Desarrollo de Procedimientos y Herramientas orientados a la clasificación de las Entidades Gráficas en entornos CAD para su futura explotación en Sistemas de Información Geográfica.

Reinaldo Togores Fernández; César Otero González; Agustín Navarro Morante; Lorenzo Gutiérrez Malde.

Departamento de Ingeniería Geográfica y Técnicas de Expresión Gráfica

Universidad de Cantabria

Avda. Los Castros, s/n 39005, Santander

e-mail: togoresr@ccaix3.unican.es; Tfno.: 942 201790; Fax.: 942 201703;

1. Resumen:

La presente comunicación recoge las experiencias de tres proyectos diferentes en que los autores han participado durante los últimos nueve años, dos de ellos dentro del marco de convenios de la Universidad con instituciones locales. Estos proyectos han exigido elaborar procedimientos que permitieran incorporar a los objetos CAD la información necesaria que facilitara su ulterior explotación en procesos de tipo SIG.

Se describen dichos procedimientos así como las herramientas de producción desarrolladas para diferentes versiones de AutoCAD, concluyendo la exposición con una herramienta implementada sobre Visual LISP para AutoCAD MAP 2000 que sirve de instrumento para la codificación de la información geográfica de acuerdo a los criterios enunciados en la *Norma de Intercambio de Cartografía Catastral (NICCa)* [10]. Esta herramienta presenta, para cada Tema, mediante Cuadros de Diálogo, los Grupos y Subgrupos que corresponden al tipo de entidad seleccionada (lineal, centroide, perímetro, punto o texto), realizando una comprobación del tipo de objeto CAD para desechar aquellos que no correspondan a las características de dicha entidad. Los códigos se incorporan en el formato de *Datos de Objeto* de MAP siendo exportables, junto a la correspondiente geometría y de manera directa, a los formatos .MIF de MapInfo, Shapefile de ArcView y Coberturas ARC/INFO que son en el momento presente los más utilizados de la industria SIG.

2. Abstract

The present paper deals with the experiences undertaken in three different projects in which the authors have taken part during the past nine years, two of them within the framework of collaboration agreements between the University of Cantabria and local institutions. These projects have demanded the definition of procedures for the attachment to CAD entities of the data needed for their future use in GIS environments.

These procedures and the coding tools developed for different releases of AutoCAD are described in detail. We conclude with a tool developed under Visual LISP for AutoCAD MAP 2000 which implements the coding of CAD entities according to the criteria exposed in the Spanish *Cadastral Cartography Exchange Standard (NICCa)* [10]. This tool operates through a Dialog Box that lists, for each Theme, the appropriate Groups and Subgroups according to the selected feature type (lineal, centroid, perimeter, point or text). The CAD entities included in the selection set to which the code is to be applied are filtered so those that are incongruent with the chosen feature type are excluded from the coding process. Feature codes are attached in MAP's *Object Data* format and, as such, are directly exportable, linked to the associated geometry, to the most widely used GIS industry formats such as MapInfo Interchange Files (.MIF), ArcView Shapefiles, and ARC/INFO Coverages.

3. Antecedentes.

Las aplicaciones de Diseño Asistido por Ordenador (CAD) para ordenadores personales (PC) han evolucionado desde los simples gestores de despliegues gráficos que fueron en su origen, hacia sistemas sofisticados cada vez más capaces de proporcionar respuestas inteligentes en una serie de campos de aplicación muy especializados. Cabe destacar entre ellos, y como tema al que nos referiremos en este trabajo, los relacionados con todo lo que se refiere a la generación automatizada de mapas temáticos, la gestión de instalaciones y las bases de datos georeferenciadas, es decir todo aquello que suele identificarse mediante los conocidos acrónimos de origen inglés AM (Automated Mapping) FM (Facilities Management) y GIS (Geographic Information Systems). La clave para todos estos desarrollos está en la posibilidad de vincular otros campos de información suplementaria a aquellos datos que resultan imprescindibles para lo que sería una definición de la entidad en un sentido exclusivamente gráfico.

Paradigma de los desarrollos que han venido realizándose en este terreno lo es la aplicación AutoCAD, que fuera la primera en comercializarse como un sistema CAD diseñado específicamente para el PC. Partiendo de su origen como un gestor gráfico en el sentido más estricto, podemos establecer una síntesis cronológica de los aportes que han venido enriqueciendo las posibilidades encaminadas a los objetivos antes apuntados. La Tabla 1 recoge dos aspectos estrechamente interrelacionados: las maneras de asignar información no gráfica a las entidades del dibujo y los entornos de programación mediante los cuales se puede asignar y gestionar esta información.

4. Experiencias sobre el tema.

Los autores han colaborado en el transcurso de los últimos nueve años en varios proyectos de gestión de la información gráfica que han ido aprovechando, a medida que aparecían, varias de estas posibilidades. En el presente informe describiremos las experiencias obtenidas en tres proyectos sucesivos, que tienen en común el hecho de haber sido desarrollados sobre la base de AutoCAD como gestor gráfico y el haber requerido dar solución al problema de la vinculación de información no gráfica al dibujo. Su estudio en conjunto resulta revelador en cuanto a las tendencias de desarrollo en este aspecto de los sistemas CAD durante la pasada década. Los medios utilizados fueron:

1. Atributos de Bloques
2. Datos de Entidad Extendidos (XDATA)
3. Tablas de Datos de Objeto

En el marco de estos proyectos se desarrollaron las herramientas necesarias orientadas a manipular dicha información, ya sea en lo que se refiere a la fase de digitalización como a la recuperación de la misma para su explotación ulterior en los sistemas de gestión elegidos. Hemos querido incluir algunos ejemplos de código fuente que aclaren los aspectos básicos de esos procedimientos para la asignación o la posterior recuperación de datos, haciendo constar que, en aras de la claridad y concisión necesarias en una exposición de estas características, en ningún caso se trataría del código final, siempre más complejo, utilizado en las aplicaciones ya puestas a punto para uso en la producción. Estos ejemplos están redactados en lenguaje AutoLISP/Visual LISP, pero seguramente permitirán encontrar claves para la implementación de procedimientos equivalentes en otros entornos de desarrollo posibles, que incluyen Visual Basic, Delphi, Java y C++ (ver Tabla 1).

Versión y Fecha	**Métodos para vinculación de información no gráfica**	**Entornos de programación**
Versión 1.4 Octubre 1983		Se introduce la utilidad para el procesamiento de *scripts* de comandos.
Versión 2.0 Octubre 1984	Posibilidad de *nombres de capas* definidos por el usuario. *ATRIBUTOS* para asociar información alfanumérica a los bloques.	Se incrementan las opciones para la ejecución de *scripts* mediante los comandos SCRIPT y RSCRIPT
Versión 2.18 Enero 1986		Primera versión del lenguaje de programación AutoLISP. Ya se había incorporado parcialmente en las versiones 2.1 y 2.16 [7]
Versión 10 Octubre 1988	Identificadores hexadecimales (*HANDLES*) opcionales de carácter permanente para todas las entidades del dibujo.	Aparece, con la Versión 10 para OS/2, el ADS (AutoCAD Development System) con rutinas en lenguaje C equivalentes a las funciones AutoLISP [6].
Versión 11 Octubre 1990	Se permite vincular *Datos de Entidad Extendidos (XDATA)* a entidades procesando ficheros *DXF* y mediante programas *AutoLISP* o *ADS*.	El Sistema de Desarrollo AutoCAD (ADS), se extiende a la mayoría de plataformas.
Versión 12 Junio 1992	La *AutoCAD SQL Extension (ASE)/Autodesk SQL Interface (ASI)* para vínculos entre *AutoCAD* y una base de datos *SQL*. La *AutoCAD Data Extension (ADE 1.0)* introduce funciones de consulta para entidades gráficas.	Versión bajo *Windows*, con gestión del entorno gráfico mediante vínculos *DDE* desde *Visual Basic*.
Versión 13 Noviembre 1994	Identificadores de entidad siempre activados. Objetos *DICTIONARY* y *XRECORD* como contenedores de datos standard. *ADE 2.0* para *R13c4* introduce el concepto de las *Tablas de Datos de Objeto*. Aparece *AutoCAD MAP R1.0* con operaciones topológicas utilizando la tecnología *ADE 2.0* [1].	Con *R13c4*, primera versión *Windows 95* para 32 bits, se introduce la *AutoCAD Runtime Extension (ARX)*, un nuevo entorno de desarrollo C++ para aplicaciones de usuario. La compañía *BASIS Software* desarrolla el entorno de programación *VITAL Lisp* para AutoCAD, origen del futuro *VISUAL Lisp* [6].
Versión 14 Febrero 1997	La utilización de objetos *DICTIONARY* y *XRECORD* pueden ahora ser gestionados como contenedores universales de datos en el desarrollo de aplicaciones de usuario.	Programación orientada a objetos del entorno CAD: • *ObjectARX*™ para C++ • *ActiveX*™ Automation (la interfaz COM para AutoCAD) • *AutoLISP*® ahora con el Entorno de Desarrollo *Visual LISP*™ como opción La interfaz *COM* hace posible el desarrollo de aplicaciones en lenguajes como *Delphi*, *Java* y versiones standard de *Common LISP* [2].
Versión 2000 1999		*Visual LISP* se incorpora al núcleo de *AutoCAD*. Se incluye la versión definitiva de la jerarquía de objetos *ActiveX* de *AutoCAD MAP* [5].

Tabla 1 - Síntesis cronológica de la aplicación AutoCAD

4.1. Información Asociada Mediante Atributos de Bloques.

A principios de la década de los '90 se acometió la conversión a formato digital de los planos de las redes de la empresa ELECTRA de Viesgo, encargada del suministro eléctrico en la Comunidad Autónoma de Cantabria. A la representación de la red se deseaba incorporar datos como serían las referencias de consumidores, los códigos identificadores de las subestaciones transformadoras y de los nodos de interconexión, las referencias a planos de detalles del soterramiento de cables y otras informaciones relacionadas. Estas referencias y códigos serían procesados mediante programas de gestión de bases de datos para generar mapas temáticos a partir del resultado de las consultas realizadas. Como resultado de la asesoría realizada, se recomendó como único método práctico disponible en el momento del diseño de la aplicación, la inclusión de atributos alfanuméricos variables, tanto visibles como invisibles, en *referencias de bloque* [7].

4.1.1. Bloques con atributos.

Los *bloques* permiten reunir un número variable de entidades gráficas en un único objeto contenedor. Este objeto contenedor BLOCK o *definición de bloque*, puede insertarse en los espacios gráficos o anidarse en otras definiciones de bloque generando objetos *referencia de bloque* (entidades de tipo INSERT). El concepto de referencia de bloque corresponde a un símbolo definible por el usuario y que está ligado a una geometría de componentes gráficos almacenada en un apartado específico de la base de datos del dibujo (tabla de bloques del archivo DWG). Puede entonces el modelo construirse a partir de ocurrencias (copias transformadas) de esa geometría. La ocurrencia de un símbolo contiene las mismas entidades gráficas de la definición de bloque original, aunque sometidas a desplazamiento, rotación y cambios de escala [11].

La incorporación a la definición del bloque de cadenas alfanuméricas como entidades del tipo TEXT no presentaría dificultades de acuerdo a esta definición. Sin embargo, la presencia de datos constantes sería de una utilidad relativamente escasa. La posibilidad de incluir como parte de una definición de bloque información alfanumérica variable para cada ocurrencia constituye una anomalía respecto a la definición estricta de bloque que presenta dificultades obvias. Ya no se trataría de una simple copia transformada en un sentido estrictamente geométrico de la definición original. Para dar solución a este caso se recurre a un proceso similar al utilizado para generar el bloque mismo: se crea una entidad gráfica especial, la entidad ATTDEF o *definición de atributo*, que actúa como plantilla, reservando un lugar y predefiniendo las características visuales que adoptará dicha información [13].

Esta viene a concretarse en su contenido literal sólo en el momento de generarse cada ocurrencia particular, es decir en el momento de la inserción del bloque (Figura 1) y pasando a generar una entidad nueva, el objeto *referencia de atributo* (entidad ATTRIB), asociados siempre al objeto *referencia de bloque* (INSERT). La *referencia de atributo* se asocia a la *referencia de bloque* estableciendo una secuencia en la base de datos de entidades, de manera tal que la entidad INSERT actúa como cabecera de una sucesión de entidades ATTRIB que viene a concluir con una entidad especial *fin de secuencia* (SEQEND).

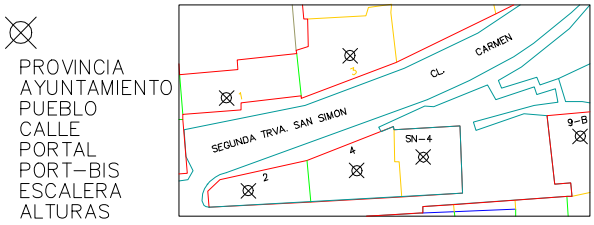

Fig. 1 Bolque con atributos y sus inserciones. En este caso los atributos se han definido como *invisibles.*

Tanto el objeto *definición de atributo* como el objeto *referencia de atributo* no son más que ocurrencias de subclases (*"AcDbAttributeDefinition"* y *"AcDbAttribute"*) derivadas de la subclase de objetos de texto *"AcDbText"* que forma parte a su vez de la clase de los objetos gráficos del dibujo *"AcDbEntity"*. En realidad lo que estamos haciendo entonces no es más que añadir al dibujo textos que se relacionan con la cabecera del bloque (INSERT) por su posición secuencial en la base de datos, modificados para añadirles una etiqueta identificadora. El recurso es aunque simple, sumamente efectivo, pero con las limitaciones que expondremos más adelante (ver apartado 4.1.3).

4.1.2. Extracción de la información contenida en los atributos.

El programa suministra el comando _ATTEXT para la extracción de estos atributos a un fichero de texto con formato a elegir entre CDF o SDF, que pueden ser leídos a su vez desde los programas de bases de datos más usuales. Los datos a extraer y su ordenamiento deberán ser definidos mediante un fichero de texto que actúa como plantilla de extracción (figura 2).

La mediación obligada de un fichero externo restaría eficacia a procesos que no exijan este paso intermedio. Esto puede obviarse, sin embargo mediante el desarrollo de funciones de usuario que gestionen dichos atributos.

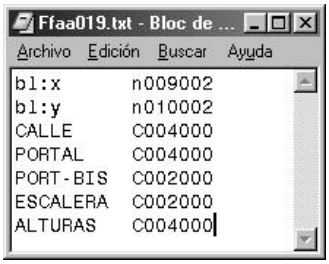

Fig. 2 - Plantilla para extracción de atributos

La Figura 3 muestra el código fuente LISP para una función de este tipo, desarrollada empleando la recursión, método usual en LISP cuando se trata de operaciones sobre listas u otras secuencias. Esta función es recursiva por la cola (tail-recursive), con lo que se asegura un alto grado de optimización una vez compilada [8].

```
;;; Recibe como argumento un Nombre de Entidad (ENAME). La función principal
;;; comprueba que se trate de un bloque y que este bloque posea atributos.
;;; La extracción se efectúa mediante una función recursiva que recorre la base de
;;; datos de entidades hasta llegar a la entidad Fin de Secuencia (SEQEND).
;;; Devuelve los valores como pares punteados dentro de una lista de asociación
;;; con el formato: '((CLAVE1 . DATO1)(CLAVE2 . DATO2)...(CLAVEn . DATOn))

;;;Función REC_ATRIB (recursiva): Lee los atributos de una referencia de bloque
(defun rec_atrib  (noment / )
  (cond                          ;cláusula terminal: cuando se llega a la entidad SEQEND
    ((equal (cdr (assoc 0 (entget noment))) "SEQEND") lista); devuelve el valor de LISTA.
    (t (setq lista
         (append lista                                  ;acumula en la variable LISTA
           (list (cons                                  ;los valores de CLAVE y DATO
             (cdr (assoc 2 (entget noment)))            ;en formato de par punteado
             (cdr (assoc 1 (entget noment)))))))        ;y efectúa la recursión con la
       (rec_atrib (entnext noment))                     ;próxima entidad como argumento
)))

;;; FUNCIÓN PRINCIPAL ATRIBUTOS:
(defun atributos  (noment / lista)
  (if (and                                              ;si se trata de un bloque
      (equal (cdr (assoc 0 (entget noment))) "INSERT")  ;y tiene atributos
      (> (cdr (assoc 66 (entget noment))) 0))           ;se llama entonces a la
    (rec_atrib (entnext noment))                        ;función recursiva REC_ATRIB
    nil                                                 ;en caso contrario se devuelve NIL
))
```

Fig. 3 - Función LISP para lectura de atributos.

4.1.3. Limitaciones al uso de Bloques con atributos.

La utilización de los atributos para asociar datos alfanuméricos a los bloques tiene una limitación: las referencias de bloque son por su naturaleza *entidades de carácter puntual*. La única información geométrica a ellas asociada se reduce a los resultados de las transformaciones aplicadas a la definición original, es decir, un punto de inserción, los factores de escala en X, Y, Z y el ángulo de rotación. Una entidad de naturaleza lineal o de superficie, por este motivo no resultaría adecuadamente representada mediante un bloque, aún cuando su apariencia en pantalla pudiera sugerir lo contrario.

4.2. Información Asociada Mediante Datos de Entidad Extendidos (XDATA).

La dificultad apuntada más arriba en cuanto a las limitaciones del método de los atributos se hizo evidente durante los trabajos de digitalización de los planos del Metro de la ciudad de Bilbao. En este caso se quería asociar a cada elemento de la vía los códigos de Instalación, Serie y Unidad utilizados en los sistemas de

control y mantenimiento de la empresa operadora. Como parte de la asesoría encargada a uno de los autores por *CIC, Consultoría Informática, s.l.*, [14] se propuso una estructura para los planos donde cada capa era identificada mediante un código asociado a la jerarquía Instalación-Serie. Pero quedaba por resolver la manera de asignar a cada objeto dentro de una misma capa sus *códigos individuales* como Unidad. Aún cuando estaba prevista la utilización de AutoCAD MAP en la explotación y mantenimiento de los planos por la Oficina Técnica del Metro, la asignación inicial de códigos sería realizada por la empresa de servicios técnicos contratada para la digitalización, y por ello debía programarse de manera que fuera asequible a las versiones normales de AutoCAD. Justo esta posibilidad era la brindaban, desde su introducción con la versión 11, los Datos de Entidad Extendidos.

4.2.1. Estructura de los Datos de Entidad Extendidos (XDATA).

Cada una de las entidades de AutoCAD admite hasta 16 Kilobytes de información como datos extendidos. En este caso el contenedor de datos no es ya un nuevo objeto ATTRIB que se ubica dentro de una secuencia en la base de datos del dibujo, sino que esos datos se integran dentro del registro de cada entidad gráfica (y también de las tablas de símbolos y otras entidades no gráficas) como campos de información adicionales. Otra ventaja de los XDATA sobre los atributos de bloque es la variedad de tipos de datos que son admitidos. La Tabla 2 presenta una reseña de estos tipos de datos, con el código de grupo que sirve para identificarlo dentro del registro de datos de la entidad.

Código DXF:	**Tipo de Dato:**	**Observaciones:**
1000	Cadena de Texto	Hasta 255 bytes de longitud
1001	Nombre de Aplicación	Hasta 31 bytes de longitud
1003	Nombre de Capa	
1005	Identificador de entidad	
1010	Valor de Punto 3D	3 números reales
1040	Número Real	Valor de coma flotante
1070	Número Entero	Entero de 16 bits (con o sin signo)
1071	Entero Largo	Entero de 32 bits con signo (gestionable por aplicaciones ARX)
1002	Cadena de Control	"{" ó "}" permite a la aplicación agrupar los datos como listas.
1004	Chunks de Datos Binarios	Hasta 127 bytes (gestionable por aplicaciones ARX)
1011	Posición en el SCU	Se actualiza al transformar la entidad propietaria.
1012	Desplazamiento en SCU	Se escala, gira o refleja pero no se estira ni se desplaza.
1013	Dirección en el SCU	Vector unitario girado o reflejado, no se escala, estira o desplaza.
1041	Distancia	Se escala igual que la entidad propietaria
1042	Factor de Escala	Se actualiza al transformar la entidad propietaria.

Tabla 2 - Tipos de Datos admitidos como XDATA

Al estar estos datos destinados a ser gestionados por aplicaciones de usuario, los mismos se agrupan según la aplicación que les haya dado origen. Estas son aplicaciones que deberán estar previamente registradas en la base de datos del dibujo mediante la función REGAPP. Una vez debidamente registrada, ese nombre de aplicación se incluirá en una tabla de nombre APPID mantenida dentro del archivo de dibujo.

El inicio de los campos XDATA en el registro de datos de la entidad se señaliza en el formato de intercambio DXF mediante un código -3. Los datos correspondientes a cada aplicación registrada se agrupan bajo un código 1001 al que se asocia el nombre de la aplicación. Los datos particulares se agrupan, según su naturaleza en

campos identificados mediante los códigos DXF que se indican en la Tabla 2. Los datos extendidos (así como los demás campos de datos definitorios de la entidad) se pueden recuperar mediante la función ENTGET.

4.2.2. Descripción de la Herramienta para Codificación mediante XDATA.

Esta herramienta se gestiona desde el cuadro de diálogo que muestra la Figura 4. Este cuadro aparece una vez que se han seleccionado uno o varios objetos gráficos para codificar. El proceso de selección sigue los cauces habituales en AutoCAD, quedando resaltado el conjunto de selección. Una vez concluida la selección el programa comprueba que todos los objetos seleccionados pertenezcan a una misma capa. De no ser así lo avisa y pide una nueva designación.

Fig. 4 - Cuadro de diálogo principal dela herramienta de codificación.

El recuadro *Objeto Seleccionado* presenta:

- El nombre de la Instalación a que pertenecen los objetos designados
- La descripción de la Serie correspondiente.

En el recuadro *Código de Mantenimiento*: dos casillas de lista desplegables:

- *Instalación*: el Código Numérico que corresponde a la Instalación.
- *Serie*: el Código Numérico que corresponde a la Serie.

  En caso de existir más de una opción para cualquiera de esos códigos deberá desplegarse la lista para inspeccionar las alternativas. El código que aparezca en la casilla será el asociado a la entidad. Estos códigos son propuestos de manera automática por el programa en correspondencia a la capa en que se encuentra la entidad seleccionada. No son casillas editables, puesto que necesariamente deberá optarse por alguno de los códigos propuestos en función de la capa. En caso de encontrarse errores en los códigos esto implica que se deben cambiar de capa los objetos CAD.

También en este recuadro tenemos la casilla de edición *Unidad*, para introducir el código particular que identifica a cada componente del sistema. La entrada de datos en esta casilla se limita a cuatro caracteres.

En el borde inferior izquierdo, se puede ver el tipo de objeto CAD seleccionado y su identificador. Si se ha realizado una selección múltiple, en lugar del tipo de objeto se indica “SELECCIÓN MULTIPLE”.

- Botón Cancelar: Cierra el diálogo para seleccionar otro objeto sin enlazar los datos al anteriormente seleccionado. Se emplea en caso de error en la designación.
- Botón Aceptar: Escribe los datos para el objeto y cierra el diálogo para permitir la selección de otro objeto. En caso de pulsar este botón sin entrar datos, aparece una advertencia pidiendo cancelar en lugar de aceptar.

Si el objeto seleccionado ya tuviera el código de la unidad, el mismo se muestra en la casilla de edición, permitiendo modificarlo. En selecciones múltiples con una mezcla de objetos codificados y no codificados, o con diferentes códigos de unidad, se abre un cuadro de diálogo (figura 5) listando los diferentes códigos detectados. Se podrá seleccionar cualquiera de estos códigos pulsando sobre la lista y seleccionando el botón Aceptar o haciendo una doble pulsación sobre la línea correspondiente. Si se pulsa el botón Aceptar sin una selección previa de la lista, se tomará como código el primero de arriba. El código de unidad que se elija de esta manera pasará a reemplazar los demás códigos en los objetos que forman parte del conjunto de selección.

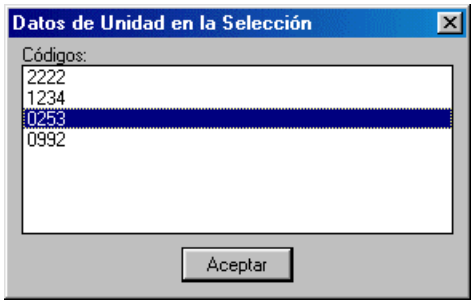

Fig. 5 - Cuadro con datos de Unidad

Todas las incidencias en la ejecución del programa se registran en un fichero nombrado como el dibujo y de extensión .LOG, que se situará en el mismo directorio que el archivo del dibujo. Estos ficheros permiten un control de calidad riguroso durante el proceso de codificación.

4.2.3. Observaciones al método XDATA.

Aunque resuelve la posibilidad de codificar de manera standard cualquier tipo de entidad gráfica, no importa cuál fuere su naturaleza, incluso los mismos bloques, deja mucho por hacer al desarrollador de aplicaciones, que debe gestionar en su totalidad esos datos que AutoCAD sólo se encarga de almacenar. Prueba de esto es el escaso número de funciones específicas para los XDATA que ofrecen los entornos de programación, apenas las necesarias para registrar la aplicación (REGAPP) y para comprobar el tamaño en bytes de la información vinculada (XDROOM y XDSIZE). La información relacionada con una misma aplicación y recuperada mediante ENTGET presenta una dificultad particular en lo que se refiere a la aparición de códigos de grupo DXF repetidos, cosa que impide su acceso, como habitualmente se haría cuando se trata de una lista de asociación, mediante la función ASSOC. Una solución sería la de recorrer la lista respetando el orden en que fueron introducidos los valores. Este procedimiento, sin embargo, pudiera no ser lo suficientemente confiable. Para la herramienta descrita se emplearon cuatro campos de tipo STRING, adicionando a la cadena de datos un prefijo con formato `"XX="`, donde `"XX"` representa un identificador particular del dato dentro de la aplicación. Este recurso, tomado de ADE 1.0, hace posible recuperar el dato de manera directa desde el API de MAP mediante la función ADE_EXPREVAL. El código de la Figura 6 demuestra la manera de añadir a cualquier entidad utilizando este formato. El argumento `noment` sería un nombre de entidad (ENAME) del dibujo actual, `nombreaplic` una cadena con el nombre de aplicación, `lis_id` una lista con los identificadores en un formato del tipo `'("DATO1=" "DATO2=" ... "DATOn=")`, `lis_val` una lista con los valores que corresponderían a dichos datos `'("valor1" "valor2" ... "valorn")`, también éstos expresados como STRING.

```
;;; Función DATOSX
(defun DatosX  (noment nomaplic lis_id lis_val / datos lista objeto)
  (setq datos       ;;Paso 1: Prepara la lista de Xdata
    (list -3
      (cons nomaplic
        (foreach term  (mapcar 'strcat lis_id lis_val)
        (setq lista (append lista (list (cons 1000 term))))
  ))))
                    ;;Paso 2: Añade la nueva lista de datos a la entidad
  (setq objeto (append (entget noment)(list datos)))
  (entmod objeto))  ;;Paso 3: Modifica la entidad
```

Fig. 6 - Código para Vincular XDATA a un objeto.

La Figura 7 muestra el código de una función que permite recuperar desde AutoCAD MAP en formatos real, entero, cadena o punto, los valores XDATA a partir de los identificadores que les fueron asociados. Por ejemplo, `(leedato objeto "MIAPLIC" "DATO3" "string")` pudiera devolver `"777"`, mientras que `(leedato objeto "MIAPLIC" "DATO3" "real")` devolvería en ese caso `777.0`

```
;;; Función LEEDATO
;;;  noment = Nombre de entidad (ENAME)
;;;  nomaplic = Nombre de la aplicación (STRING)
;;;  cod_id = Identificador del dato (STRING)
;;;  tipo = Valores permitidos: "string" "long" "short" "real" "point"

(defun leedato (noment nomaplic cod_id tipo)
  (ade_expreval noment (strcat "$" cod_id "@" nomaplic) tipo)
)
```

Fig. 7 - Función para recuperar los XDATA vinculados a un objeto mediante ade_expreval

4.3. Datos asociados mediante Tablas de Datos de Objeto.

El Ayuntamiento de Santander ha encargado recientemente una nueva cartografía a partir de la restitución digital de vuelos realizados con este fin. El objetivo es el de renovar su base cartográfica actual, compuesta por planos que abarcan el territorio sólo de manera parcial y con notables desigualdades en cuanto a su precisión y continuidad. Pero el Plan General de Ordenación Urbana (PGOU) de reciente aprobación, se expresa gráficamente sobre la antigua base cartográfica. De ahí la necesidad de volcar ahora las categorías del PGOU sobre la nueva cartografía. Como parte de la asesoría que viene realizando la Universidad de Cantabria en el proceso de implantación de un Sistema de Información Territorial para este Ayuntamiento [12] se ha diseñado un proceso de trabajo que utiliza las prestaciones GIS incorporadas a AutoCAD MAP, y en particular el uso de *Tablas de Datos de Objeto* en el proceso de codificación de la nueva base cartográfica.

4.3.1. Las Tablas de Datos de Objeto.

En el caso descrito en el apartado 4.2 el contenedor de la información necesaria para la codificación es el mismo objeto gráfico. Ya a partir de la versión 13 AutoCAD incorporó objetos concebidos como contenedores de datos no gráficos dentro del mismo archivo de dibujo. Esto, junto a la posibilidad de desarrollar nuevas clases de objetos personalizados mediante ObjectARX (ver Tabla 1) hizo posible el desarrollo para AutoCAD MAP de un contenedor de datos adecuado a las prestaciones GIS: las *Tablas de Datos de Objeto* (*AcMapODTable*).

Cuando un objeto gráfico tiene datos de objeto asociados, esa información se guarda en *una Tabla de Datos de Objeto*. Una Tabla de Datos de Objeto contiene registros de estructura similar, determinada por la definición previa dicha Tabla. Los Datos de Objeto resultan preferibles a los XDATA en cuanto esta estructura facilita el acceso a la información desde toda una serie de funciones suministradas en el API de MAP. Internamente MAP mantiene los Datos de Objeto ya sea en objetos personalizados derivados de la clase XRECORD o en formato XDATA. El formato XDATA se prefiere en aquellos casos en que la velocidad resulta crítica (como en la construcción de topologías) mientras que el formato XRECORD permite superar cualquier limitación en cuanto al volumen de la información guardada [9]. Aún cuando la documentación de AutoCAD afirma que *"los registros individuales en una Tabla de Datos de Objeto pueden estar asociados a diferentes objetos del dibujo"* [5], en realidad las funciones disponibles actualmente crean un nuevo registro para cada objeto gráfico al que se vincula el mismo conjunto de datos [9]. La asociación entre los datos y el objeto se produce a nivel del registro individual, no al nivel de la tabla, con lo que un objeto CAD puede tener asociados registros que pertenezcan a diferentes tablas.

4.3.2. Sistema de Codificación Propuesto.

El sistema propuesto se fundamenta en la posibilidad de asignar, utilizando funciones de AutoCAD MAP, registros de una *Tabla de Datos de Objeto* a cualquier entidad gráfica. Los *Datos de Objeto* que se asignarán coincidirán con la codificación de seis dígitos establecida por la norma *NICCa* [10]. Estos códigos se estructuran según tres niveles jerarquizados:

- Primero y segundo dígitos: Temas. Capítulos independientes en que se organiza el total de la información.

- Tercero y cuarto dígitos: Grupos. Conjuntos de entidades de carácter homogéneo para cada tema. El sistema incluye 118 códigos de grupo. Los grupos contendrán entidades del mismo tipo, a saber:
  - Grupos numerados del 01 al 39: Entidades de carácter Lineal.
  - Grupos numerados del 40 al 69: Centroides de Entidades de Superficie.
  - Grupos numerados del 70 al 79: Áreas representadas por sus Perímetros.
  - Grupos numerados del 80 al 89: Entidades de carácter Puntual.
  - Grupos numerados del 90 al 99: Textos y Rotulación.

Quinto y sexto dígitos: Subgrupos. Entidades de características comunes dentro de un grupo. Se incluyen en el sistema unos 400 subgrupos.

Para ello se creará una *Tabla de Datos de Objeto* con cuatro campos: *Tema*, *Grupo*, *Subgrupo* y *Observaciones.* Este último queda disponible para la anotación de incidencias del proceso de codificación como las que se describen más abajo.

4.3.3. Proceso de Trabajo para la Codificación.

Se establecerán sesiones de trabajo a partir de consultas sobre las entidades gráficas que delimitan en la documentación actual las categorías del PGOU. En el momento de redactar esta ponencia se trabaja sobre las delimitaciones de manzanas en función del viario propuesto. Una vez extraídas de los planos actuales las delimitaciones de manzanas correspondientes a todo el territorio, éstas se guardan como un archivo nuevo que se utilizará en las siguientes sesiones de trabajo para la selección a su vez de las entidades que proceden de la nueva cartografía. La consulta de los objetos que presuntamente representarían las entidades del Plan se realiza estableciendo un buffer, cuya anchura será determinada según las circunstancias por el operador, en torno a las entidades lineales que representan las delimitaciones de manzanas tomadas del PGOU. Los objetos de la nueva cartografía que estuvieren incluidos o atravesaran este buffer se incorporarán a la sesión de trabajo actual. Este proceso puede realizarse mediante las herramientas convencionales de MAP, pero en ese caso los buffers se deberán crear uno a uno. Por ello ha sido necesario diseñar una función específica de consulta que permite realizar selecciones múltiples. Una vez importados a la sesión de trabajo actual los objetos de la nueva cartografía que cumplen el requisito de proximidad establecido, se determinará su correspondencia según los siguientes criterios:

- Se encuentra un objeto de la nueva cartografía que coincide de manera exacta con una entidad del plan.
- La coincidencia no es total y por ello la correspondencia es dudosa.
- No existe ningún objeto en condiciones de ser seleccionado.

En el primer caso, se le asignará un código en formato Datos de Objeto que la identifique como perteneciente a la correspondiente categoría. En caso de duda, se adicionará, dentro de la misma tabla, además del código de categoría, un código en el campo Observaciones que indique la provisionalidad de esta asignación. Las líneas no existentes se digitalizarán, codificándolas en el campo Observaciones como nuevas. Los objetos que fueran codificados como nuevos o dudosos estarían sujetos a la revisión y aprobación por las autoridades del Plan en la Gerencia de Urbanismo. Los objetos ya codificados se restituirán a sus archivos de origen. Debe señalarse que los atributos gráficos de estos objetos no sufren alteración alguna. Sólo se les incorpora un vínculo a un registro de la tabla, una copia de la cual se mantiene dentro de cada dibujo de origen. Cada objeto gráfico puede tener *vínculos a varios registros* de dicha tabla o de diferentes tablas, lo que implica que no será necesario *duplicar* entidades cuando en un mismo objeto gráfico coincidieran las delimitaciones de varias categorías del plan. Dicho todo esto se comprende que el proceso de codificación no implicará en general operaciones de edición de tipo CAD.

4.3.4. La Herramienta de Codificación.

La herramienta de codificación desarrollada para el proyecto descrito en el apartado 4.2 sirvió como antecedente a partir del cual desarrollar esta nueva herramienta más compleja, que incorpora el sistema de códigos de la norma *NICCa*. Al invocar la función se presenta inicialmente un Cuadro de Diálogo (ver figura 8) con tres áreas diferenciadas:

- El recuadro *Tipo de Entidad* contiene cinco botones de opción de los cuales aparecen activados sólo aquéllos válidos para el tema seleccionado.

- El recuadro *Codificar Entidades* presenta tres casillas de lista para Tema, Grupo y Subgrupo. Al seleccionar un Tema diferente se activan o desactivan según convenga los botones de opción del recuadro Tipo de Entidad, y se despliegan los grupos y subgrupos del tema que correspondan a la entidad que aparece marcada.
- El recuadro *Tabla de Datos* muestra la tabla de Datos de Objeto en que se grabarán los datos. Tiene un carácter puramente informativo, ya que la tabla de nombre TTGGSS es creada por el programa mismo si no existiera anteriormente. En caso de error en su creación aparecerá el mensaje *NINGUNA*.
- Una vez seleccionados los epígrafes deseados en las tres listas, se pulsará el botón Aceptar, con lo que desaparece el cuadro de diálogo y se solicita al operador el designar los objetos que se desee codificar. En caso de que el objeto seleccionado no corresponda al tipo de entidad marcado, el sistema lo rechazará y solicitará una nueva selección. Se permiten selecciones múltiples que serán filtradas para asegurar siempre que los objetos correspondan a la tipología de entidad elegida.

Fig. 8 - Diálogo principal de la herramienta de codificación SITSAN

Concluida la designación de objetos, aparece otro cuadro de diálogo que presenta los códigos para Tema, Grupo y Subgrupo que habían sido elegidos en el cuadro de diálogo inicial, así como un recuadro Observaciones, con una casilla de edición que permite introducir cualquier comentario adicional que se guardará en un campo más de la tabla de Datos de Objeto y puede ser empleado para la formulación de consultas.

Una vez rellena la casilla observaciones, se pulsará el botón Aceptar, con lo se codificarán las entidades antes designadas y volverá a aparecer la solicitud de *Designar Objetos*, lo que permite continuar la codificación usando los mismos códigos de Tema, Grupo y Subgrupo. Si no se deseara llenar la casilla de Observaciones durante la presente sesión, bastaría con marcar la opción "*No volver a mostrar este diálogo*", con lo que se saltaría este paso, quedando ya codificadas las entidades al concluir la designación de objetos. Para volver a mostrar el cuadro de diálogo inicial basta hacer una selección nula (pulsar INTRO sin haber designado objeto alguno).

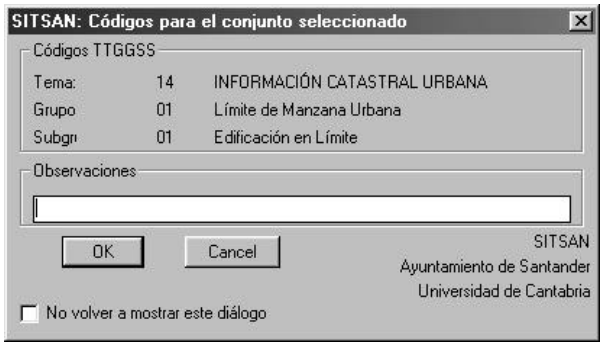

Fig. 9 - Confirmación de la selección.

4.3.5. Funciones de Programación de AutoCAD MAP

Las funciones específicas para la programación LISP [4] que se suministran con AutoCAD MAP facilitan el acceso a sus prestaciones mediante aplicaciones como la descrita. A manera de ejemplo se proponen dos funciones básicas. En primer lugar (Figura 10) una función general para la creación de tablas de objeto con cualquier número de campos, número que vendrá determinado por la cantidad de términos incluidos en las cuatro listas que se le deberán pasar como argumentos.

```
;;; Función DEFINETABLA
;;;   nomtabla  = Nombre de la Tabla (hasta 25 caracteres, sin espacios)
;;;   desctabla = Descripción de la Tabla (hasta 25 caracteres, con espacios)
;;;   lis_id    = lista de nombres de campo a definir (STRING)
;;;   lis_desc  = lista de descripciones de esos campos (STRING)
;;;   lis_tipo  = lista de tipos de dato de los campos definidos
;;;   lis_val   = lista de valores por defecto para los campos definidos

(defun DefineTabla
  (nomtabla desctabla lis_id lis_desc lis_tipo lis_val / temp)
  (if (not (ade_odtabledefn nomtabla))
    (ade_oddefinetab
      (list (cons "tablename" nomtabla)(cons "tabledesc" desctabla)
            (list "columns"
              (foreach term (mapcar 'list lis_id lis_desc lis_tipo lis_val)
              (setq temp (append temp (list
              (mapcar 'cons '("colname" "coldesc" "coltype" "defaultval") term)))))))))
```

Fig. 10 - Define una Tabla de Datos de Objeto con un número variable de campos

La creación de la Tabla es el paso previo a la vinculación de registros a los objetos gráficos. La Figura 11 muestra la función que ejecutaría este segundo paso. Se requieren como argumentos un nombre de entidad (ENAME) una lista de identificadores y una lista de los valores que a ellos corresponden.

```
;;; Función REGISTRO
;;;  noment   = Nombre de entidad (ENAME)
;;;  nomtabla = Nombre de la Tabla
;;;  lis_id   = Lista de los nombres de campos de la tabla
;;;  lis_val  = Lista de los valores que corresponden a los campos de lis_id
;;;  numreg   = Variable local que guarda el número de registros vinculados

(defun registro  (noment nomtabla lis_id lis_val / numreg)
  (setq numreg (ade_odrecordqty noment nomtabla))
  (if (ade_odaddrecord noment nomtabla)
    (mapcar '(lambda (campo valor)
       (ade_odsetfield noment nomtabla campo numreg valor))
       lis_id lis_val)))
```

Fig. 11 - Añade un registro más a la entidad cuyo ENAME se le pasa como argumento

5. Conclusiones.

Los tres proyectos descritos han explorado distintas aproximaciones al problema de asociar a las entidades gráficas la información necesaria para categorizarlas como objetos geográficos específicos. Ellos abarcan tres momentos diferentes en la aproximación a este problema en un sistema CAD:

- En el primer caso los datos adicionales de carácter literal están contenidos en objetos gráficos que heredan las características de los objetos de TEXTO.
- En el segundo caso estudiado, las limitaciones de este primer método se superan al hacer posible con los XDATA que cualquier objeto incorpore información adicional en su propio registro de la base de datos del dibujo. Información que no se limitaría a simples cadenas de caracteres alfanuméricos (ver Tabla 2).
- En el más reciente, se aprovechan las posibilidades de objetos cuya función exclusiva es la de contener información. Estos objetos permiten la gestión de información en el entorno GIS de AutoCAD MAP.

Sin duda lo realizado hasta el presente no agota las posibilidades, pero ha permitido reunir experiencias desarrollos utilizables como punto de partida para futuras soluciones aún más prácticas y eficaces. La programación LISP para AutoCAD, ha demostrado ser un medio especialmente adecuado en situaciones como las descritas, donde la rapidez de los resultados y la fiabilidad de las herramientas creadas son factores decisivos. El compromiso de Autodesk con este lenguaje, que la introducción de *Visual LISP* demuestra, permite profundizar en el mismo en la confianza de que se trata de un entorno con perspectivas de futuro, a pesar de tratarse de un dialecto de uno de los lenguajes de programación más antiguos aún en uso. Entre los desarrollos previstos [6] se encuentran el enriquecerlo con nuevos tipos de datos, sintaxis y funciones LISP, habilitar el uso de *JAVA Beans* y componentes *COM*, así como permitir la creación mediante LISP de objetos personalizados, cosa hasta ahora sólo posible con C++ a través de *ObjectArx*.

6. Referencias.

[1] Autodesk, inc.; "Frequently Asked Support Questions - AutoCAD Map R1.0", Autodesk Product Support, http://www.autodesk.com. US-MHK-td251322.doc, Revisión 1.2, enero 29, 1997.

[2] Autodesk, inc.; "Java™ - Another Choice for Applications", http://www.autodesk.com, julio 14, 1998.

[3] Autodesk, inc.; "AutoCAD 2000 Visual LISP™ Developer's Guide", http://www.autodesk.com Documento 00120-010000-5160, marzo 01, 1999.

[4] Autodesk, inc.; "AutoCAD Map® 2000 ADSRX/AutoLISP API Help", http://www.autodesk.com, julio 01, 1999.

[5] Autodesk, inc.; " AutoCAD Map® 2000 ObjectARX API HELP", http://www.autodesk.com, julio 01, 1999

[6] Carr, H.; Holt, R.; "The AutoLISP Platform for Computer Aided Design", 40th Anniversary of Lisp Conference: Lisp in the Mainstream. November 16-18, 1998, Berkeley, California.

[7] Casuso P.; Casuso I.; Otero C.; "SIGRID: Sistema Gráfico de Instalaciones de Distribución". V Congreso Internacional de Expresión Gráfica. Tomo II, pág. 49-60. Oviedo. 1993.

[8] Graham. P.; "On Lisp, Advanced Techniques for Common Lisp", Englewood Cliffs: Prentice-Hall, 1994. ISBN: 0-13-030552-9. Capítulo 2, pág. 22-24.

[9] Howard, R. D.; Subject: Re: XDATA VS OBJECT DATA, news://autodesk.autocadmap.general, Mensajes de 07/03/00 y 09/03/00

[10] Dirección General del Centro de Gestión Catastral y Cooperación Tributaria; "Norma de Intercambio de Cartografía Catastral (NICCa) V. 1.0 / 1994", Madrid; Secretaría de Estado de Hacienda, junio de 1994.

[11] Hearn, D.; Baker, M.; "Gráficas por Computadora", México: Prentice-Hall Hispanoamericana, S.A., 01 Junio, 1993. ISBN: 968-880-122-4. Capítulo 15, pág. 340-348.

[12] Otero, C.; Hoyuela, A.; Togores, R.; "Informe sobre Asesoría en la Planificación del S.I.T de la Ciudad de Santander", Santander; 1996.

[13] Sutphin, J.; "AutoCAD 2000 VBA Programmer's Reference", Birmingham: Wrox Press, Ltd., 1999. ISBN: 1-861002-2564. Capítulo 12, pág. 274.

[14] Togores, R.; "Condiciones para la Digitalización de los Planos Topográficos, Parcelarios y de Servicios de la Línea de Metro Bilbao", Santander; CIC, Consulting Informático, s.l., 1997.