Número de elementos. En el caso de referencia a texto será igual a 1.

b3: Libre

4-bytes 3 -- ... : Los valores. En caso de referencia a texto será un único 4-byte con una referencia a texto

Para cada clave el *4-byte 0* no indica el tamaño de la clave/valor sino la distancia hasta el principio del siguiente par clave/valor. El tamaño será igual o menor.

Se recomienda almacenar la clave en Unicode puro (codificación FE). Un *4-byte 0* a 0 indica el fin de las claves. Una referencia al nombre de la clave igual a FFFF FFFF indica un par clave/valor eliminado y se ignora. Un número de elementos igual a 0 indica que la clave no tiene un valor asignado.

Elemento CCL 13-2: Fichero de configuración insertado (*Opcional*)

Los parámetros de representación gráfica de cada elemento (color, grosor ...) así como el significado de cada tipo y subtipo se definen en un fichero de texto. No obstante, es conveniente que los parámetros de representación se incluyan en el propio fichero de gráfico, de manera que el fichero almacene toda la información necesaria para la correcta representación gráfica de su contenido. Para ello se incluye este elemento, que no es más que un fichero de texto incrustado. Puede aparecer más de uno.

4-byte 2:

b01: libre

b2: libre

b3: Codificación del fichero, *no se aplica al nombre*.

4-bytes 3 -- 127: El nombre. Los b23 del último 4-byte han de ser 0

4-bytes 128 -- ... : El fichero. Los b23 del último 4-byte han de ser 0

El nombre es una cadena que identifica el fichero de configuración, de manera que pueden aparecer varios elementos CCL 13-2. Cada aplicación puede poner un límite al número de elementos CCL 13-2 que pueden aparecer en un fichero. Dicho límite ha de ser ≥ 15 . Los que tengan nombres iguales acumulan su contenido como si se tratase de un único fichero. El nombre no necesita ser un nombre de fichero válido, y cada visor lo interpretará según la codificación que prefiera, si es que lo llega a interpretar. La codificación del propio fichero es la indicada en el *4-byte 2, b3*.

Como los ficheros de configuración permiten hacer referencia a otros ficheros mediante un comando `\input`, es muy común emplear este elemento para almacenar la parte más variable de la configuración (representaciones gráficas) mientras que la parte más estable y que puede ser común a varios ficheros se almacena en un fichero real al que se hace referencia mediante un comando `\input`. El formato de estos ficheros se describe más abajo.

Elemento CCL 13-3: Tabla de elementos (*Obligatorio*)

Este elemento indica la posición en el fichero de cada elemento gráfico:

4-byte 2: Principio del bloque gráfico
4-byte 3: Tamaño del bloque gráfico
4-bytes 4-5: Posición del elemento 1; Elemento compuesto de que forma parte
4-bytes 6-7: Posición del elemento 2; Elemento compuesto de que forma parte
 ...
4-bytes $2n+2$ -- $2n+3$: Posición del elemento n ; Elemento compuesto de que forma parte
4-bytes $2n+4$ -- $2n+5$: 0000 0000; Libre

La posición de cada elemento se indica en 4-bytes desde el principio del bloque gráficos. Dicha posición se indica en el *4-byte 2*, en 4-bytes a contar desde el principio del fichero. El *4-byte 3* indica el tamaño del bloque gráfico para así poder copiar todo el bloque de una vez. La posición de un elemento no podrá ser 0, porque este valor especial se utiliza para indicar el final de la tabla. Por ello el bloque de elementos gráficos contendrá siempre al menos un 4-byte libre al principio.

Los elementos compuestos se empiezan a contar en 1. Si un elemento no forma parte de ningún elemento compuesto se indicará FFFF FFFF. Una posición igual a FFFF FFFF indica que el elemento no existe.

Cuando en algún otro punto (como se verá en los siguientes elementos) se hace referencia a un elemento gráfico mediante un número, por ejemplo «el elemento 7», ese número es el orden en esta tabla de elementos. Así, el elemento 7 es el indicado en el *4-byte 16* de este elemento.

Elemento CCL 13-4: Tabla de compuestos (*Opcional*)

4-bytes 2-3: Posición del e.c. 1; Elemento compuesto de que forma parte
4-bytes 4-5: Posición del e.c. 2; Elemento compuesto de que forma parte
 ...
4-bytes $2n$ -- $2n+1$: Posición del e.c. n ; Elemento compuesto de que forma parte
4-bytes $2n+2$ -- $2n+3$: 0000 0000; Libre

La posición de cada elemento compuesto se indica en 4-bytes desde el comienzo del elemento CCL 13-11 (bloque de compuestos).

Elemento CCL 13-10: Orden de los elementos (*Obligatorio*)

4-byte 2: Elemento primero
4-byte 3: Elemento segundo
 ...
4-byte $n+1$: Elemento n -ésimo
4-byte $n+2$: 0000 0000

Este elemento indica el orden en que se van dibujando los elementos, de manera que los primeros quedan por debajo de los últimos. A cada elemento se hace referencia mediante un número, que es su orden en la tabla de elementos. Un número de elemento FFFF indica una entrada borrada y se ignora.

El orden de dibujo de los elementos que forman parte de un elemento compuesto no es posible respetarlo, salvo que el elemento CCL 13-10 se mantenga constantemente actualizado. Los elementos que conforman un compuesto deben dibujarse todos en el orden que corresponda a uno de ellos (cuál de ellos es algo que se deja a la elección de cada visor) y respetando el orden relativo de los mismos.

Elemento CCL 13-11 : Bloque de compuestos (*Opcional*)

La estructura de cada elemento compuesto es como sigue :

4-byte 0 : Número de elementos que lo forman = n

4-byte 1 : Primer elemento

4-byte 2 : Segundo elemento

...

4-byte n : elemento n -ésimo

4-byte $n+1$: Elemento principal

Los elementos son números que indican el número de elemento en la tabla de elementos o en la tabla de compuestos. Si el bit más alto es un cero se trata de un elemento simple, si es 1 se trata de un elemento compuesto. El resto de bits es el número del elemento en su tabla correspondiente. El elemento principal es aquél cuyas propiedades se aplicarán a todo el compuesto cuando sea necesario. Por ejemplo el significado o el nombre. Si no existe ningún elemento principal este campo se dejará a FFFF FFFF.

La posición de cada elemento compuesto dentro del bloque de compuestos está indicada en la tabla de compuestos. Un elemento compuesto al que no apunte ninguna entrada de la tabla de compuestos no es más que espacio libre dentro del bloque de compuestos.

Todos los elementos que forman un elemento compuesto se seleccionan como si fuese uno solo, y así se mueven, copian, deforman, se cambian propiedades, etc. Se dibujan todos uno a continuación de otro, de manera que otro elemento no puede estar dibujado por encima de uno de los elementos que forman un elemento compuesto y por debajo de otro. Las composiciones de elementos pueden hacerse y deshacerse.

Elemento CCL 13-12 : Bloque de textos (*Obligatorio*)

El formato de cada texto es el siguiente :

4-byte 0 :

b01 : 4-bytes que ocupa, incluyendo los bytes 0 del final = m

b2: reference count
b3: codificación
4-bytes 1 -- m: El texto

La posición de cada texto se indica en la tabla de cadenas de texto. Textos a los que no apunte ninguna entrada de la tabla no son más que espacio libre dentro del bloque de textos.

Elemento CCL 14: Sistema de coordenadas (*Opcional*)

Este elemento especifica el sistema elipsóidico de coordenadas en el que se encuentran las coordenadas del fichero. El subtipo indica el tipo de sistema de coordenadas y el resto del elemento incluye los parámetros necesarios para su definición. Si este elemento no aparece se supone que el sistema es «Rectangular». Los posibles subtipos son :

0: Rectangular
1: Conforme Genérico
2: Lambert (y Mercator)
3: Estereográfica
4: UTM
5-9: Reservado para otras proyecciones conformes
10: Geográficas
11: Ortogonal Genérico (meridianos y paralelos son ortogonales en la proyección, pero la proyección no es conforme)
12-19: Reservado para proyecciones ortogonales

La estructura del elemento depende del subtipo. La indicación (*Opc.*) significa que el campo puede ser desconocido, en cuyo caso todos los bits estarán a 1.

0 Rectangular

Nada más. El último 4-byte es el *4-byte 1*, que estará a 2.

1 Conforme Genérico

4-byte 1: = 28
4-bytes 2-3: X_0
4-bytes 4-5: Y_0
4-bytes 6-7: Offset Z (ondulación del geoide): Altura de los puntos con $Z=0$ en relación a la superficie de referencia para las coordenadas planimétricas
4-bytes 8-9: Radio de curvatura mayor N
4-bytes 10-11: Radio de curvatura menor ρ
4-bytes 12-13: $\cos \gamma_0$. γ_0 = «Convergencia de meridianos», ángulo que forma el eje Y del sistema en el punto (X_0, Y_0) con el Norte geográfico. Positivo si el eje está hacia el Este.
4-bytes 14-15: $\sin \gamma_0$.

4-bytes 16-17: k_0 . Factor de escala por el que están afectadas las coordenadas planimétricas en relación a las altimétricas, en el punto (X_0, Y_0)

4-bytes 18 -- 27: Coeficientes a_1 - a_5 del desarrollo en serie de k :

$$k = k_0 + a_1(X - X_0) + a_2(Y - Y_0) + a_3(X - X_0)^2 + a_4(X - X_0)(Y - Y_0) + a_5(Y - Y_0)^2 + \dots$$

La comprobación para ver si $\gamma_0 \neq 0$ tiene que ser $\sin \gamma_0 \neq 0$; no debe ser $\cos \gamma_0 \neq 1$.

Esta sistema de coordenadas permite imitar una proyección conforme cualquiera, pero también puede aparecer cuando se sabe que las coordenadas siguen una cierta proyección pero no se sabe cuál. En este caso los coeficientes de k estarán todos a 0, así como γ_0 , y probablemente también se tenga $k_0 = 1$.

2 Lambert

La proyección Mercator es una Lambert con $\varphi_0 = 0$.

4-byte 1: = 17

4-bytes 2-3: a (radio equatorial)

4-bytes 4-5: e^2 (1ª excent. al cuadrado)

4-bytes 6-7: Ondulación

4-bytes 8-9: Latitud central (latitud de escala mínima). En grados sexagesimales

4-bytes 10-11: k_0 . Factor de escala en el paralelo central

4-bytes 12-13: Desplazamiento X

4-bytes 14-15: Desplazamiento Y

4-byte 16: Meridiano central, en grados sexagesimales (*Opc.*)

3 UTM

4-byte 1: = 15

4-bytes 2-3: a

4-bytes 4-5: e^2

4-bytes 6-7: Ondulación

4-bytes 8-9: k_0 . Factor de escala en el meridiano central

4-bytes 10-11: Desplazamiento X

4-bytes 12-13: Desplazamiento Y

4-byte 14: Meridiano central, en grados sexagesimales (*Opc.*)

4 Estereográfica

4-byte 1: = 15

4-bytes 2-3: Radio polar ($= a^2/b$)

4-bytes 4-5: e^2

4-bytes 6-7: Ondulación

4-bytes 8-9: k_0 . Factor de escala en el meridiano central

4-bytes 10-11: Desplazamiento X

4-bytes 12-13: Desplazamiento Y

4-byte 14: Meridiano seguido por el sentido positivo del eje Y, en grados sexagesimales. Debe ser negativo si estamos en el Polo Norte ($-0 - -360$) y positivo en el Sur ($0-360$). (*Opc.*)

10 Geográficas

Las coordenadas de los elementos del fichero han de ser grados sexagesimales.

4-byte 1: = 9

4-bytes 2-3: a

4-bytes 4-5: e^2

4-bytes 6-7: Ondulación

4-byte 8: El bit más bajo indica si orden de latitud-longitud. = 0 si el orden es ϕ, λ (latitud, longitud); = 1 si es λ, ϕ .

11 Ortogonal genérico

4-byte 1: = 30

4-bytes 2-3: X_0

4-bytes 4-5: Y_0

4-bytes 6-7: Ondulación

4-bytes 8-9: φ_0

4-bytes 10-11: Radio N

4-bytes 12-13: Radio ρ

4-bytes 14-15: $\cos \gamma_0$

4-bytes 16-17: $\sin \gamma_0$

4-bytes 18-19: k_0^x . Factor de escala de las coordenadas X en el punto (X_0, Y_0) , en relación a la Z.

4-bytes 20-21: k_0^y . Idem de las Y.

4-bytes 22 -- 29: Coeficientes de k : $k^x = k_0^x + a_1(Y - Y_0) + a_3(Y - Y_0)^2 + \dots$
 $k^y = k_0^y + a_2(Y - Y_0) + a_4(Y - Y_0)^2 + \dots$

Elementos gráficos

Son los elementos cuyo tipo es FFFFA o menor. El tipo FFFFA es el elemento gráfico de la aplicación. Se trata de un elemento cuyo significado es conocido por la aplicación. Su formato debe coincidir con el de los elementos gráficos en los *4-bytes 0--16*, excepto por el *4-byte 3*, en donde solamente ha de coincidir en su bit más alto (el bit de borrado). El resto es interpretado libremente por la aplicación y será ignorado por AeroTri.

En las secciones siguientes se especifica el formato para cada clase de elemento. Tras el último 4-byte indicado vendría la información adicional en caso de que exista. Esto ya no se repite en cada especificación.

Parte común

4-byte 2: Posición de la información adicional desde el comienzo del elemento, en 4-bytes

4-byte 3:

b01: Libres

b23: Un bit 1 en la posición más alta significa que el elemento se ignora. El siguiente bit indica si ciertos campos están almacenados con simple o doble precisión, y además si se representan según la proyección o en la unidad de la coordenada Z:

0: simple precisión, unidad de Z

1: doble precisión, proyección

Los 4 bits más bajos indican el tipo de información geométrica almacenada, a lo que se llamará clase de elemento, y se dirá indistintamente «clase polígono» o «clase 2», por ejemplo. Para todos los elementos de un mismo tipo debe tener el mismo valor. Las posibles clases se indican más abajo.

4-bytes 4--15: Xmin Xmax Ymin Ymax Zmin Zmax

4-byte 16: Referencia al nombre. Si todos los bits están a 1, e.d. FFFF FFFF, el elemento no tiene nombre.

Si el *4-byte 2* es igual que el *4-byte 1* (tamaño del elemento) entonces no existe información adicional. En cualquier caso el *4-byte 2* es siempre el tamaño de la parte gráfica del elemento. A partir de la posición indicada en el *4-byte 2* y hasta completar el tamaño indicado en el *4-byte 1* se encuentra la información adicional, cuyo formato se describe más abajo.

Los posibles valores para la clase de elemento, que se escribe en los 4 bits más bajos del *4-byte 3, b23*, son los siguientes:

0: punto

1: línea poligonal

2: polígono y centro

3: segmento escalable (origen y vector)

4: línea poligonal escalable (sucesión de pares origen-vector)

5: elipse

6: elipsoide

7: spline

8: texto plano

9: texto tridimensional

20: elemento complejo (célula)

El resto de valores está reservado para futuras versiones. Para las clases de la 3 a la 6 el bit de simple/doble precisión arriba mencionado se refiere a los desplazamientos, σ_x ,

etc. (todo menos las coordenadas del origen).

Nombres de elementos

Los números almacenados en un nombre de elemento, tras decodificarlo si es necesario (p.e. UTF-8) no son simples posiciones de una fuente sino entidades con significado de acuerdo a una página de códigos: Unicode, Windows cp1252... y ha de tenerse en cuenta las distintas codificaciones a la hora de compararlos entre sí o de escribirlos a otro fichero con una determinada codificación. No se permite que incluyan códigos que representen comandos, como indicaciones de dirección, "escapes", etc., ni otros espacios que el normal de la codificación. Si apareciese alguno de los primeros debe ser ignorado a la hora de representar el texto, mientras que uno de los segundos debe representarse como si fuese un espacio normal. Se recomienda que no incluya ligaduras ni formas finales, etc. La razón de ello es que el nombre almacena conceptualmente el nombre del elemento, no la representación gráfica del mismo, y el empleo de estas formas podría llevar a no reconocer como iguales nombres que sí lo son, o a escribirlos incorrectamente en ficheros de texto u otros ficheros.

Clase 0 (punto)

4-bytes 17--22: X, Y, Z

Clase 1 (línea poligonal)

4-byte 17:

b01: Número de vértices = n

b23: Libres

4-bytes 18-23: X, Y, Z del primer punto

...

4-bytes 18+6(n-1) -- 18+6n-1: X, Y, Z del último punto

Clase 2 (polígono y centro)

4-bytes 17-22: X, Y, Z del centro

4-byte 23:

b01: Número de vértices = n

b23: Libres

4-bytes 24-29: X, Y, Z del primer punto

...

4-bytes 24+6(n-1) -- 24+6n-1: X, Y, Z del último punto

El último punto podrá o no ser igual que el primero; es decir, el primer punto podrá o no estar repetido al final. Nótese que el centro puede ser cualquier punto del espacio.

El significado de cada uno de estos elementos limitará en su caso la posible posición del centro, pero el formato del fichero no establece ninguna limitación.

Clase 3 (segmento escalable)

a) con bit de doble precisión a 0

4-bytes 17--22: X, Y, Z del origen

4-bytes 23--25: dx , dy , dz

b) con bit de doble precisión a 1

4-bytes 17--22: X, Y, Z del origen

4-bytes 23--28: dx , dy , dz

Clase 4 (línea poligonal escalable)

a) con bit de doble precisión a 0

4-byte 17:

b01: Número de vértices = n

b23: Libres

4-bytes 18--23: X, Y, Z del primer punto

4-bytes 24--26: dx , dy , dz del primer punto

...

4-bytes $18+9(n-1)$ -- $18+9n-1$: X, Y, Z, dx , dy , dz del último punto

b) con bit de doble precisión a 1

4-byte 17:

b01: Número de vértices = n

b23: Libres

4-bytes 18--23: X, Y, Z del primer punto

4-bytes 24--29: dx , dy , dz del primer punto

...

4-bytes $18+12(n-1)$ -- $18+12n-1$: X, Y, Z, dx , dy , dz del último punto

Clase 5 (elipse)

a) con bit de doble precisión a 0

4-bytes 17--22: X, Y, Z del centro

4-bytes 23--25: Coordenadas de un extremo del eje mayor respecto al centro

4-bytes 26--28: Coordenadas de un extremo del eje menor respecto al centro

b) con bit de doble precisión a 1

4-bytes 17--22: X, Y, Z del centro

4-bytes 23--28: Coordenadas de un extremo del eje mayor respecto al centro

4-bytes 29--34: Coordenadas de un extremo del eje menor respecto al centro

Clase 6 (elipsoide)

a) con bit de doble precisión a 0

4-bytes 17--22: X, Y, Z del centro

4-bytes 23--28: Media matriz :

23: σ_{xx}

24: σ_{xy} **25:** σ_{yy}

26: σ_{xz} **27:** σ_{yz} **28:** σ_{zz}

b) con bit de doble precisión a 1

4-bytes 17--22: X, Y, Z del centro

4-bytes 23--34: Media matriz :

23-24: σ_{xx}

25-26: σ_{xy} **27-28:** σ_{yy}

29-30: σ_{xz} **31-32:** σ_{yz} **33-34:** σ_{zz}

Clase 7 (spline)

Por definir, pero está definido que habrá las dos variantes según el bit de doble precisión.

Definición de un plano

Para la clase 8 es necesaria la definición de un plano y un sistema de coordenadas en él, que se lleva a cabo mediante tres puntos: p_1 , p_2 y p_3 . Se dan las coordenadas (X, Y, Z) de p_1 y las coordenadas de p_2 y p_3 respecto a p_1 , o lo que es lo mismo, los vectores $p_2 - p_1$ y $p_3 - p_1$. El punto p_1 se toma como origen de coordenadas del sistema (x, y) del plano y se llama punto de referencia, R.

Si el módulo del vector (dX, dY, dZ) de los valores indicados para p_2 es a , las coordenadas de p_2 en el plano son (a, 0), y sea $(x_2, y_2, z_2) = (dX, dY, dZ)/a$. Sea (x_3, y_3, z_3) el vector (dX, dY, dZ) de los valores indicados para p_3 y b su módulo. Las coordenadas en el plano de p_3 son (c, d) con $c = x_3x_2 + y_3y_2 + z_3z_2$ y $c^2 + d^2 = b^2$. Esto define la transformación de coordenadas del plano al sistema (X, Y, Z) y viceversa. En concreto, si llamamos (x_4, y_4, z_4) al vector $(x_3, y_3, z_3) - c(x_2, y_2, z_2)$ dividido por su módulo, las coordenadas tridimensionales de un punto (x, y) del plano son $R + (X, Y, Z)$, con

$$X = xx_2 + yy_4, \quad Y = xy_2 + yy_4, \quad Z = xz_2 + yz_4,$$

Si esto se aplica a un elemento cuyo bit de doble precisión está a cero las coordenadas (X, Y, Z) son en un sistema local en las unidades del eje Z, como siempre.

Clase 8 (texto plano)

El bit de doble precisión estará obligatoriamente a 1.

4-bytes 17--34: Definición del plano :

17--22: X, Y, Z de $p_1 = R$

23--28: dX, dY, dZ de p_2 respecto a p_1 .

29--34: dX, dY, dZ de p_3 respecto a p_1 .

4-byte 35:

b0: Modo de posición :

0: Punto, incremento y justificación

1: Línea poligonal y distancia

2: Punto, radio y ángulos inicial y final

b1: Flags que indican el modo de situar los caracteres :

bit 0: Un 0 indica que se sitúa el texto según el rectángulo contenedor y un 1 según el rectángulo de alineación. Ambos conceptos se explican más abajo.

bit 1: Se aplica al modo *Línea poligonal y distancia*. Un 0 significa que primero se sitúan los caracteres sobre la línea y luego se desplazan la distancia indicada; un 1 significa que primero se calcula la línea desplazada y a continuación se sitúan los caracteres sobre ella.

b23: Libres

4-byte 36:

b01: Longitud en 4-bytes del texto, incluyendo los bytes 0 del final = m

b2: Libre

b3: Codificación

4-bytes 37 -- 37+m-1: El texto

4-bytes 37+m -- 37+m+x: Parámetros para definir la posición del texto dentro del plano. El número de 4-bytes que esto ocupa, x , depende del modo de posición.

La razón de que la información geométrica se sitúe tras el propio texto y no antes es que, al ser de longitud variable, si se añaden nuevos modos de posición en el futuro haría el texto inalcanzable para visores anteriores.

Antes de describir los 4-bytes que restan para cada modo de posición conviene definir algunos términos. La descripción de esta clase se completará más adelante.

Interpretación de los caracteres del texto

Los caracteres no son los bytes de entrada, si no el resultado de convertir estos mismos de acuerdo a la codificación. Lo que se explica en esta sección se refiere siempre a caracteres. Sin que suponga una excepción a esto, un **b23** a cero siempre indica que el texto ya se ha terminado.

Todos los caracteres se representan a sí mismos, es decir, indican el carácter de la fuente que tiene ese número, excepto los siguientes :

0: Un 0 indica el final de cadena.

1 : Carácter de control. Forma una pareja con el número siguiente, que nunca será 0 o 1, ni siquiera en futuras versiones del formato de gráfico. Las siguientes parejas son válidas en la versión actual :

1, 2 : Carácter 0 de la fuente. 1, 3 : Carácter 1 de la fuente. 1, 4 : El espacio normal entre palabras. Existe este comando porque algunas fuentes no incluyen el espacio entre sus caracteres, pues al fin y al cabo no lo es, pero definen la longitud del mismo. En las que sí lo incluyan, este comando es lo mismo que haber escrito el número del carácter espacio.

1, '66 hasta 1, '77 (octal) : Comandos de dirección. Se explican un poco más abajo.

Clase 9 (texto tridimensional)

Por definir.

Presentación gráfica de los textos

Esta sección se ocupa de los elementos texto (clases 8 y 9). Al final se incluye un apartado para la representación de los nombres de los elementos.

Tipos de letra

El fichero de configuración puede definir fuentes virtuales basadas en fuentes reales, llevando a cabo ciertas modificaciones. Desde el momento en que está definida se trata de una fuente distinta más, que de cara al dibujo de un texto no se distingue en nada de una fuente original.

Definiciones

Cada carácter de una fuente tiene asociada una *caja* que lo contiene. Los bordes de las cajas son siempre horizontales y verticales, aunque los caracteres no sigan esa dirección (p.e., un tipo de letra cursiva o inclinada). La forma real de un carácter puede salirse fuera de la caja, pero esto es irrelevante para lo que sigue. De hecho, la forma real del carácter no se emplea hasta la hora de imprimirlo (mostrarlo), en ningún momento se emplea en el cálculo de su posición. Los caracteres se sitúan uno a continuación de otro sin más que situar adyacentes sus cajas correspondientes. Esta separación entre cajas podrá estar modificada por información de la fuente, el llamado *kerning*. El fichero de gráfico no es un *código fuente* ni las aplicaciones que lo leen son procesadores de texto, de manera que no se lleva a cabo ninguna lógica de reemplazo de unos caracteres por otros. Si se quieren emplear ligaduras o las varias formas de una letra han de aparecer explícitamente en la cadena de texto.

La caja tiene definido un sistema de coordenadas con el eje X horizontal y hacia la derecha y el eje Y vertical y hacia arriba. Si la fuente no indica nada al respecto el origen de este sistema de coordenadas ha de tomarse en la esquina inferior izquierda de la caja. En una fuente que incluya letras latinas minúsculas, por ejemplo, el origen que la fuente indique para la letra q estará en el borde izquierdo de la caja pero no en la parte inferior, sino a la altura de la parte inferior del ojo, de manea que los puntos del palo que sobresale hacia abajo tienen coordenadas Y negativas. Además del origen habrá otro punto de referencia opuesto, que si la fuente no indica nada se tomará en el extremo superior derecho de la caja. Las líneas paralelas a los ejes que pasan por ese punto junto con los propios ejes constituyen las alineaciones de la caja: dos alineaciones horizontales, una superior y una inferior, y dos alineaciones verticales, una izquierda y una derecha. Las cuatro intersecciones de estas líneas son los cuatro puntos de alineación: inferior izquierda, *ii*, es el origen de coordenadas; inferior derecha, *id*, el otro punto sobre el eje X; superior izquierda, *si*, el otro punto sobre el eje Y, y superior derecha, *sd*, el punto de alineación opuesto al origen. Nótese que no se obliga a que la caja del carácter contenga el origen de coordenadas ni el punto *sd*, ni a que las coordenadas del punto *sd* sean positivas, no obstante lo cual se empleará siempre la terminología derecha, izquierda, arriba, abajo como se ha hecho en este párrafo.

Hay un total de ocho posibilidades para la dirección de escritura y alineación de caracteres. En primer lugar hay cuatro direcciones posibles de escritura, según la caja de cada carácter se sitúe a la izquierda, a la derecha debajo o encima de la anterior. Las dos primeras son escrituras *horizontales* y las dos segundas *verticales*. En segundo lugar hay dos posibilidades para la alineación de las cajas: para las escrituras horizontales pueden situarse las cajas de manera que queden alineadas según la alineación inferior (el eje X) o bien según la superior, mientras que para las verticales pueden situarse de manera que queden alineadas según la alineación izquierda (eje Y) o la derecha.

Nótese que en ningún momento se habla de girar o reflejar el carácter. Las cajas se situarán siempre con el eje Y apuntando hacia arriba y el eje X hacia la derecha, salvo por caracteres que necesiten ser reflejados, como un paréntesis de apertura. Cómo situar estos caracteres se explicará más abajo.

Dirección de escritura

Los comandos de dirección se dividen en cuatro comandos de alineación, 1, '71 -- 1, '74, tres comandos de sentido, 1, '75 -- 1, '77, y tres comandos de reflejo, 1, '66 -- 1, '70. Los comandos de sentido se dividen a su vez en comandos de sentido fuerte, los 1, '75 y 1, '76, y un comando de sentido débil: 1, '77. Un comando de alineación solamente puede aparecer como primer número de la cadena (por tanto sólo puede haber uno) y debe ir inmediatamente seguido por uno de los comandos 1, '75 o 1, '76, como se explica a continuación. Los otros comandos pueden aparecer en cualquier parte de la cadena.

Un texto puede comenzar con comandos que indiquen explícitamente la dirección y

alineación de escritura. Para ello se emplean dos comandos. En primer lugar debe aparecer un comando de alineación que indica según cuál de las cuatro alineaciones se alinean las cajas, y a continuación uno de los dos comandos 1, '75 y 1, '76, que indica el sentido de escritura :

- 1, '71 : Alineación inferior (eje X)
- 1, '72 : Alineación superior
- 1, '73 : Alineación izquierda (eje Y)
- 1, '74 : Alineación derecha

Por tanto los dos primeros implican una escritura horizontal mientras que los dos siguientes suponen una escritura vertical.

- 1, '75 : Para escritura horizontal significa que a partir de este punto todos los caracteres se han de escribir de izquierda a derecha; para escritura vertical, de arriba a abajo.
- 1, '76 : Para escritura horizontal significa que a partir de este punto todos los caracteres se han de escribir de derecha a izquierda; para escritura vertical, de abajo a arriba.

Los comandos 1, '77, 1, '66 -- 1, '70 se explican más abajo.

La dirección y alineación en la que ha de comenzar a escribirse un texto se determina por los siguientes puntos, de mayor a menor prioridad :

1. Si el texto empieza con un código de alineación, la dirección será la definida por ese comando y el siguiente.
2. La dirección indicada para este elemento en el fichero de configuración, de acuerdo a su tipo, subtipo y otros parámetros, como se explicará al describir ese fichero.
3. La dirección indicada para esa fuente en el fichero de configuración.
4. Si la fuente indica una dirección para cada carácter, o bien si no lo hace pero se conoce la codificación de la fuente, será la dirección inherente al carácter. Por ejemplo, una letra griega implica una dirección de izquierda a derecha y alineación inferior. El primer carácter anterior a un comando de sentido fuerte que implique (el carácter) una dirección define la dirección del texto desde el principio. Este criterio no se puede aplicar por tanto si el texto consta únicamente de caracteres neutros, por ejemplo asteriscos, o si tales caracteres son los únicos antes del primer comando de sentido fuerte.
5. La dirección por defecto que emplee el sistema operativo, cuando este dato esté disponible. Así, en un sistema operativo en hebreo, se escribirá de derecha a izquierda con alineación inferior.
6. Si todo lo anterior falla se comenzará a escribir de izquierda a derecha con alineación inferior.

En caso de que uno de estos puntos permita decidir el sentido pero se desconozca la alineación se empleará alineación inferior para direcciones horizontales e izquierda para verticales.

Una vez definida la dirección y alineación de comienzo para el texto, esta última permanece fija y solamente se permite cambiar la dirección. La dirección de comienzo se llamará la dirección *original* del texto.

Además de la dirección de escritura hay en cada momento un *modo*, que puede ser fuerte o débil. En el modo fuerte el sentido de escritura solamente se puede cambiar mediante un comando. En el modo débil el sentido de escritura de cada carácter es el que defina la fuente para ese carácter, bien de manera explícita, bien de manera implícita a través de la codificación de la fuente. Si ninguna de esta información está disponible no hay diferencia entre modo fuerte y débil. En modo débil, la aparición de un carácter que deba escribirse en el sentido contrario al que se está siguiendo cambia el sentido de escritura para ese carácter y los que sigan hasta que otro carácter o un comando vuelva a cambiar el sentido (por lo tanto, caracteres verticales en una escritura horizontal o viceversa no cambian el sentido).

Si el criterio que permitió definir el sentido inicial fue alguno de los tres primeros se comienza en modo fuerte y nunca se abandonará este modo; si fue alguno de los siguientes se comienza en modo débil.

Para cambiar la dirección se pueden emplear los comandos 1, '75, 1, '76, cuyo significado ya se explicó, y a lo que sólo cabe añadir que además hacen que se entre en modo fuerte si es que no se estaba ya, o el comando 1, '77:

- 1, '77: Si la dirección inicial se decidió por alguno de los tres primeros criterios este comando hace que se vuelva al sentido original y se continúa en modo fuerte. En caso contrario: a) En modo débil es ignorado. b) En modo fuerte hace que se entre en modo débil y se vuelva a la orientación de la última vez que se estuvo en modo débil.

Nótese que la presencia del comando 1, '77 exactamente al comienzo del texto está bien definida. Se comenzará el proceso de decisión de la dirección de escritura y el modo inicial tal como se definió. Cualquiera que sea el criterio que finalmente permita decidir la dirección y el modo, tras haber definido ambos se comenzará la lectura de los caracteres, y el comando 1, '77, que será el primero en leerse, se interpretará normalmente y no producirá ningún cambio. Por supuesto, se obtiene el mismo efecto si el comando es ignorado.

Los comandos de reflejo indican si los caracteres que, según la fuente, deben mostrarse reflejados, se reflejan o no. En algún caso puede que haya que aplicar un giro en lugar de un reflejo. Por ejemplo, un paréntesis de apertura en una escritura vertical. Salvo que se

haya indicado lo contrario en el fichero de configuración, en un principio los caracteres que deben reflejarse se reflejan. Si de un carácter la fuente indica que debe reflejarse, pero no dice nada de qué hacer cuando se escribe en una dirección perpendicular a su dirección propia, y si esto no supone una indicación implícita de que no debe hacerse nada, en esos casos el carácter se girará 90° en el mismo sentido en que la dirección de escritura se encuentra girada respecto a la dirección propia del carácter. Nótese que es necesario saber la dirección para la que está diseñado un carácter para poder reflejarlo o girarlo. Así, si una fuente indica que un carácter debe ser reflejado pero no se sabe cuál es la dirección de escritura en la que se escribe sin reflejar, ni se reflejará ni se girará.

Los comandos de reflejo fuerzan uno u otro comportamiento :

1, '70 : No reflejar ni girar ningún carácter

1, '67 : Reflejar o girar los caracteres convenientes

1, '66 : Reflejar todos los caracteres

El comando 1, '66 significa, obviamente, reflejar según el sentido de escritura. Cuando se refleja o gira un carácter lo hace su caja y las líneas de alineación. Éstas y los puntos de alineación se renombran en la manera obvia. Por ejemplo, en caso de una reflexión sobre una línea vertical la imagen del punto *id* pasa a ser el *ii* (el origen). El eje X siempre se dirigirá hacia la derecha, el eje Y hacia arriba y si las coordenadas originales del punto *sd* son (x, y) las del nuevo *sd* serán (x, y) si el carácter se ha girado o (y, x) si se ha reflejado.

Si la fuente no permite deducir la necesidad de reflejar o girar un carácter entonces no se reflejará ni girará ninguno, y no hay diferencia entre el comportamiento 1, '70 y el 1, '67.

Una vez que está decidida la orientación de cada carácter del texto se calcula la posición de cada caja. Todas las cajas comparten alineación (inferior, superior, izquierda o derecha), por lo que lo único que hay que decidir es la posición de la caja en una dimensión. Para ello se tendrá en cuenta las indicaciones de kerning especificadas por la fuente para cada pareja de caracteres, atendiendo a las reflexiones. Los números que aparezcan en el texto que no se correspondan en la fuente con caracteres (comandos, saltos de línea, tabulaciones ...) serán ignorados. Los distintos espacios se tratan como caracteres. Si la fuente no define para un espacio una caja sino solamente su longitud, se tomará para él una caja cuadrada de lado la medida del espacio. Esta operación se lleva a cabo en la fuente real, no en la virtual, de manera que en la fuente virtual se tratará la caja así definida como si ya estuviese en la fuente real. El concepto de la caja cuadrada asegura que el espacio siempre actúa igual (para una fuente real) se trate de modo vertical u horizontal, se gire o no se gire.

Los distintos tramos del texto en un sentido y en otro no están encajados según una lógica sino que se suceden unos a continuación de otros. Cada tramo se imprime según su dirección y unos tramos se colocan a continuación de otros siguiendo el sentido original. Por ejemplo, en un texto que comienza escribiéndose de izquierda a derecha cada tramo

aparecerá a la derecha del anterior.

A la hora de aplicar el kerning hay que tener en cuenta los pares de caracteres adyacentes en la posición final.

Más definiciones

Una vez situado cada carácter a lo largo de la alineación como se explicó en la sección anterior se definen los rectángulos contenedor y de alineación y algún otro concepto.

El rectángulo contenedor es el mínimo rectángulo de lados horizontales y verticales que contiene a todas las cajas, salvo que haya cajas de dimensiones negativas, en cuyo caso puede no coincidir. La definición rigurosa se muestra más abajo para que los casos extremos no compliquen la explicación de los casos normales.

El rectángulo de alineación se obtiene desplazando en paralelo dos lados adyacentes del rectángulo contenedor de manera que pasen a coincidir con sendas alineaciones de la primera caja. Los lados que se desplazan son los siguientes: Por una parte el correspondiente a la alineación del texto, que se lleva a dicha alineación. Por ejemplo, si el texto se escribió con alineación inferior, el lado inferior del rectángulo contenedor se sustituye por la línea de alineación del texto. Por otra el opuesto al sentido de escritura original se hace coincidir con la alineación de la primera caja opuesta a dicho sentido de escritura. Por ejemplo, si la escritura es hacia la derecha se reemplaza el lado izquierdo del rectángulo contenedor por la alineación izquierda de la primera caja.

Para situar el texto puede emplearse bien el rectángulo contenedor bien el de alineación. El que se vaya a emplear será llamado rectángulo de referencia, y la esquina “desplazable”, es decir, la opuesta a aquélla en la que ambos rectángulos son coincidentes por definición, se llamará punto de referencia del rectángulo. De los dos lados que convergen en el punto de referencia, el que corresponde a la alineación del texto se llama línea base, y se le da el mismo sentido que el sentido original del texto; el perpendicular se llama línea de apoyo y se le asigna el sentido opuesto a la alineación del texto, es decir, “hacia arriba” en caso de alineación inferior, etc.

Es posible que en el rectángulo de alineación uno de los lados desplazados respecto al rectángulo contenedor (o los dos) pase al otro lado del lado paralelo que se mantiene fijo. En ese caso se considera para los cálculos siguientes que la dimensión correspondiente es negativa y las calificaciones de inferior/superior o izquierdo/derecho se mantienen como si eso no hubiese sucedido. Esto se da, por ejemplo, si el sentido principal de escritura es de izquierda a derecha y la línea de apoyo del rectángulo de alineación (el borde que se corresponde con el lado izquierdo del rectángulo contenedor) queda más a la derecha que el lado derecho, común a ambos rectángulos. En este caso la anchura sería negativa y *lado derecho* hace referencia al mismo lado tanto en rectángulo contenedor como en el de alineación, aunque en este último se encuentre más a la izquierda que el lado izquierdo.

Como se seguirá de la definición rigurosa, también el rectángulo contenedor puede tener una o ambas dimensiones negativas.

La definición rigurosa del rectángulo contenedor es como sigue: Los lados perpendiculares al sentido de escritura coincidirán, si todas las cajas tienen dimensiones positivas, con sendos lados de la primera y última cajas. Pues bien, dicha coincidencia se toma por definición, de modo que, por ejemplo, en una escritura de izquierda a derecha el lado izquierdo del rectángulo contenedor se toma coincidente con el lado izquierdo de la primera caja (que no tiene por qué ser la caja de más a la izquierda si hay cajas de anchura negativa) y el lado derecho se toma coincidente con el lado derecho de la última caja; es decir, del último carácter de la cadena de texto o bien, si el último tramo sigue el sentido opuesto al sentido principal del texto, del primer carácter leído para ese tramo. Los lados paralelos al sentido de escritura se definen como coincidente cada uno con un lado análogo de una caja, y de entre todas las cajas el que se encuentre más “hacia esa dirección”; por ejemplo, el lado superior en una escritura horizontal se toma coincidente con el más alto de los lados superiores de las cajas. Una vez más, en teoría es posible que el lado superior esté por debajo del lado inferior.

Aunque no es necesario para lo que sigue se pueden definir las cuatro alineaciones del rectángulo contenedor, de la misma manera que como se define el rectángulo contenedor sin más que cambiar en la definición la palabra *lado* por *alineación* allí donde aparece.

Definición completa del elemento de clase 8 (texto plano)

Se describe ya la estructura completa del la clase texto plano. La parte ya descrita anteriormente se muestra de manera resumida. El número de los 4-bytes que siguen al texto se indica de la forma +0, +1... , dando a entender que se cuentan a partir del final del texto. Las coordenadas x e y a que se haga referencia para los 4-bytes +0, +1... estarán todas en el sistema de coordenadas del plano, con origen en $R \equiv p_1$, y definido por los puntos p_1, p_2, p_3 como se explicó.

4-bytes 17--34: $(X, Y, Z)_1, (dX, dY, dZ)_2$ y $(dX, dY, dZ)_3$, que definen el plano y su sistema de coordenadas.

4-byte 35:

b0: Modo de posición

b1: Flags que indican el modo de situar los caracteres

bit 0: Se emplea el rectángulo contenedor (0) o el rectángulo de alineación (1).

bit 1: Se situán los caracteres y luego se desplazan (0) o lo contrario (1).

b23: Libres

4-byte 36:

b01: Longitud en 4-bytes del texto, incluyendo los bytes 0 del final = m

b2: Libre

b3: Codificación

4-byte 37 -- 37+m-1: El texto

Resto del elemento según el *byte 35- b0*:

35-b0 = 0: Punto, incremento y justificación

+0 - +1: x, y del punto de alineación, A

+2 - +3: a_x, a_y que definen la alineación. Sean w, h el ancho y el alto del rectángulo de referencia del texto, que según el bit 0 del *byte 20-b1* será ora el rectángulo contenedor, ora el de alineación. La esquina inferior izquierda del rectángulo de referencia tiene por coordenadas $(x - a_x * w, y - a_y * h)$, en donde (x, y) son las coordenadas de A.

35-b0 = 1: Línea poligonal y distancia

+0: distancia, d

+1:

b01: Número de vértices de la línea poligonal, = n

b23: Libres

+2 -- +4n+1: x, y de los sucesivos vértices (double)

La distancia se aplica en perpendicular a cada punto de la línea definida, de manera que si se hace coincidir la línea base del rectángulo de referencia con un segmento de la línea, en dirección y sentido, el sentido de la línea de apoyo marca el desplazamiento para valores de d positivos.

Si el bit 1 del *byte 20-b1* está a 0, se calcula la posición de los caracteres sobre la línea y posteriormente se desplazan éstos en perpendicular. Si por el contrario está a 1, primero se calcula la línea desplazada y a continuación se sitúan sobre ella los caracteres. En este último caso la línea desplazada se define como el lugar geométrico de los puntos a distancia d de la línea indicada, limitada por las perpendiculares a la misma en su primer y último puntos, situada del lado que se sigue de lo explicado en el párrafo anterior.

En cualquiera de los dos casos se compara la longitud w del rectángulo de referencia con la longitud l de la línea sobre la cual se van a situar los caracteres. Se define el exceso e como $e = l - w$. e puede ser positivo o negativo. Si e es igual a cero no hay que hacer nada. En caso contrario el espaciado entre palabras y letras debe aumentarse o reducirse para cubrir exactamente la línea. La razón r de aumento o disminución del espacio entre palabras y el espacio entre caracteres se toma de la fuente y si esta información no está disponible se aplicará la razón 4:1.

Una fuente puede incluir espacios de distinta longitud pero tendrá en cualquier caso un espacio de referencia, la separación normal entre palabras. A cada uno de ellos se aplica una razón que está en relación lineal con su longitud, de manera que es 1 para una longitud igual a 0 y r para una longitud igual al espacio de referencia.

Dado que en este punto los espacios también son cajas, esto puede entenderse como que el espacio a insertar entre dos cajas consecutivas será $0,5(r' + r'')f$, en donde r' y r'' son las razones que corresponden a una y otra caja, y junto a las cajas de los extremos no se inserta nada si no son cajas de espacio, y si lo son se inserta $0,5rf$.

Sea kf la suma de todos los espacios a insertar, en donde k es conocido. Se plantea $kf = e$. Si k es cero la posición de cada caja no se modifica, y por tanto en este caso el texto no cubrirá exactamente la línea. Si es distinto de cero se obtiene $f = e/k$ y se calcula la nueva posición de cada caja; es decir, su distancia al punto de referencia a lo largo de la línea base.

Ahora se sitúan los caracteres a lo largo de la línea sobre la que hay que colocarlos situando cada carácter a la distancia que le corresponde del inicio de la línea, y la base de la caja del carácter se orienta según el segmento o arco a donde haya ido a parar el carácter. En concreto, para definir la orientación de la caja se toma el punto de la línea que corresponde al centro del carácter. Si fuese exactamente un vértice se aplicará la del segmento o arco más avanzado. Si algunas cajas (centros de caja) están situadas respecto al punto de referencia a una distancia mayor que l o menor que cero se prolongarán los segmentos último y primero respectivamente de la línea indicada cuanto fuese necesario.

35-b0 = 2: Punto, radio y ángulos inicial y final

+0: radio, r

+1: Ángulo inicial, en grados sexagesimales (float)

+2: Arco abarcado, en grados sexagesimales

Estos tres valores definen un arco de círculo. El centro es el punto R del plano, los ángulos se cuentan en sentido horario y para el ángulo inicial el origen es el punto superior (de mayor y) del círculo.

El texto se sitúa igual que en el modo anterior, con el arco como línea y $d = 0$. Por lo tanto, para cualquier tipo de escritura el texto avanzará en sentido horario si el arco abarcado es positivo y en sentido antihorario si es negativo. Por ejemplo, si el arco abarcado es positivo el texto se sitúa en la parte exterior del arco para una escritura de izquierda a derecha con alineación inferior así como para una escritura de arriba a abajo con alineación izquierda, en el primer caso con la parte de arriba de los caracteres apuntando hacia afuera del arco y en el segundo hacia el sentido contrario de avance del arco, mientras que si es negativo se sitúa en la parte interior y en ambos casos con los caracteres apuntando en sentido opuesto al caso anterior; exactamente lo contrario sucede para escrituras de derecha a izquierda a derecha con alineación inferior y escrituras de abajo a arriba con alineación izquierda, respectivamente.

Tamaño del texto

El tamaño del texto, que se indica en el fichero de configuración, se muestra en las unidades del plano, sin más que sustituir la palabra *punto* por la perífrasis *unidad del plano*.

Representación de los nombres de elementos

El fichero de configuración no puede incluir indicaciones de dirección para los nombres. Sí que define una fuente en función del tipo, subtipo, etc., y a la hora de mostrar el texto se ha de seleccionar cada carácter adecuado de la fuente teniendo en cuenta las codificaciones del nombre en el fichero de gráfico y de la fuente. Si el fichero de configuración no define la fuente a emplear el visor escogerá la que mejor le parezca, pudiendo depender la elección de la codificación del nombre o los caracteres en él empleados.

La determinación de la dirección a emplear sigue el mismo principio que para los elementos texto pero es más sencilla porque algunos puntos no se aplican por imposibles o no permitidos. Así, los puntos 1 y 2 no son aplicables y se comienza directamente en el 3. La aplicación del punto 4 es más simple y queda como sigue :

4. La dirección definida por el primer carácter que implique una dirección. Si el nombre consta solamente de caracteres neutros pero la codificación tiene una dirección principal (por ejemplo, una codificación para un idioma en el que los textos se escriben de derecha a izquierda, con independencia de que los números se escriban en sentido contrario y de que la fuente contenga las letras latinas en las posiciones indicadas por la codificación ASCII) o bien si no la tiene (p.e. Unicode) pero es la codificación de la fuente la que tiene una dirección principal, será esa dirección.

Todo el nombre se representa siguiendo una única dirección. Se debe aplicar el kerning entre caracteres y se recomienda reemplazar las secuencias de caracteres por sus ligaduras correspondientes (siempre que no se haya indicado lo contrario para esa fuente en el fichero de configuración).

El nombre se representa en unidades píxel, de manera que los parámetros tamaño y grosor así como los que definen la posición del texto, todos ellos definidos en el fichero de configuración, se interpretan en píxeles.

El nombre se representa en el modo *punto, incremento y justificación*. El punto de referencia es el centro o punto origen del elemento, que está definido para todos las clases de elemento excepto 1,4,7,8 y 9. Para estos elementos, el visor tomará como punto de referencia el que quiera. Los parámetros : x , y , a_x y a_y , así como la elección entre rectángulo contenedor/de alineación, no se indican para cada elemento sino que se especifican en el fichero de configuración. Si no se especifican el visor empleará los valores que quiera.

Información adicional

La información adicional deberá estar formada por bloques estructurados del siguiente modo :

4-byte 0:

b0: Código que indica el tipo de información almacenada en el bloque.

b1: El bit alto indica, si está a 1, que el bloque ha de ser ignorado.

El segundo bit más alto indica que el bloque no almacena toda la información, sino que incluye un enlace a información fuera del fichero de gráfico (así, por ejemplo, una aplicación puede saltar rápidamente este bloque si sabe que la información enlazada no va a estar disponible).

b23: 4-bytes que ocupa el bloque (por lo tanto empieza a contar en el 4-byte 0)

4-bytes 1 -- n-1: La información

Los códigos 250--255 están reservados para su definición presente o futura. Los demás nunca serán definidos.

Código 255

Conjuntos de elementos a los que pertenece el elemento. Cada elemento puede tener solamente un bloque de inf. adic. con código =255.

4-byte 1: b01; b23: n° de agrupación; conjunto al que pertenece dentro de esa agrupación.

4-byte 2: b01; b23: n° de agrupación; conjunto al que pertenece dentro de esa agrupación.

etc.

Cada agrupación es, como su nombre indica, una agrupación de los elementos en conjuntos, de manera que los conjuntos de una agrupación son independientes de los de otra.

En caso de que unas agrupaciones contengan superconjuntos de otras se recomienda que los números de las agrupaciones vayan ascendiendo a medida que los conjuntos son más grandes.

Además pueden existir conjuntos implícitos, formados por elementos cuya referencia al nombre sea igual (y distinta de FFFF FFFF). Esta interpretación depende de cada visor. Un visor puede ignorarlo, o incorporarlo como un criterio de búsqueda y selección de elementos, o tratarlo de cualquier manera que se considere interesante.

La representación de cada elemento se define en el fichero de configuración en función de los tipos, subtipos y conjuntos. Por ello, si se quiere que un elemento tenga una representación única, distinta de las de otros similares, es necesario incluirlo en un conjunto a él solo. Se recomienda emplear para ello las agrupaciones 0 y 255. Véase extensiones futuras al respecto.

Código 254

Descripción, comentario o propiedad.

4-byte 1: Tipo de información almacenada.

FFFF FFFF : Descripción del elemento

FFFF FFF0 -- FFFF FFFD : Comentario

Ninguno de los anteriores : Referencia al nombre de la propiedad

Se empleará un número par cuando se trate de una descripción y un número impar si se trata de un comentario.

4-byte 2: Referencia al texto que contiene la descripción, comentario o valor de la propiedad.

Extensiones futuras

Clase 20 : célula

Clase 1020 : definición de célula

Las células se identificarán por el nombre. Cada elemento de una célula tendrá uno o varios nombres. Las cualidades que sean NombreQual+Nombre_ElemenCel se aplican a ese elemento, las otras no (véase la explicación del fichero de configuración). Si alguno de sus nombres es «ausencia de nombre» (FFFF FFFF) se le aplicará la cualidad NombreQual. Por ejemplo :

ColorBorde 339A56

ColorFrame 0459FF

Color 786666

ColorBorde se aplicará a todos los elementos de la célula entre cuyos nombres se encuentre «Borde». Los elementos de la célula que no tengan nombre se colorearán con el color indicado por *Color*, esto es, 786666.

Se empleará uno de los códigos reservados de la información adicional para permitir una simbología particular para cada elemento.

FICHERO DE CONFIGURACIÓN

Formato

El carácter % en este fichero indica el comienzo de un comentario y todo lo que aparezca a partir de ese carácter y hasta el final de la línea es ignorado, salvo situaciones excepcionales en las que se indicará claramente.

A parte de los comentarios este fichero está estructurado en unidades delimitadas por `\begin` y `\end`, o por comandos de una única línea, siempre comenzando por `\` (de momento solamente uno: `\input`). Los bloques pueden contener otros bloques.

Bloque Info

El primer bloque, antes incluso que cualquier comando `\input`, ha de ser el siguiente:

```
\begin Info
Version 00
%Otra información
\end
```

El formato de este fichero todavía es inestable y podrá cambiar sin que se modifique el número de versión. El bloque Info consta de varios pares nombre/valor, entre los cuales ha de aparecer obligatoriamente *Version*. Se recomienda que sea el primero de los pares. De momento están definidos además los elementos *Formas*, *Fondo* y *Resalte*:

```
Formas "misformas.fdf"
Fondo DDDDDD
Resalte FFFFFF
```

Fondo es el color de fondo, en hexadecimal, como se explica más abajo, y *Resalte* el color de los elementos reslatados, por ejemplo los seleccionados.

Formas indica el fichero de definición de formas para los elementos puntuales. El nombre del fichero se extiende desde el segundo carácter distinto de espacio (en este caso la 'm') hasta el que preceda a la última aparición en la línea del primer carácter, en este caso una comilla "; es decir, hasta la 'f' que precede a la segunda y última ". Por ejemplo

```
Formas .misformas.ldf.
Formas .misformas.ldf.xxxx
```

ambas líneas indican el fichero misformas.ldf

El nombre puede por tanto contener el carácter %, que en ese caso no se interpreta como comentario. El carácter delimitador no puede ser % (pues sería interpretado como comentario).

De esta manera el nombre del fichero puede contener cualquier combinación de caracteres excepto saltos de línea. Si el fichero no es una ruta absoluta se buscará primero tomando como directorio base el directorio en el que se encuentra el fichero de gráfico, a continuación tomando como base uno o más directorios fijos que cada visor/editor definirá (por ejemplo, C:\programas\Visor\configuraciones\formas\); el visor/editor puede no definir ningún directorio para este paso. Por último se buscará el fichero con el nombre indicado, ignorando la ruta, en uno o más directorios fijos que cada visor/editor ha de definir obligatoriamente. Por ejemplo,

Formas.\puntos\parcela.fdf

haría que se buscase el fichero parcela.fdf en los siguientes directorios, por ejemplo :

- 1º. .\puntos\ en relación al fichero de gráfico
- 2º. 1. C:\usuarios\Javier\programas\Visor\formas\puntos\
2. C:\programas\Visor\configuraciones\formas\puntos\
3º. C:\programas\Visor\configuraciones\formas\

Si el fichero no se encuentra o si no se indica un fichero deberá emplearse el fichero por defecto formas.fdf.

Comando \Input

El comando **\input** va seguido de un nombre de fichero exactamente igual que el nombre *Formas* dentro del bloque **Info**. Si el fichero no es una ruta absoluta se buscará igual que en el caso anterior según los criterios 1º y 2º, en orden, pero no el tercero. Los directorios empleados como base en este caso pueden ser distintos de los empleados para las formas.

Si el fichero se encuentra y se puede leer, se lee el fichero y cuando se termine se continúa con el fichero actual. Los ficheros abiertos por **\input** pueden contener a su vez este comando, sin límite o hasta un límite que imponga la aplicación. Deben permitirse al menos 8 ficheros abiertos a la vez, contando con el fichero inicial.

El comando **\input** es útil para que las partes más estables de un fichero --definición de tipos y subtipos, agrupaciones, etc.-- se lea de un fichero fijo.

Un fichero abierto por **\input** no debe obligatoriamente contener el bloque **Info**, y si lo contiene éste no tiene por qué especificar el número de versión. En ese caso el fichero se lee de acuerdo al número de versión del fichero desde el que se abrió. Los otros elementos

del bloque **Info** serán ignorados en caso de que ya hayan sido leídos en algún otro fichero (es decir, tiene preferencia la primera aparición).

Fuentes

Un bloque **Fuente** define una fuente virtual, basada en una fuente real o en otra virtual :

```
\begin Fuente
Nombre CMR vertical
Font .cmr10.
\begin Transformacion
Trans O 1.2 0 0 0.8
\end
Direccion ga
RazonEspacio 10:1
\end
```

Nombre es el nombre que se la dará a la fuente. Consta de todos los caracteres que siguen hasta el final de línea o carácter % de comentario. *Font* es la fuente de la que deriva. Puede ser una fuente real o una virtual que se encuentre ya definida. Se escribe de la misma forma que los nombres de fichero, como se explica arriba. Los demás campos son opcionales :

Dirección La dirección en la que se ha de mostrar el texto cuando se emplee esta fuente (salvo indicación contraria en algún sitio de mayor preferencia). Se indica mediante dos caracteres consecutivos en minúscula. El primero indica la alineación a emplear : s : arriba, i : abajo, g : izquierda, d : derecha. El segundo indica el sentido de escritura. Una 'a' indica escritura hacia la derecha si la alineación es arriba o abajo y hacia abajo si es izquierda o derecha, y una 'b' el sentido contrario.

Ligaduras Puede ir seguida de «Si» o bien de «No». Indica si se aplican o no para los nombres de elemento (en los elementos texto no tiene sentido).

Transformación La transformación a aplicar a cada carácter, junto con su caja y alineaciones. El origen de coordenadas para la transformación se toma en el punto de alineación ii.

Trans La transformación del carácter propiamente. La letra que viene a continuación es el tipo de transformación, que va seguida de los parámetros separados por espacio. Una 'O' significa ortogonal, y va seguida de cuatro parámetros a, b, c, d , de manera que si las coordenadas originales de un punto son (x, y) las transformadas son $(ax + by, cx + dy)$. Además tienen que ser cero b y d o bien c y d , y los otros no pueden ser cero. Los puntos de alineación y las alineaciones se renombran de manera que sigan

manteniendo la misma posición relativa que en la caja original. El eje X siempre apuntará hacia la derecha y el eje Y hacia arriba.

No hay más elementos dentro de *Transformación*. Serían necesarios para transformaciones más complicadas, pero seguramente nunca se definan. Si se quiere llevar a cabo modificaciones más complicadas será necesario aplicarlas a la fuente real mediante otra herramienta.

La caja del espacio se transforma como cualquier otra. El kerning debe también ser transformado, y será necesario tener en cuenta los cambios de orientación de los caracteres. Por ejemplo, si se aplica la transformación «O -1.2 0 0 1», a parte de multiplicar por 1,2 todos los valores de kerning horizontales, el que antes existía entre la k y la o para una escritura de izquierda a derecha es ahora el kerning entre la o y la k para ese mismo sentido de escritura.

La transformación no cambia el sentido de escritura asociado a cada carácter.

RazonEspacio La razón entre los espacios a insertar (ya sean positivos o negativos) entre palabras y entre caracteres cuando haya que separar o juntar los caracteres para ajustar la longitud del texto a un valor determinado. Puede indicarse mediante un número decimal normal, como 3 o 2.5, o mediante una razón de dos enteros como en el ejemplo. Esta última es la única manera de indicar que las letras de una palabra no se separen ni se junten, mediante N:0, para cualquier N (p.e., 1:0).

Puede existir un bloque **Fuente** especial, cuyos parámetros se aplican a todas las fuentes salvo en lo que indique cada una en su bloque **Fuente**.

```
\begin Fuente General
\begin Transformacion
Trans O 1.2 0 0 0.8
\end
Direccion ga
RazonEspacio 10:1
\end
```

Contiene los mismos datos que un bloque **Fuente** normal salvo por los campos *Nombre* y *Font* que no pueden aparecer.

Tipos y subtipos

Esta es normalmente la información que forma la mayor parte del fichero. Para los tipos, su representación y significado viene indicado como sigue, en donde se ejemplifica con el tipo 32. La explicación se desarrolla a continuación.

\begin Tipo 32

Significado Vértices de la red Regente

\begin Representacion %Opcional

Forma 0424

Color B060D8

\end Representacion %Opcional

\begin Texto

Color A0A0A0

Alineacion 1 3 4 0 0

\end Texto

\begin Subtipo criterio 1 1 2

Significado Empleado

\begin Representacion %Opcional

Color 8800DD

\end Representacion %Opcional

\end Subtipo

\end Tipo

Las especificaciones que pueden aparecer en **Representacion** dependen del tipo geométrico de elemento: el valor de los bits bajos del *byte 6*. Los que no sean de aplicación serán ignorados. De momento Gráfico tiene definidos los siguientes para todos los elementos a excepción de las clases 8 y 9 (elementos de texto):

Color En hexadecimal, en formato RRGGBB (RojoVerdeAzul)

Forma Para los puntos. Un número de forma, expresado en base 10, 8 o 16 con la notación del lenguaje C. Las formas se encuentran definidas en el fichero de formas. La forma 0200 es vacía (no se representa el punto). La forma por defecto para todos los elementos excepto puntos y centros de polígono es 0200 (no representar el punto). Para puntos y polígonos (byte6 bajo igual a 0 o 2) cada visor tendrá sendas formas por defecto.

Escala aplicada para representarlo, para elementos (*byte 6* bajo) entre 2 y 10. En el elemento 2 (polígono) se aplica respecto al punto indicado como centro. En los demás elementos su aplicación es obvia.

Grosor En píxeles. Valor con decimales, p. e., 0.8, 2.25, etc.

GrosorT Grosor en unidades terreno (por lo tanto aumenta al aumentar el zoom).

Estilo Estilo de línea. Todavía no están definidos.

FormaP, *ColorP* Para el centro de los polígonos.

Color2, Color3, Grosor2, GrosorT2... Para elementos que puedan necesitar más colores y grosores colores. En los elipsoides, si se representan como una elipse más un segmento, los parámetros *Color, Estilol, Grosor y GrosorT* son los de la elipse, mientras que *Color2*, etc. son los del segmento. Si no aparecen estos últimos se tomarán iguales a los de la elipse.

Las clases 8 y 9 pueden incluir los siguientes :

Grosorb Grosor del borde. Por defecto es 0, salvo que la fuente defina lo contrario.

Colorb Color del borde.

Colorf Color de relleno del carácter. Si el grosor del borde es 0, es el color en el que se mostrarán los caracteres.

Modo Puede ser 00, 01, 10 o 11. El primer valor indica si se dibuja el borde y el segundo si se pinta el relleno. Por tanto, si se indica 00 o bien 01 con grosor de borde 0, no se mostrará nada.

Color Equivale a *Colorb* más *Colorf*.

Las líneas «**\begin Representacion**» y el «**\end**» correspondiente son opcionales.

Dentro del bloque **Texto** pueden aparecer los siguientes campos :

Font Por defecto en el visor de Aerotri es Times new Roman, pero cada visor puede definir una fuente por defecto. Va desde el primer caracter distinto de espacio tras la palabra *Font* hasta el final de línea o un carácter %. Por tanto, *no* se escribe como un nombre de fichero.

Size El valor por defecto es 14.

Color Por defecto se emplea el mismo que para el elemento.

Alineación Va seguida de una letra que indica la línea de referencia y cuatro valores que son los parámetros x , y , a_x y a_y para su posicionamiento respecto al punto de referencia del elemento. Véase su descripción en el modo *Punto, incremento y justificación* de los elementos texto (clase 8). La letra indica si se emplea el rectángulo contenedor o el de alineación. Se indica mediante las letras c y l respectivamente.

Especificación «from»

Tras cada cualidad (*Color, Forma*, etc.) puede aparecer la palabra «from», en cuyo caso el valor de esa cualidad se toma de otro tipo/subtipo según las siguientes opciones :

from subtipo m	Igual que la especificada para el subtipo m de este mismo tipo
from tipo n	Igual que la especificada para el tipo n

from tipo n subtipo m Igual que la especificada para el tipo n , subtipo m

Si tras la especificación de tipo/subtipo se encuentra la palabra «texto», la cualidad de toma del texto del tipo/subtipo en cuestión :

Color from tipo 3 texto

Si lo que aparece es el nombre de una cualidad se tomará de esa cualidad en lugar de la del mismo nombre, siempre que tenga sentido. Pueden combinarse ambas indicaciones :

Color from tipo 2 Color2

Color3 from tipo 2 texto Color

Aunque en este último caso, si se omitiese la palabra «Color», un programa visor de gráficos podría deducir que dado que el texto no tiene definido *Color3* el valor se ha de tomar de *Color*.

«from .» significa tomar la cualidad igual que la del elemento. Se usa normalmente para el texto o para propiedades secundarias :

Grosor2 from . Grosor

\begin Texto

Color from .

\end

El tipo indicado tras from puede ser el propio tipo en el que estamos, si el resultado no es una referencia a la cualidad que se está definiendo. Es válido por ejemplo dentro de un bloque \begin Subtipo ... \end, o \begin Texto ... \end, o indicando otra cualidad al final («Color2 from tipo 13 Color3», por ejemplo). Fuera de bloques \begin Subtipo ... \end indicar el tipo corriente es lo mismo que «from .».

La especificación from puede ir precedida de la “cualidad” *All*, con el significado obvio. Si *All* aparece fuera de \begin Representacion ... \end también copia *Significado*, la representación del **Texto** y, en un tipo, los subtipos. Pero si ese tipo incluye subtipos estos no se ven afectados por la indicación *All*. Por ejemplo

\begin Tipo 21

All from tipo 20

\begin Subtipo 1

Color FF0000

\end

\end Tipo

Si el tipo 20 incluye un subtipo 1, no se copiará nada de éste al subtipo 1 del tipo 21.

Si aparece una cualidad dos veces, una con la especificación from y otra con un valor

concreto, la especificación from tiene preferencia. Si embargo la indicación «All from» es la que menos preferencia tiene de todas, de modo que cualquiera otra la reemplaza. Ejemplo :

```
\begin Tipo 35
Significado Límite de parcela
All from tipo 34
\end Tipo
```

Con lo cual el tipo 35 toma todo del 34 excepto *Significado*, que es reemplazado por «Límite de parcela». Los *Significados* de los subtipos del tipo 34 se añaden a «Límite de parcela» para el tipo 35.

De cara a la indicación «All from», *Significado* y *SignificadoR* (que se explica más abajo) funcionan como una unidad, de manera que si existe cualquiera de ellos en el tipo que se está definiendo o en uno de sus subtipos, no se copiara ninguna de las dos del tipo al que se hace referencia.

All from . %no es válido, ni siquiera dentro de Texto.

Dentro de **Texto**, la indicación from que sigue a *All* se sobreentiende que se refiere al texto, y fuera de él no está permitida una referencia *All* a un texto :

```
\begin Texto
All from tipo 2 %es lo mismo que : All from tipo 2 texto
\end
All from tipo 3 texto %no es válido
\begin Representacion
All from tipo 3 texto %no es válido
\end
```

En un tipo una cláusula «All from» que apunte a un subtipo, o viceversa, es válida pero se recomienda no emplearla. Copia las cualidades y el bloque **Texto**, pero no el *Significado*. Pero en un tipo no puede apuntar a un subtipo de ese mismo tipo y viceversa. Una referencia from copia las cualidades literalmente, lo que significa que si el tipo/subtipo al que se apunta tiene una cualidad para la cual se indica «from .», «from .» es lo que debe entenderse para esa cualidad, y no el resultado de resolver eso para el tipo/subtipo al cual se apunta.

Subtipos

La información específica para subtipos puede darse individualmente o por grupos. Si lo que viene a continuación de la palabra **Subtipo** es un número o una lista de números separados por comas se entiende que la información se refiere a esos subtipos.

`\begin Subtipo 3`

`\begin Subtipo 3, 4, 9`

De lo contrario será una palabra clave y se está especificando un conjunto de subtipos. Hay varias palabras claves con distintos significados.

`min m max M` Entre un valor mínimo y uno máximo. Si «min» no aparece el mínimo es 0. Si «max» no aparece el máximo es 255 (el máximo subtipo posible). Por ejemplo,

`\begin Subtipo min 100 max 200 %Intervalo [100,200]`

`\begin Subtipo max 8 %internvalo [,8] = [0,8]`

`\begin Subtipo min 5 %intervalo [5,) = [5,255]`

`mask` Que tenga algún bit de la máscara a 1.

`\begin Subtipo mask 0x0F %Alguno de los últimos cuatro bits a 1`

`Mask` Que tenga todos los bits de la máscara a 1.

`\begin Subtipo Mask 0x0F %Los cuatro últimos bits a 1`

Por último, si tras la palabra subtipo sigue un paréntesis de apertura entonces se trata de una expresión lógica que se evalúa. El subtipo se representa mediante la letra `s`. Si el resultado de la expresión es un 1 lógico (true) o un número distinto de cero el elemento con subtipo `s` satisface el criterio. En caso de que sea 0 lógico, 0 o la expresión no esté bien formada se considera que el elemento no satisface el criterio. Por ejemplo

`Subtipo ((s&07==07) || (s&070==070))`

Una vez más, los números se pueden escribir base 10, 8 o 16.

La sintaxis es la del lenguaje C. Se pueden emplear los operadores de bit `&`, `|`, y `~`, así como `<<` y `>>`; los operadores numéricos de relación `==`, `!=`, `>`, `<`, `>=` y `<=`, y los operadores booleanos `&&` y `||`. El resultado de los operadores de relación y booleanos no podrá ser un operando de un operador que no sea `&&` o `||`, pero el recíproco no es cierto. Es decir, se puede escribir

`(s&0xF && s&0x70)`

sin necesidad de escribir los `!=0`.

El orden de prioridad de los operadores es

`~`

`<< >>`

`& |`

== != > < >= <=
&& | |

Los operadores mostrados en una misma línea tienen la misma prioridad y las expresiones se evalúan de izquierda a derecha.

La información de *Significado* para los subtipos se añade a la del tipo, y la de **Representación** y **Texto** la sustituye en los campos que se especifiquen. Si se quiere que el significado no se aplique también al subtipo se empleará la palabra clave *SignificadoR* en lugar de *Significado*. Pueden emplearse ambas; solamente se heredará la de *Significado*, mientras que el significado para el tipo se formará con *Significado* + *SignificadoR*.

Los subtipos pueden anidarse, heredándose el significado igual que de un tipo al subtipo. Ejemplo :

```
\begin Tipo 1
\begin Subtipo mask 07
    Significado Punto de Apoyo
    SignificadoR incompleto
    Forma 740
    Color 88FF88
    \begin Subtipo Mask 07
        Significado completo
        Color 0000FF
    \end
\end
\begin Subtipo min 010 max 0100
    Significado Punto de Control
    SignificadoR incompleto
    Forma 116
    Color 88FF88
    \begin Subtipo 070
        Significado completo
        Color 7700DD
    \end
\end
\end
```

El visor de gráficos de Aerori no implementa subtipos anidados.

Tipos CCL en un fichero cfg

El tipo FFFFF no puede aparecer en un fichero de configuración. Todos los demás pueden aparecer. En particular pueden existir los tipos FFFFC, FFFFD y FFFFE; es decir, elemen-

tos CCL-12 a CCL-14. Dado que estos elementos no se representan pueden emplearse en un fichero cfg para almacenar valores de cualidades a los que hacer referencia mediante indicaciones from.

Agrupaciones y conjuntos

Se muestra a continuación un ejemplo :

\begin Agrupacion 5

Significado Elementos agrupados por fecha de creación

\begin Conjunto 0

Significado Elementos que estaban en el mapa base

...

\end Conjunto

\begin Conjunto 1

Significado Elementos creados en el mapa original

...

\end Conjunto

\begin Conjunto 2

Significado Elementos incorporados en la primera revisión

...

\end Conjunto

etc.

\end Agrupacion

Ni que decir tiene que las agrupaciones y conjuntos no tienen por qué aparecer en orden ascendente ni existir exactamente en un intervalo 0-N.

Dentro de cada **\begin Conjunto \end Conjunto** pueden aparecer cualidades, p.e. «**Color 80FF80**», que afectan a todos los elementos de ese conjunto, así como bloques **\begin Tipo n ... \end Tipo**, en cuyo caso las indicaciones de representación en él contenidas afectan solamente a los elementos del tipo y en su caso subtipo en cuestión.

La jerarquía entre la representación indicada para cada tipo fuera de los conjuntos y lo indicado en los conjuntos se establece como sigue

\begin Jerarquia

<selección 1>

<selección 2>

etc.

\end Jerarquia

Cada selección es una línea indicando un criterio de selección en función de Agrupación, Conjunto, Tipo y Subtipo. Para cada elemento gráfico, la primera línea que se satisfaga para ese elemento indica de dónde se toman los parámetros de su representación. Si el elemento no satisface ningún criterio de selección se representará de acuerdo a lo indicado para su tipo y subtipo. Por lo tanto si el bloque Jerarquia no existe todos los elementos se representan según su tipo y subtipo sin que la información de los conjuntos afecte para nada.

Cada línea selección comienza bien por *Agrupacion* bien por *Tipo*. En el primer caso, si el elemento satisface el criterio de selección se empleará la simbología del conjunto al que pertenece para la agrupación indicada, en lo que reemplace a la del tipo/subtipo (si no la reemplaza en nada será igual que la del tipo/subtipo), en el segundo caso se empleará la del tipo/subtipo.

Agrupacion a %Para todos los elementos que pertenezcan a algún conjunto de la agrupación *a* se empleará la simbología del Conjunto

Agrupacion a Conjunto c_1 Conjunto c_2 ... Conjunto c_n %Los elementos que pertenecen a alguno de los conjuntos c_1, c_2, \dots, c_n de la agrupación *a*.

Agrupacion a !Conjunto c_1 !Conjunto c_2 ... !Conjunto c_n %Los elementos que pertenecen a alguno de los conjuntos de la agrupación *a* distintos de c_1, c_2, \dots, c_n .

No pueden aparecer mezclados Conjunto's y !Conjunto's. Esta manera de indicar un conjunto de Conjuntos se representará por [Conjunto], incluyendo en ello la indicación vacía, e.d., que no aparezca ningún Conjunto ni ningún !Conjunto. Por { !} se representará un ! opcional pero que tiene que ser igual para cada aparición de { !}. El formato completo de un criterio que comience por *Agrupacion* es

Agrupacion a [Conjunto]

Agrupacion a [Conjunto] { !}Tipo t_1 {Subtipo xxx1} { !}Tipo t_2 {Subtipo xxx2} ...

en donde xxx1, xxx2, etc. indica uno o varios subtipos de cualquiera de las maneras que se explicó más arriba y {Subtipo xxx1} significa que la indicación de Subtipo es opcional. El ! en los Tipos con indicación de Subtipo se ha de interpretar como !(Tipo t Subtipo xxx).

Para cualquier criterio de selección más complicado se ha de emplear el formato

Agrupacion a (<expresión lógica>)

en donde la sintaxis de <expresión lógica> ya se expresó más arriba y en ella el conjunto, tipo, subtipo y clase se representan por c , t , s y b respectivamente (la b es de «byte 6»).

Un criterio de selección que comience por *Tipo* puede ser de dos maneras :

a) Una indicación de tipo única, con o sin subtipo :

Tipo *t* Subtipo xxx

b) Exactamente igual que los de *Agrupación* pero con la palabra *Tipo* delante :

Tipo Agrupacion *a* ...

Visibilidad de elementos

\begin Ver

ON 2 3 10

OFF 21 22 23

TON 1 2 10

TOFF 3

<selección 1>

<selección 2>

etc.

\end Ver

Si el bloque **Ver** no está presente elementos estarán todos visibles y los nombres de elemento todos ocultos. Los elementos del bloque **Ver** modifican este comportamiento por defecto.

Los elementos del bloque **Ver** pueden ser alguna de las palabras clave seguida de unos número de tipo o bien criterios de selección. Las palabras clave son *ON*, *OFF*, *TON* y *TOFF* que indican que los tipos que siguen están visibles o no (*ON* y *OFF*) o que el nombre del tipo está visible o no (*TON* y *TOFF*). Así, la primera línea del ejemplo mostrado indica que los elementos de los tipos 2, 3 y 10 han de verse, mientras que la última indica que el nombre de los elementos de tipo 3 no debe mostrarse.

Cada <selección> sigue el formato de una selección de jerarquía que empiece por *Agrupacion*, pero la indicación de agrupación es opcional. Los elementos que satisfagan el criterio se mostrarán, pero si la línea comienza por ! no se mostrarán. Así pues el formato de una selección dentro de **Ver**, en donde el significado de { ! } se explicó más arriba y el resto de llaves {} indican opcionalidad, es

{!}{Agrupacion *a* [Conjunto]} {{ !}Tipo *t*₁ {Subtipo xxx1} { !}Tipo *t*₂ {Subtipo xxx2} ... }

o bien

{!}{Agrupacion *a*} (<expresión lógica>)

Por ejemplo los siguientes :

```
!Tipo 3 Subtipo mask 0xF
Agrupacion 3
Agrupacion 0 Tipo 8 Tipo 1 Tipo 2
! (b==0 || (t==2 && s&077==0))
```

Estos criterios se aplican al elemento, no a su texto. Si se quiere que se apliquen al texto se escribirá al principio o al final de la línea {texto}; si se quiere que afecte a ambos se escribirá {. texto} o bien {texto .}. Cualquier otra palabra que aparezca entre llaves se ignorará sin constituir un error en el formato (lo que no excluye que las aplicaciones que muestren gráficos avisen de una palabra irreconocida).

Al contrario que en el bloque **Jerarquía**, en donde los parámetros de representación para un elemento los define el primer criterio de selección que satisfaga el elemento, en el bloque **Ver** la visibilidad queda definida por el último criterio de selección que el elemento cumple.

Varias vistas

Los bloques **Ver** y **Jerarquía** pueden indicar, tras la misma palabra, un número que especifique el número de vista a la que se aplican, para los programas que permitan varias vistas. Ej.,

```
\begin Ver 1
...
\end

\begin Ver 2
...
\end

...
```

Si existe un bloque **Ver** sin número sus indicaciones se aplican a todas las vistas; las de las demás vistas acumulándose a ellas. Lo mismo rige para un bloque **Jerarquía** sin número, pero en este caso sus indicaciones se acumulan detrás de las de cada bloque numerado.

El resto de la información específica para cada vista se indica en bloques **Vista**, por ejemplo

```
\begin Vista 2
...
\end
```

La transformación geométrica que define la posición en la vista de cada punto del objeto se dará en un subbloque **Transformación**, siendo su formato específico para cada aplicación. Puede ir desde una matriz de rotación a una proyección cartográfica completa. El resto de la información en un bloque **Vista** es específica para cada aplicación.

Extensiones futuras

Añadir a los criterios de selección variables que representen parámetros del elemento. Para permitir por ejemplo que se vean solamente los elementos con Z mayor que 500, o que los segmentos de tipo=60 (por ejemplo) se representen más gruesos si su longitud es mayor que 0,2, etc.

FICHERO DE FORMAS

Cada forma es un mapa de bits. El fichero de formas está constituido por una sucesión de bloques como el que sigue :

```
\begin formas  
numero 020  
limites -3 3 -2 3  
centro 0 -0.29  
uno #
```

```
<formas>
```

```
\end
```

número indica el número que corresponde a la primera forma tras la cabecera del bloque; las siguientes tendrán los números sucesivos. Cada forma tiene un píxel con coordenadas (0, 0) de cara a la definición de la forma. La línea *limites* indica los valores xmin xmax ymin ymax según ese sistema de coordenadas. Ningún píxel puede salirse del recuadro definido por esos valores. Puede existir opcionalmente otra línea igual que la de límites pero encabezada por la palabra *box*. Los valores xmin, etc. indicados para *box* definen otro rectángulo que puede ser distinto del de límites. El visor de gráficos de Aerotri lo ignora.

centro indica las coordenadas del centro de la figura respecto al centro del píxel 0,0. Es el punto de la forma que ha de situarse exactamente en las coordenadas del punto que la forma está representando. Se emplea para dibujar las formas con precisión subpíxel. Sus coordenadas deben estar comprendidas en el intervalo $[-0,5, 0,5]$ para que los programas que no empleen esta información situen las formas en posición correcta.

Tras la cabecera (la línea *uno* se explica más adelante) vienen las formas, cada una de las cuales debe constar exactamente de $y_{\max} - y_{\min} + 1$ líneas con $x_{\max} - x_{\min} + 1$ caracteres cada una. De los caracteres que aparezcan se interpretan todos como 0, e.d., ausencia de figura, excepto uno en particular que se interpreta como 1, y es el definido en la línea *uno*. De esta manera cada cual puede emplear para indicar ceros y unos los caracteres que le permitan ver mejor la forma. Por ejemplo, el autor escribió las formas viendo el fichero con la fuente Wingdings 2 a tamaño 5, y para la combianción de fuente, tamaño y editor de texto consideró que como mejor se veían las formas es empleando para 0 y 1 los caracteres que se pueden ver en el fichero formas.fdf. Los caracteres nulo (0), espacio (32), tabulación horizontal (8), así como los finales de línea 10 y 13 (varía según el sistema operativo) no se pueden emplear como uno, pero nótese que el espacio y la tabulación sí se pueden emplear como cero.

A todas las formas de un bloque son de aplicación todos los parámetros de la cabecera excepto número, que va aumentando de uno en uno dentro del bloque como se explicó. Cada *numero* de un bloque tiene que ser mayor que el número de la última figura del bloque anterior, pero no necesariamente una unidad mayor exactamente. No es necesario que el *numero* del primer bloque sea cero. Ninguna figura puede tener el número 0200 (e.d., 128), porque está reservado para la forma vacía; es decir, que un punto con ese número de forma tiene una representación vacía (pero existe, en su posición, y se puede seleccionar o emplear para cualquier otro fin).