Modelización de Campos Continuos en Sistemas de Información Geográfica

Lic. Silvia Gordillo

Director: Lic. Gabriel Baum

Tesis presentada al **Departamento de Informática de la Universidad Nacional de La Plata** como parte de los requisitos para la obtención del título de Master en Ingeniería de Software

La Plata, 11 de agosto de 1998

Departamento de Informática
Facultad de Ciencias Exactas
Universidad Nacional de La Plata – Argentina

Agradecimientos:

A Gabriel Baum, por la ayuda constante, el apoyo y las discusiones enriquecedoras.

A Federico Balaguer, por el trabajo conjunto que me permitió investigar y desarrollar este tema, por las discusiones y por el apoyo constante.

Al grupo que trabaja en este tema: Fernando Das Neves, Luis Polasek, Diego Cano, Javier Bazzocco, Gisela Trilla, Arturo Zambrano, Alejandra Lliteras, Cecilia Islas, Emilio D´ Angelo , Catalina Mostaccio, Gabriela Arévalo y Ana Levato por todo su esfuerzo que redundó en aportes importantes a esta tesis.

Al Profesor Robert Laurini que me sugirió el tema de esta tesis y facilitó material esencial.

A mis compañeros del LIFIA por la ayuda, el apoyo y el aliento.

Finalmente, a Gustavo y a Gabriela, que con toda su paciencia y cariño acompañaron el desarrollo de esta tesis.

Indice

1-Introducción

El incremento del uso de sistemas de información para la representación de recursos naturales se debe fundamentalmente a la mayor disponibilidad de medios tecnológicos para atacar este tipo de aplicaciones, cada vez mas complejas. En este proceso se ha producido una evolución que va desde el uso de información con fines puramente descriptivos, donde la única funcionalidad requerida es el almacenamiento y recuperación de datos, al uso de la misma con fines predictivos, para lo cual la funcionalidad requerida es considerablemente mas compleja.

El área de estudio de los recursos naturales es una de las mas favorecidas por la evolución del campo de la tecnología. Antes de esta evolución los procesos se realizaban en forma manual y se registraban en papel. Hoy en día, la mayoría de estas tareas utilizan procesamiento digital y se cuenta con paquetes de software que permiten manejar datos en este dominio con la misma facilidad con la cual se manipula texto en los procesadores de texto.

La transformación de los datos originados en la naturaleza en información utilizable como soporte de decisiones, está basada en sistemas de representación y análisis que soportan estructuras de datos poderosas y técnicas sofisticadas de organización y búsqueda.

Entre los años 1960 y 1970 los mapas se manipulaban exclusivamente como imágenes sin ningún tipo de semántica, utilizándose algún tipo de cartografía automatizada que solo proveía la capacidad de realizar modificaciones y re-dibujado de los mismos. Este tipo de sistemas fueron los primeros antecesores de la tecnología de los Sistemas de Información Geográfica (de ahora en más SIG); en ellos, la capacidad de operar con la información representada en un mapa estaba muy limitada y era imposible realizar ningún tipo de análisis exhaustivo de los mismos.

A partir de 1970 y hasta mediados de 1980, esta tecnología evoluciona con la aparición de los sistemas de bases de datos espaciales [Seeger et al.88], [Neugebauer91]. Estos sistemas se caracterizan por poseer mecanismos de almacenamiento y recuperación eficientes de datos geográficos; dichos mecanismos están soportados por estructuras de datos como quad-trees, oct-trees, etc., que permiten manipular información proveniente de fuentes tales como imágenes satelitales o sensores remotos; por otro lado permiten registrar transacciones que involucran las posiciones de las entidades representadas y el manejo de relaciones espaciales (topografía); finalmente, estos sistemas de bases de datos permiten asociar los mapas digitalizados con bases de datos que contienen información descriptiva de esos mapas (en general bases de datos relacionales), y poseen una interfaz dedicada mediante la cual se puede operar con la información espacial y los mapas definidos.

La posibilidad de automatizar el manejo de información geográfica contando con herramientas como las descriptas anteriormente, incrementa considerablemente la viabilidad de realizar análisis, tanto de información real, tal como determinar la variación en la distribución geográfica de una especie determinada, como la de

realizar análisis predictivo, como por ejemplo, evaluar el impacto ambiental de la tala de un bosque o la construcción de una represa.

Resuelto el problema de la representación y manipulación de elementos espaciales y existiendo ya buenas herramientas de análisis, a partir de la década de los 90, el esfuerzo de investigación y desarrollo se ha centrado en resolver temas tales como:

- la asociación entre las características espaciales y las descriptivas
- la mejora de interfaces para comunicar ideas sobre aspectos complejos como manejo de objetos globales y medio ambiente;
- establecer buenos mecanismos de comunicación entre los distintos especialistas que están involucrados en este tipo de sistemas,
- disponer de herramientas conceptuales para modelización que permitan expresar las decisiones tomadas por dichos especialistas

Para poder cumplir con los objetivos descriptos es necesario definir modelos de diseño que permitan especificar todos los documentos generados durante las diferentes etapas del desarrollo de un sistema, desde los requerimientos hasta el diseño e implementación de los mismos. Además es importante utilizar un formalismo uniforme que permita mejorar la comunicación entre especialistas en informática y usuarios. Es indispensable entonces que nuestros modelos de diseño reduzcan el "gap semántico" entre la realidad (geográfica) que se modela y los documentos que se generan durante el desarrollo de aplicaciones.

Dichos modelos deben tener en cuenta la complejidad de los datos que manejan los SIG y ser compatibles con la variedad de herramientas que permiten manipular este tipo de información. Entre las facilidades que se deben proveer se pueden mencionar la capacidad de:

- representar tipos arbitrarios y procedimientos que interactúen con diferentes fuentes de información. Por ejemplo, operar con datos representados con diferentes estructuras (raster y vector).

- consultar, modificar, insertar y borrar información multimedial (incluyendo búsquedas asociativas). Es decir, la posibilidad de recuperar objetos basándose en sus relaciones con otros más que en sus características individuales.

- tratar con distintas fuentes de datos de una manera uniforme. Esto incluye acceso a un dato en una fuente determinada y la migración de esa fuente a otra.

En los últimos años se han realizando numerosos esfuerzos para llegar a la definición de un modelo que permita especificar las etapas del desarrollo de una aplicación de esta naturaleza. La experiencia ha demostrado que el costo de mantenimiento, reuso y extensión de este tipo de sistemas es mucho mayor si no se aplican sistemáticamente los principios de la ingeniería de software moderna.

Una linea cada vez más estudiada está dirigida al uso de la tecnología de orientación a objetos para modelar e implementar sistemas SIG [Medeiros et al.94], [Tryfona et al.95]; debido a las posibilidades que esta tecnología brinda para la representación de información compleja a través de la combinación de las ideas de encapsulamiento y abstracción de datos, y la utilización de polimorfismo que permite construir sistemas a partir de componentes potencialmente interoperables, entre otras características, resulta una de las soluciones mas apropiadas.

El objetivo de esta tesis es la definición de un modelo de diseño, basado en la tecnología orientada a objetos que permite definir aplicaciones geográficas y contar con las herramientas necesarias para su mejor comprensión. En particular, se utiliza el modelo definido en [Gordilloet al.97] y se lo extiende para permitir la especificación y manipulación de información continua en el espacio. Esta información resulta particularmente compleja de representar y manipular dadas sus características intrínsecas:

- la definición de los valores de sus atributos en todas las posiciones de un campo (información no discreta)
- la falta de definición en sus límites y
- la existencia de discontinuidades en sus valores.

El aporte mas importante de este trabajo es la definición de una arquitectura que permite la definición y el tratamiento de un campo continuo como un objeto del sistema; esto significa:

- El campo continuo se define como un conjunto finito de pares (posición, valor) conocido a través de algún método de toma de datos (la muestra origen) mas un método de estimación necesario para calcular puntos intermedios que no han sido relevados.

- Se pueden asociar distintos algoritmos de estimación a una muestra y, en el caso necesario, cambiar dicho algoritmo dinámicamente, de manera transparente a la aplicación subyacente.

- Es posible definir y asociar distintos métodos de implementación para una muestra determinada.

- El encapsulamiento del campo continuo permite utilizarlo como un objeto en sí mismo o asociar valores del campos a objetos (discretos) a través de atributos.

En el capítulo 2 se describen las principales características de los sistemas de información geográfica. Además de considerar cuáles son los principales elementos de estos sistemas, se detallan los diferentes procesos que aparecen en el desarrollo y utilización de los sistemas en particular:

- La entrada de datos, la cual tiene una importancia vital en el proceso, pues ésta determina en gran parte cuán correcta (y por lo tanto cuán confiable) es la información que se está manipulando

- La manipulación de los datos se describe a partir de las las funciones de análisis que utilizan la mayoría de los productos de SIG existentes en el mercado.
- La definición de las interfaces que definen la salida de información y el análisis que se puede realizar sobre los datos geográficos.

En el capítulo 3 se detallan las estructuras de datos que se utilizan para soportar información geográfica y se plantean discusiones sobre el uso de cada una.

En el capítulo 4 se describen los conceptos básicos del modelo de objetos que se utiliza como base para la definición de la arquitectura para manipular campos continuos. Este modelo está basado en la tecnología de "Patrones de Diseño" [Gamma et al.95] la cual se describe suscintamente incluyendo un ejemplo.

En el capítulo 5 se introducen los conceptos básicos de un campo continuo y las operaciones que habitualmente se requieren sobre este tipo de datos. Se discuten los problemas que existen actualmente para la representación y manipulación de los mismos y se describen algunas investigaciones referentes al tema.

En el capítulo 6, se describe la arquitectura definida para campos continuos sobre la base del modelo descripto en el capítulo 4. Se detalla la arquitectura y se especifican las ventajas de su definición. Para mayor claridad se define un ejemplo en el que se muestra su uso. En el capítulo 7 se presenta una implementación posible para la arquitectura definida en el capítulo 6.

En el capítulo 8 se describen algunos trabajos relacionados que utilizan técnicas de la orientación a objetos para modelizar aplicaciones SIG. También se hace una discusión sobre las ventajas de la arquitectura presentada en esta tesis.

En el capítulo 9 se describen las conclusiones, así como los trabajos futuros que completarán el proyecto.

2- Sistemas de Información Geográfica. Características

2.1 Introducción

Ciencias como la Geografía y la Cartografía tienen varios siglos de existencia describiendo y midiendo propiedades de La Tierra, sin embargo, en los últimos años el crecimiento de los desarrollos que utilizan datos geo-referenciados se ha vuelto cada vez mas importante. Este crecimiento se debe en parte a la comprensión de cómo el uso de las ciencias mencionadas mas otras como la Geología y la Meteorología contribuyen al mejor uso y aprovechamiento de los recursos naturales; y también a la evolución que se ha producido en el campo de los sistemas de información y que ha permitido el enriquecimiento y desarrollo de aplicaciones en este campo.

Los Sistemas de Información Geográfica, se caracterizan por trabajar con datos que describen fenómenos que ocurren en la superficie terrestre (también conocidos como Sistemas de Información Espacial). En este contexto, espacial es el término utilizado para referirse a datos con una ubicación en cualquier espacio, por lo que un SIG representa una perspectiva o manera de analizar y razonar acerca de este tipo de problemas.

Las áreas en las que actualmente se están realizando desarrollos utilizando SIGs resultan de una gran variedad y en muchos casos representan áreas estratégicas de una región. Entre estas áreas podemos encontrar: uso de los recursos de la tierra, utilización automática de mapas y herramientas de análisis, cartografía marina, sistemas de sensores remotos, modelos de la superficie terrestre, modelos ambientales, planeamiento de transportes, arqueología, respuesta en situaciones de emergencias, etc. Todos estos sistemas no solamente permiten modelar la realidad geográfica en cada caso, sino que además permiten realizar estimaciones para poder prever situaciones que podrían ocurrir, por ejemplo, hacer un análisis de impacto ambiental en el caso de la construcción de una obra de envergadura como una represa, un puente, etc. Estas aplicaciones de tipo predictivas son de gran uso en la actualidad.

Sin embargo, los diseñadores aún hoy subestiman la complejidad de la información que se debe manipular ya que existen varias características que hacen problemático el manejo de estos sistemas desde el punto de vista de los datos:

- El monto de información a ser procesada es usualmente muy grande. Por ejemplo imágenes diarias tomadas de satélites.
- Habitualmente el manejo de los datos es distribuido. Los datos se capturan, almacenan y procesan en distintos sitios.
- Los datos que se manipulan son extremadamente heterogéneos, tanto en término de las plataformas de software como de hardware. Los datos se organizan basados en una gran variedad de modelos de datos.

- Los objetos generalmente tienen una estructura interna muy compleja (por ejemplo compuesta) y pueden tener asociados tipos heterogéneos y distintos medios.
- Los datos muchas veces son espacio-temporales, es decir tienen una ubicación y una extensión espacial que varía con el tiempo.
- En ocasiones los datos son "inciertos" y se deben utilizar técnicas de la estadística y la inteligencia artificial para manejarlos.
- El procesamiento de las consultas requiere conexiones y joins complejos y por lo tanto se deben utilizar técnicas de resolución particulares.

Esto hace que el desarrollo de este tipo de sistemas sea una tarea ardua, en la que básicamente hay involucrados tres niveles de apreciación del mundo [Laurini et al.96], los que se muestran en la Figura 2.1

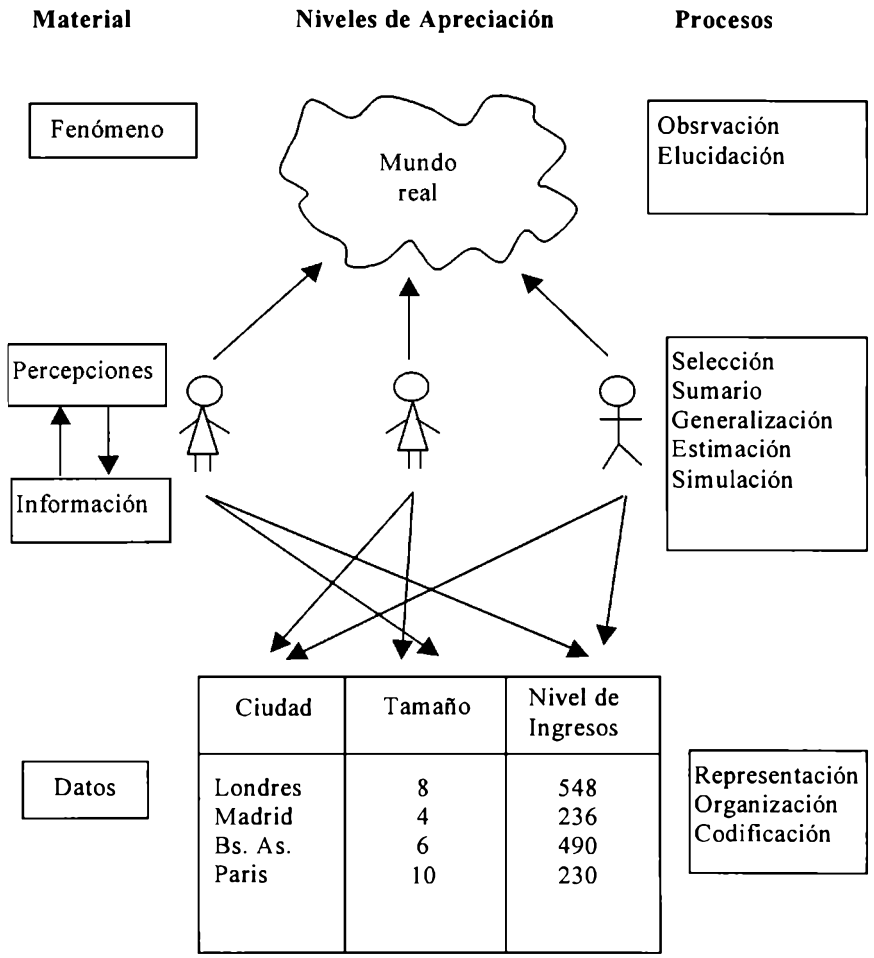


Figura 2.1: Distintas etapas del proceso de desarrollo

El primer nivel es el del mundo real, donde ocurren los fenómenos con sus propiedades, relaciones y comportamiento. El segundo nivel es el que define cómo perciben el mundo los seres humanos mediante procesos cognitivos tales como la selección, generalización y síntesis. Finalmente el último nivel corresponde a la

representación física de la información percibida y que constituye un modelo de los fenómenos observados.

A partir de la observación del mundo real se realiza la colección de datos de interés para la aplicación, esto genera una fuente de información que define los datos que deben ser ingresados en el sistema. Una vez que los datos fueron almacenados en algún formato, están disponibles para realizar el análisis espacial que se utiliza para adquirir el conocimiento necesario y tomar las acciones que se deseen. La Figura 2.2 muestra un gráfico que describe este proceso [Aronoff95].

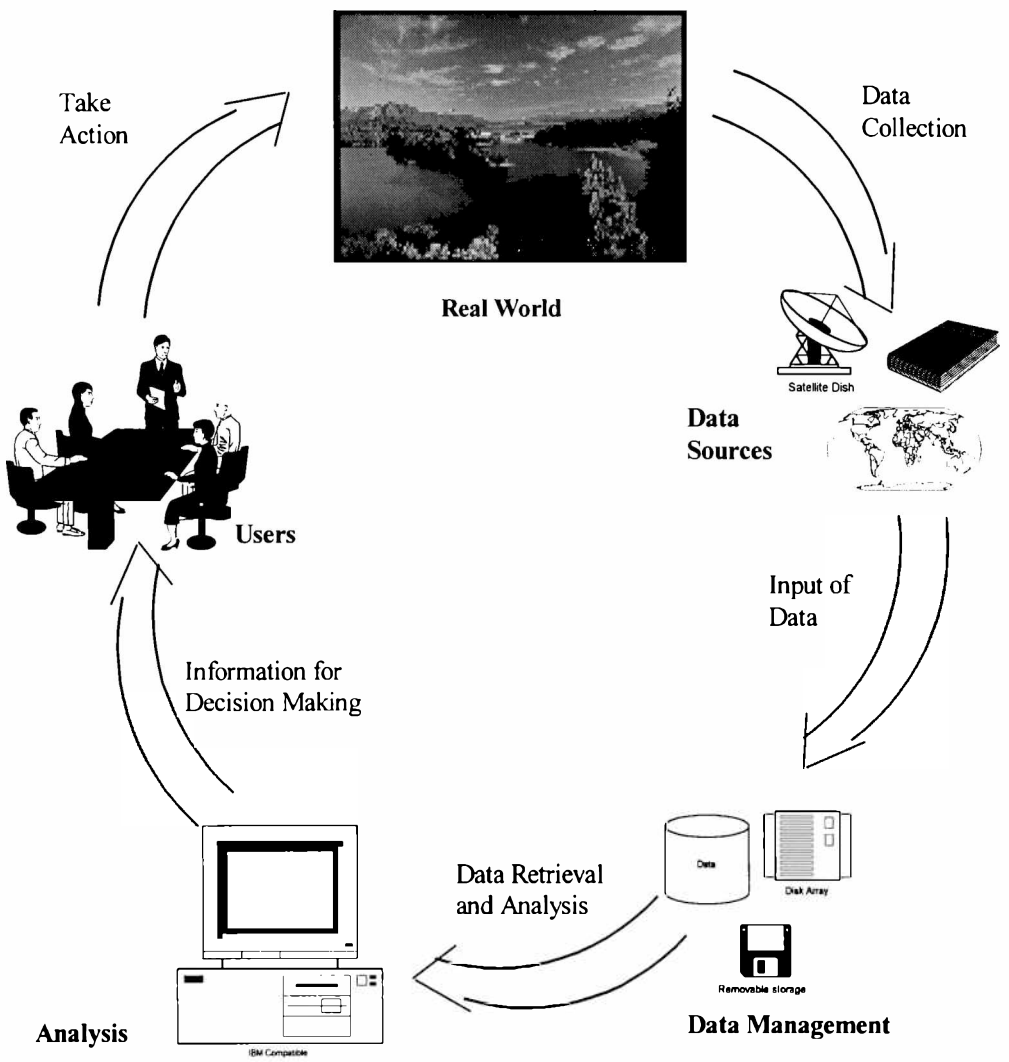


Figura 2.2: Fases del proceso de desarrollo

Estos son los principales procesos involucrados en el desarrollo de una aplicación geográfica. A continuación se describen cada uno de estos procesos en detalle.

2.2-Ingreso de la información

El proceso de ingresar los datos al sistema es generalmente uno de los mas costosos y lentos de todo el desarrollo, se estima que crear una base de datos de tamaño considerable, cuesta de 5 a 10 veces lo que el software y el hardware empleados. El ingreso de la información incluye datos espaciales y no espaciales que describen a los primeros (atributos descriptivos).

Existen varios métodos para la toma de datos en un SIG, los datos no espaciales generalmente son tomados a partir de archivos existentes o por teclado, mientras que los espaciales se pueden tomar a partir de geometría de coordenadas, digitalización manual y scanners.

Los procedimientos de geometría de coordenadas permiten ingresar datos por teclado y a partir de ellos se calculan las coordenadas espaciales, cada elemento espacial que se representa tiene un identificador que indica el tipo de elemento: punto, línea o polígono. Este método es uno de los mas precisos pero es de 6 a 20 veces mas caro que la digitalización manual [Aronoff95].

La digitalización manual es el método mas utilizado. El mapa se pone en una mesa digitalizadora y se van trazando los mapas con un cursor que calcula coordenadas en forma digital. La mesa digitalizadora codifica la posición a la que apunta el cursor con una precisión de fracciones de milímetro. Este proceso es fuertemente dependiente del operador que va trazando el mapa y que es quien determina qué puntos ir digitalizando.

En el proceso de scanning el mapa digital se obtiene a través de un detector electrónico que va recorriendo la superficie del mapa. Existen dos tipos de scanners: en el scanner "flat-bed" el mapa se pone sobre una superficie plana y el detector se mueve en dirección de X e Y; en el scanner "drum" el mapa se monta sobre una superficie cilíndrica, el detector se mueve horizontalmente a medida que el cilindro gira. El scanner genera una imagen digital, y el nivel de detalle obtenido depende del tamaño del área del mapa que el detector puede ver (spot-size), que generalmente es del orden de 0.02 mm.

2.3 Manejo de Datos

La forma más común de representación de datos espaciales es el mapa. Un mapa consiste de un conjunto de puntos, líneas y áreas, se representa sobre una superficie plana y posee una serie de atributos espaciales y no espaciales como por ejemplo nombre del lugar, colores, símbolos etc.

Un mapa cumple dos funciones diferentes dentro del sistema: la de representación de los datos que se van a manipular y la de visualización. Estas dos funciones se implementan en forma totalmente independiente e incluso puede variar el nivel de detalle de la información que se maneja. Por ejemplo, un mapa puede almacenar datos con un gran nivel de detalle para ser manipulado dentro del sistema y sin embargo manejar menos información para la presentación, dependiendo por ejemplo de la escala con que se está presentando; los objetos de un mapa también pueden presentarse de diferentes maneras en distintas presentaciones e incluso

pueden ser mostrados con diferentes tipos de datos, por ejemplo textual o mediante un símbolo.

Desde el punto de vista de la manipulación de la información los mapas presentan cuatro características principales: su posición, sus atributos, sus relaciones espaciales y el tiempo, las cuales corresponden a: dónde está el objeto, qué es, cuáles son sus relaciones con otros objetos espaciales y cuándo existió [Aronoff95].

Posición: Cada dato espacial se caracteriza por tener una posición que debe especificarse de una manera uniforme. La definición de la posición es uno de los temas mas complejos debido a que no hay un patrón específico para la representación, ya que los elementos que se representan en el sistema pueden ir desde líneas sinuosas (ríos), hasta grafos (redes de transportes).

Las posiciones de los objetos se establecen en algún sistema de referencias, que debería ser claramente especificado ya que no es lo mismo establecer que la posición de un objeto se está definiendo en coordenadas latitud/longitud que en un sistema relativo con puntos de referencia (TICs). Es necesario también contar con las funciones necesarias para transformar distintos sistemas de referencias, ya que pueden existir datos referenciados por distintos criterios.

También es necesario contar con distintos mecanismos de proyección de los datos para transformar referencias no planares (latitud/longitud) en referencias planares para poder trabajar con los mapas.

Atributos: Además de las características espaciales, los objetos poseen atributos descriptivos, por ejemplo un bosque podría incluir sus especies arbóreas, la altura promedio de los árboles, etc.

Relaciones espaciales: cuando se maneja un sistema geográfico, no solamente es importante conocer la posición de un elemento sino también como se relaciona ese elemento con otros del sistema, por ejemplo, no solamente es vital poder ubicar donde hay un incendio, sino dónde se encuentran las centrales de bomberos más cercanas para poder combatirlo. Sin embargo, reflejar todas las relaciones espaciales que existen en estos sistemas puede ser muy costoso y hasta imposible, por eso sólo algunas de ellas se definen explícitamente en el sistema y otras se calculan.

Tiempo: Muchas veces la información geográfica se refiere a un momento o un intervalo de tiempo. Este factor es crítico para el sistema ya que muchas veces determina la validez o no de un dato. El momento en el que el dato fue tomado determina la calidad de la información. En aplicaciones en donde la información cambia rápidamente, por ejemplo en aplicaciones de agricultura, un dato de una región que se está estudiando y que fue tomado 2 años atrás, probablemente no sirva para el estudio.

Existen básicamente dos modelos para representar los datos espaciales en un SIG: el modelo de *vector* y el de *raster*. Mientras que el primero establece una forma de referencia directa a los objetos geográficos con sus características espaciales (posición y tiempo) y los representa en término de puntos, líneas y polígonos, el segundo referencia a un área que se divide en zonas mas pequeñas y los objetos geográficos se determinan dentro de las zonas. Las estructuras espaciales más utilizadas en este tipo de sistemas se describen en el capítulo 3.

2.4 Análisis Espacial

Respecto de la manipulación de la información hay tres situaciones básicas que se deben tener en cuenta:

1- Acciones que requieren atributos no espaciales
2- Acciones que involucran sólo atributos espaciales
3- Acciones que combinan atributos espaciales y no espaciales

La primera categoría incluye aquellas operaciones que manipulan atributos tales como los nombres de las entidades, la composición de las entidades o valores numéricos de atributos. La característica de éstas operaciones es que ninguna trata con atributos que tienen que ver con el georeferenciamiento de las entidades, es decir ni con su posición ni con las variaciones temporales que produzcan cambios en las posiciones. Estos atributos habitualmente son almacenados en una base de datos relacional y por lo tanto recuperados desde un lenguaje de consultas relacional.

Las acciones de la segunda categoría involucran operaciones tales como determinar cuáles son los países fronterizos a uno dado, medir el área de una región, determinar la longitud total de un sistema de cañerías. Es decir no se está tratando con características descriptivas de las entidades, sino de características espaciales.

La tercera categoría incluye información tal como encontrar la cantidad de estudiantes que viven dentro de un área, o encontrar el área dañada de alguna región que ha sufrido tormentas.

En los dos últimos casos, donde se utilizan datos espaciales, la forma de resolver las consultas que se realizan depende directamente de cómo está organizada la información. Se podría pensar en realizar un pre-procesamiento de manera que la información espacial pueda ser almacenada en tablas de alguna manera y así compatibilizar los datos. Por ejemplo en el caso de los países fronterizos con uno dado se podría pensar en reemplazar una representación de grafos en donde se manipulan caminos en una red, por una tabla en la que aparezcan cada uno de los países y sus países vecinos. La información de áreas y perímetros podría ser almacenada directamente en lugar de almacenar un conjunto de coordenadas y las reglas necesarias para calcularlos. Sin embargo no todas las operaciones de este tipo pueden resolverse sin procesamiento espacial, por ejemplo, encontrar cuáles propiedades podrían ser perjudicadas por la construcción de una autopista requiere este tipo de procesamiento.

En este tipo de situaciones es muy importante tener una buena organización de la información espacial y de los atributos descriptivos de manera que la conexión entre ambos tipos de datos sea fácil y rápido.

La eficiencia con que las operaciones se realizan no solo depende de qué datos se requieren sino de cómo están organizados.

2.4.1 Organización de la información espacial

La información geográfica se organiza a menudo como un conjunto de temas, como por ejemplo, tipo de suelo, vegetación, caminos, etc. Cada uno de estos

temas es pensado como una "capa" de un mapa general que representa a una región, estas capas son llamadas *"layers"*. Cada layer puede ser manipulado como una unidad separada o superpuesto con el resto para formar el mapa completo.

Según [Aronoff95] un *Data Layer* consiste de un conjunto de características geográficas relacionadas. La elección del conjunto de características es arbitrario y depende de qué se está representando y el tipo de análisis que se quiere realizar.

Por otro lado, puede ocurrir que el área que se está tratando de representar sea muy grande y es necesario dividirla. El área se divide en unidades mas pequeñas llamadas *"Tiles"*. El tamaño de un tile depende de las restricciones del software. La grilla definida por la latitud y la longitud es ampliamente utilizada para definir tiles.

2.4.2 Operaciones sobre los datos espaciales

Una vez organizada la información, los datos están preparados para realizar diferentes tipos de análisis espacial. En forma general el análisis que se realiza sobre los datos se clasifica en tres grandes líneas [Kim95] el análisis local, el zonal y el focal. El análisis local involucra trabajar con posiciones comunes en distintos layers, por ejemplo saber que conjuntos de características hay en alguna posición determinada. El análisis zonal se define en función de posiciones que tienen el mismo valor de un atributo, por ejemplo determinar dónde crece trigo. El análisis focal trata con relaciones de vecindad y proximidad entre posiciones del mismo layer, por ejemplo, encontrar todas las posiciones donde se cultiva trigo que estén a menos de 5 km de una donde se cultiva maíz. Para poder realizar estas tareas, los sistemas cuentan con una serie de operaciones que son las que se utilizan mas comúnmente cuando se hace análisis espacial. Una lista de estas operaciones es:

a) Reclasificación y agregación
b) Determinación del centroide
c) Conversión de estructuras de datos
d) Operaciones espaciales
    Conectividad y operaciones de vecindad
e) Medidas
    Distancia y dirección
f) Análisis estadístico
    Estadísticas descriptivas
    Regresión, correlación, y tabulación cruzada

a) Reclasificacion y agregación:

Las operaciones que alcanzan esta categoría en general involucran dos situaciones posibles:

-La categoría de la información necesita ser modificada de manera de que pueda ser mejor comprendida en una aplicación determinada.

-La resolución de la información debe ser ampliada para captar características espaciales que no están disponibles en los datos.

En el primer caso, la codificación original de la información puede no ser la mas eficaz o la correcta para realizar ciertos análisis. Por ejemplo, supongamos que tenemos codificados los distintos tipos de rocas que componen el suelo desde el punto de vista geológico; un ingeniero necesita estudiar la zona para ver si el suelo permite la construcción de cierto tipo de edificio. Proveer los códigos que se utilizan en la geología para que verifique la posible construcción resultaría bastante ineficiente desde su punto de vista. Seguramente en este caso, es mejor reconvertir los códigos en "construible" y "no construible" según determinadas características del suelo. Esta operación, no solamente sugiere el cambio en los códigos, sino también en el mapa, ya que los tipos de suelos tendrán límites entre sí, que seguramente se modificarán cuando se abstrae la información de manera de identificar solamente zonas en donde se puede construir y zonas en las que no; seguramente habrá límites en los datos originales que desaparecerán.

Este tipo de operaciones son mas complejas de realizar sobre una estructura de vector que de raster. En la primera remover una línea podría significar redefinir polígonos y sus atributos, que están definidos explícitamente. Superponiendo layers la operación es mucho más simple ya que podemos asignar un promedio para cada celda en la superposición.

La agregación es una operación que se utiliza cuando es necesario incrementar el tamaño de la unidad espacial elemental, para, por ejemplo, ignorar ciertos detalles. Esta también es una operación mucho mas común en una estructura de raster que de vector, aunque también existen operaciones (del tipo del join) entre entidades representadas como vectores para obtener la información como se la desea, en este caso se requieren reglas de decisión para establecer como se van a reorganizar los atributos.

b) Determinación del centroide

El centroide es una manera habitual de encontrar la ubicación promedio de un polígono o una línea. En un polígono bidimensional se pueden calcular la ubicación promedio de sub-áreas muy pequeñas dentro del polígono y a partir de éstas determinas la ubicación del centroide. En un raster se podrían tomar los centros de cada una de las áreas que están definidas y a partir de ahí encontrar el centroide.

c) Conversión de estructuras de datos

Estas operaciones se utilizan para transformar objetos almacenados en diferentes estructuras de datos, de manera de poder operar entre ellos. Incluyen tanto la conversión de estructuras internas de datos que están almacenados con la misma filosofía, por ejemplo dos objetos almacenados utilizando el modelo de vector se quieren almacenar bajo una estructura de tipo spaghetti, así como la posibilidad de transformar objetos que fueron codificados bajo el modelo de vector a raster y viceversa.

d) Operaciones espaciales

El análisis espacial incluye operaciones que consideran las características espaciales de la información que existe en mas de un layer de datos. Se dividen básicamente en dos tipos: aquellas que involucran operaciones concernientes con la conectividad entre distintas posiciones y las relativas a características de vecindad.

i) Proximidad

Una de las operaciones principales en este tipo de sistemas es poder establecer la proximidad de dos objetos. Esta operación se utiliza en un gran número de aplicaciones, incluso cuando se está en condiciones de establecer un criterio de distancia, se puede utilizar para ubicar entidades que queden a una distancia determinada de otra.

Obviamente, la operación de proximidad incluye objetos como polígonos, líneas, puntos y sus combinaciones.

ii)Vecindad

Este tipo de operaciones nos permiten determinar, por ejemplo, cuando dos polígonos son adyacentes, o cuando un polígono es adyacente a una línea o a un punto.

e) Medidas

Estas operaciones incluyen el cálculo de distancias, longitud de perímetros de áreas, áreas, volúmenes, etc.. El cálculo de distancias entre dos áreas involucra generalmente definir un criterio para saber qué distancia se está buscando, por ejemplo puede ser la distancia mínima, o puede ser la distancia entre los centroides. En los rasters, las distancias generalmente se toman como las distancias entre los centriodes de los pixels.

f) Análisis estadístico

Hay una variedad de técnicas para realizar este tipo de análisis, que incluyen estadísticas descriptivas que implican cálculo de media, varianza, co-varianza de valores de atributos de un layer o en un área determinada de un layer. Histogramas y recuento de frecuencias, que es el estudio de la distribución de los atributos de una región. O el cálculo de correlaciones, a partir del cual se puede estudiar la distribución espacial de un atributo en distintos layers de datos.

2.5 Salida de Datos

Una de las herramientas más importantes con las que debe contar un SIG, es el software necesario para mostrar la información que se está manipulando. Esto significa, mapas, grafos, información en tablas, etc.

Las funciones cartográficas deben proveer la producción de mapas que claramente muestren la distribución geográfica de los fenómenos.

Los tipos de mapas más comunes que se utilizan en la salida de datos son:

Los mapas temáticos, que se concentran en las variaciones espaciales de un fenómeno simple (por ejemplo, la lluvia) o la relación que existe entre un fenómeno (por ejemplo tipos de suelos).

Los mapas "choropleth" se usan para comunicar magnitudes relativas de datos continuos que ocurren en ciertas áreas. Estos mapas se utilizan para representar información tal como, densidad de población, o el ingreso per cápita anual, etc. En estos mapas generalmente se indican con distintos colores los distintos valores que aparecen. En general, los contornos de las áreas de estudio están predefinidas y representan, en general áreas políticas o de otro tipo.

Los mapas de contorno, son utilizados para representar cantidades por líneas o valores iguales. Las isobaras e isotermas son ejemplos de estos mapas o las curvas de nivel que definen regiones de igual altura.

Muchas veces también es necesario mostrar la información no ya en forma de mapas, sino por ejemplo en forma de gráfico, por ejemplo, gráfico de torta, de barras, etc. , por lo que el SIG debe también proveer herramientas para este tipo de información.

3- Estructuras de Datos

3.1 El modelo de datos Vector

Este modelo provee una manera de posicionar objetos espaciales en términos de puntos, líneas y polígonos, es decir objetos 0-dimensionales, 1-dimensionales y 2-dimensionales. Este modelo asume que las coordenadas de la posición de un objeto es matemáticamente exacta, la precisión tiene que ver con la cantidad de bits de la representación. Las posiciones se registran en un sistema de coordenadas Cartesiano, así la posición de un punto es una coordenada XY, la de una línea es una lista de coordenadas XY, y la de un polígono es una lista cerrada de pares. La Figura 3.1[a] muestra un área que se quiere representar, mientras que la figura 3.1[b] muestra la representación de esa área en el modelo de Vector.

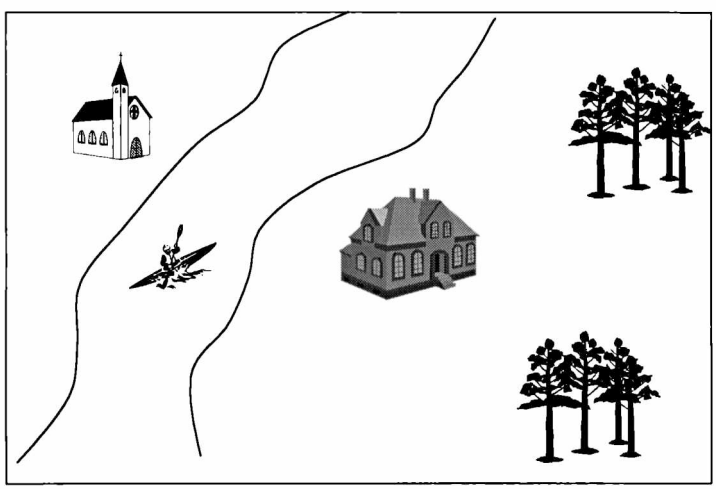


Figura 3.1[a]: Zona a representar

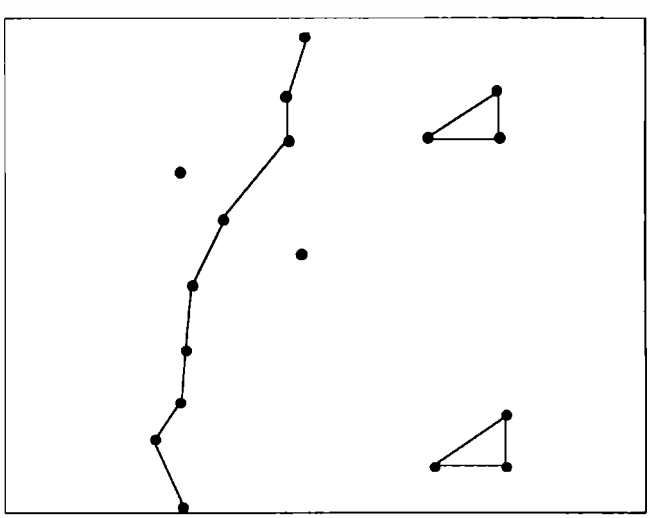
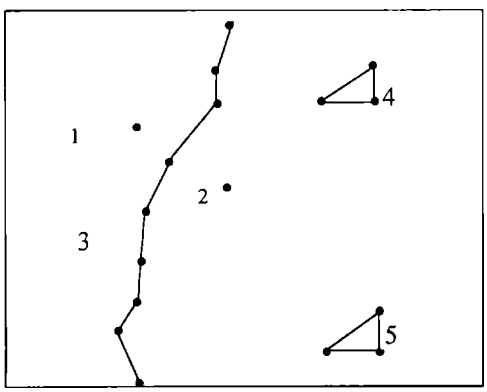


Figura 3.1[b]: Representación de vector

Los datos representados en el modelo de vector pueden ser almacenados de manera diferente, de acuerdo a las posibilidades y a las operaciones que se realicen sobre ellos. A continuación se detallan algunas de las estructuras utilizadas.

3.1.1 El modelo Spaghetti

En este modelo el mapa se traslada línea por línea a una lista de coordenadas XY. Las líneas que definen polígonos adyacentes se deben almacenar dos veces, una por cada polígono. El modelo resulta muy simple de manejar y comprender, sin embargo las relaciones espaciales entre los objetos no se representan, por lo que realizar análisis topológico (por ejemplo, qué objetos son adyacentes a un polígono) se hace mucho mas ineficiente.
En la figura 3.2 se puede ver la representación, usando el modelo Spaghetti, de la figura 3.1[a].

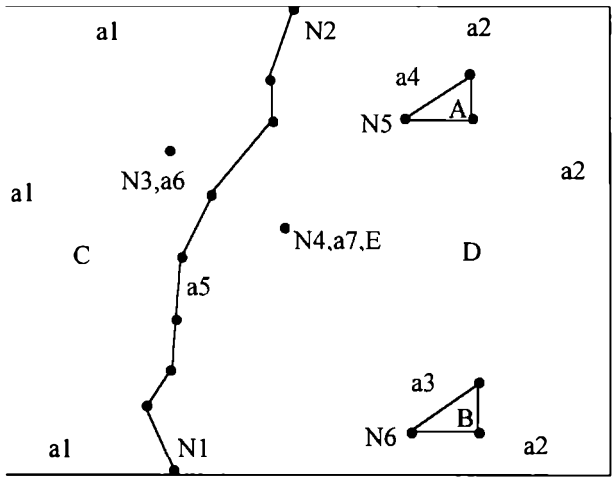


Tipo	Ident.	Posicion
Punto	1	X1,Y1
	2	X2,Y2
Linea	3	X3,Y3; X4,Y4; .....
Polig.	4	X5,Y5; X6,Y6;...;X5,Y5
	5	X7,Y7; X8,Y8;...;X7,Y7

Figura 3.2: Modelo Spaghetti

3.1.2 El modelo Topológico

Este modelo se caracteriza por reflejar las relaciones espaciales que existen entre los objetos. La entidad básica es el *arco* que se define como una serie de puntos que empiezan y terminan en un *nodo*. Un *nodo* es un punto de intersección donde uno o mas *arcos* convergen. Un polígono se representa como una cadena cerrada de arcos que determinan un área acotada. En el modelo topológico se almacenan una tabla de polígonos, una de nodos, una de arcos y una de coordenadas. En la figura 3.3 se puede ver esta representación correspondiente al área mostrada en la figura 3.1[b].



Polígono	Arcos
A	a4
B	a3
C	a1,a5
D	a2,a5
E	a7
F	área fuera del mapa

Nodo	Arcos
N1	a1,a2,a5
N2	a1,a2,a5
N3	a6
N4	a7
N5	a4
N6	A3

Arco	Nodo Inic.	Nodo Final	Polígono Izq.	Polígono Der.
a1	N1	N2	F	C
a2	N1	N2	D	F
a3	N6	N6	D	B
a4	N5	N5	D	A
a5	N1	N2	C	D
a6	N3	N3	C	C
a7	N4	N4	D	D

Arco	Coord. Inic.	Coord. Intermedias	Coord. Final
a1	X_{11},Y_{11}	X_{12},Y_{12}; X_{13},Y_{13}	X_{14},Y_{14}
a2	X_{21},Y_{21}	X_{22},Y_{22}; X_{23},Y_{23}	X_{24},Y_{24}
a3	X_{31},Y_{31}	X_{32},Y_{32}; X_{33},Y_{33}	X_{34},Y_{34}
a4	X_{41},Y_{41}	X_{42},Y_{42}; X_{43},Y_{43}	X_{44},Y_{44}
a5	X_{51},Y_{51}	X_{52},Y_{52}; X_{53},Y_{53}; X_{54},Y_{54};....	X_{5N},Y_{5N}
a6	X_{61},Y_{61}		X_{61},Y_{61}
a7	X_{71},Y_{71}		X_{71},Y_{71}

Figura 3.3: Reresentación topológica del área de la figura 3ª.

3.1.3 TIN (Triengulated Irregular Network)

Este modelo se usa para la representación de terrenos. Un TIN representa el terreno como un conjunto de zonas triangulares interconectadas. Cada uno de los tres vértices tiene asociada una coordenada XY y una coordenada Z representando

la elevación. Cada triángulo está codificado (generalmente con una letra) así como a cada uno de los nodos de los triángulos le corresponde un número.
La figura 3.4[a] muestra una representación con TIN y la 3.4[b] las tablas en las que se representa la información.

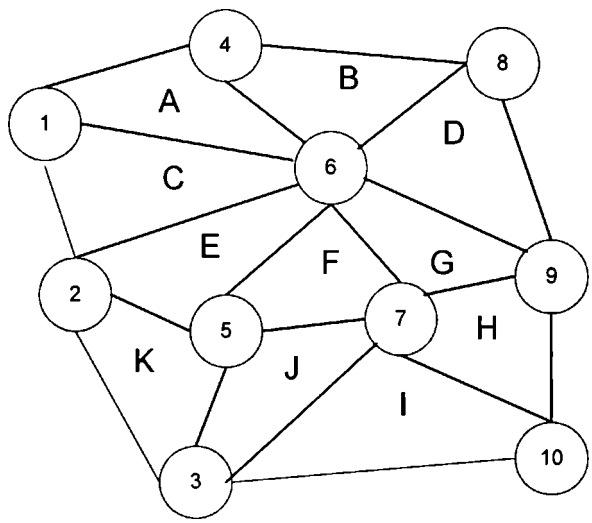


Figura 3.4[a]: Representación de un TIN

NODOS	ARISTAS
A 1 4 6	A B C
B 4 6 8	B A D
C 1 2 6	C A E
D 6 8 9	D B G
E 2 5 6	E C F K
F 5 6 7	F E G J
G 6 7 9	G D F H
H 7 9 10	H G I
I 3 7 10	I H J
J 3 5 7	J I K F
K 2 3 5	K E J

COORDENADAS XY	COORDENADA Z
1 X1,Y1	1 Z1
2 X2,Y2	2 Z2
...	
10 X10,Y10	10 Z10

Figura 3.4[b]: Tablas de representación de un TIN

Existen diferentes algoritmos para obtener las triangulaciones y la corrección de los datos depende fuertemente de cuál es el utilizado. En la práctica, aquellas triangulaciones que generan triángulos mas equiláteros son las mas correctas.

Cuando se usan TINs, características como el declive, se calculan para cada faceta y se almacenan como atributos descriptivos.

3.2 El modelo de datos Raster

Este modelo de datos organiza el espacio como una grilla dividida en celdas regulares. La ubicación de los objetos geográficos se define por la posición de la celda que ocupan. La figura 3.5 muestra la representación de raster de la figura 3.1[a]

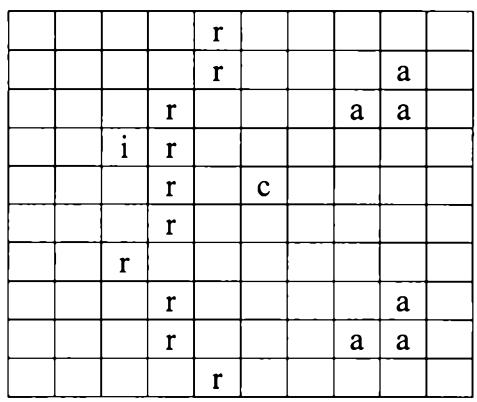


Figura 3.5: Raster representando el área de la Figura 3[a]

El área que representa cada celda determina la resolución de los datos. En cada celda se almacena un valor que representa un objeto geográfico determinado que se encuentra en esa ubicación. En este caso la unidad espacial es la celda, cada una de las cuales corresponde a un área en una ubicación específica, es decir, las unidades del raster no representan los objetos espaciales del mundo real sino sus áreas. El valor almacenado en la celda reporta la condición dentro de toda la celda. Este formato es fácilmente adaptable a datos tomados con dispositivos determinados como por ejemplo los satélites.

Como a cada celda se le asocia un único valor, atributos diferentes deben almacenarse en distintos archivos, por ejemplo los tipos de suelo de una región y la vegetación se deben almacenar separadamente. Cuando el análisis requiere de los dos tipos de información los raster deben ser combinados para obtenerla, este proceso se realiza superponiendo celdas de igual ubicación para obtener la información de cada clase. Si dentro de una celda existen distintos valores para el mismo tipo de información se debe almacenar un promedio o adoptar algún criterio para resolverlo. La resolución de los datos en un raster, como ya dijimos depende del tamaño del área que representa cada celda, cuanto más pequeña es esta área mas precisión tenemos en los datos, hoy en día la definición de una celda de un raster representando una imagen satelital es de aproximadamente 30m X 30 m.

La representación de raster puede generar considerable redundancia, por ejemplo en el caso de muchas celdas con el mismo valor, por lo que existen técnicas de compresión entre las cuales se encuentran: "run-length encoding" y "quadtrees".

Run-length encoding: este método intenta aprovechar el hecho de que existan muchas celdas vecinas con la misma codificación, en ese caso en lugar de almacenar el raster completo se almacena el valor del código una sola vez y se especifica cuáles son las celdas en las que aparece.

Existen varias maneras de realizar compresiones de este tipo, dos de ellas son el standard *run-length encoding* en donde se almacena el valor del atributo, el número de celdas con ese valor y las filas en las que aparece este valor; y el *value point encoding* en donde se especifican los valores de las celdas y la cantidad de veces que aparecen comenzando desde la esquina superior izquierda. La Figura 3.6 muestra un ejemplo de cada una de estas técnicas.

1	1	1	1	1	1	1	1
1	1	1	2	2	2	2	2
3	3	3	3	4	4	4	4
4	4	4	5	5	5	5	5
6	6	6	6	6	6	6	6
6	6	6	6	6	7	7	7
7	7	7	7	7	7	8	8
8	8	8	8	8	8	8	8

Valor	Long.	Fila
1	8	1
1	3	2
2	5	2
3	4	3
4	4	3
4	3	4
5	5	4
6	8	5
6	4	6
7	3	6
7	6	7
8	2	7
8	8	8

run-length encoding

Valor	Punto
1	11
2	16
3	20
4	27
5	32
6	45
7	54
8	64

value point encoding

Figura 3.6: Dos modelos de compresión

Por supuesto la efectividad del método de compresión depende del mapa que se esté representando, si existen muchos códigos de representación distintos

agrupados en pequeñas cantidades la compresión será menos efectiva que si están en grupos de mayor cantidad.

Los Quadtrees son otra forma de representar los rasters. En este caso, el área que representa el raster se va dividiendo en áreas regulares más pequeñas de manera de encontrar aquellas en donde los valores de todas las celdas incluidas en el área sean los mismos.

Supongamos la representación del área de la Figura 3.7, y las sucesivas divisiones que se realizan.

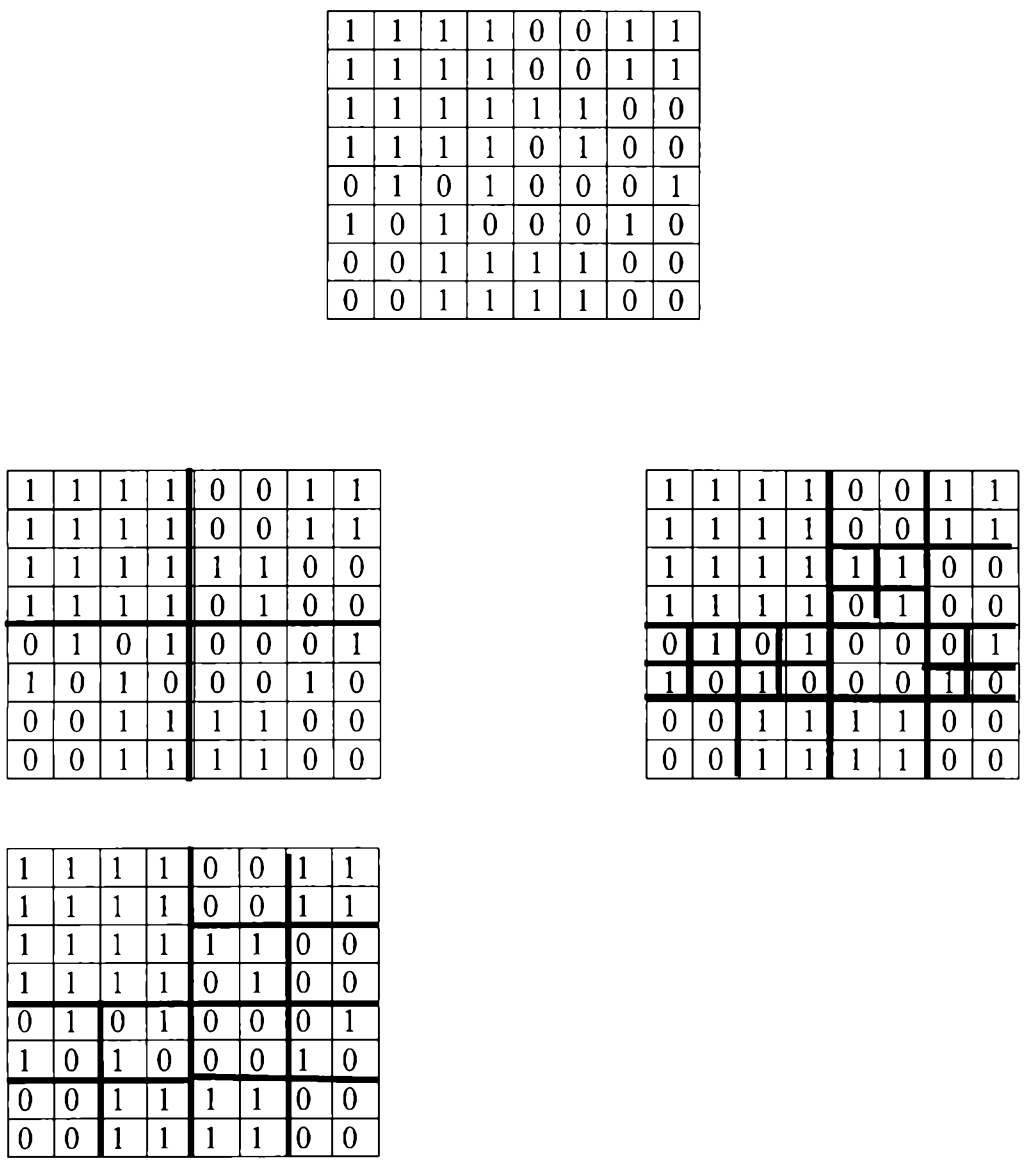


Figura 3.7: Divisiones sucesivas del área

El área se va dividiendo en cuadrantes iguales y si en cada cuadrante el valor representando el tema es el mismo, no se divide más. Para aquellos cuadrantes en donde existan celdas con distintos valores, el proceso se vuelve a repetir.

Basándose en estas divisiones, se construye un árbol en el que cada nodo tiene a lo sumo cuatro hijos representando a cada uno de los cuadrantes. La figura 10 muestra el quadtree correspondiente a las divisiones de la Figura 3.8.

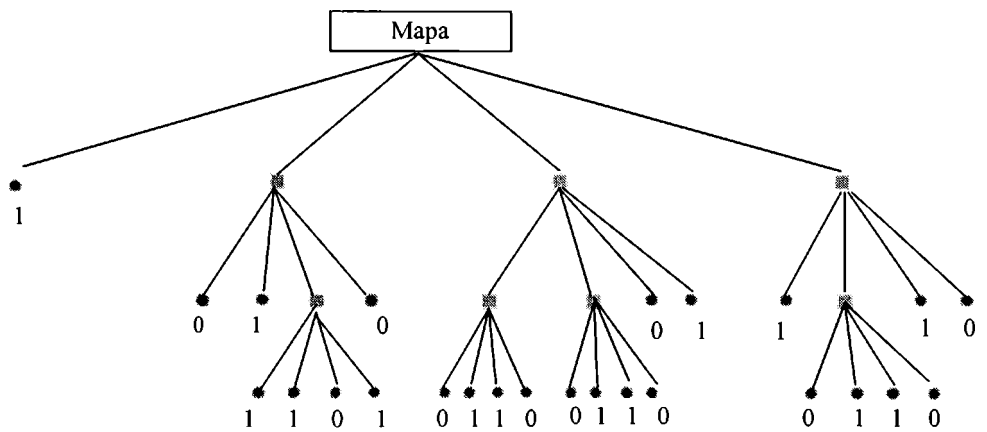


Figura 3.8: estructura del quadtree

Esta estructura permite además de optimizar el almacenamiento en determinadas ocasiones, operar el mapa por capas, ya que es posible acceder a los distintos niveles de detalle que se requieran.

3.3 Comparación entre estructuras de datos

Una de las decisiones más importantes para decidir cuál es la estructura de datos a utilizar tiene que ver con la necesidad, o no, de incluir información topológica en el sistema, es decir la necesidad no solamente de manipular la posición de los objetos sino también sus relaciones espaciales. La información topológica es a menudo muy importante y constituye la base del sistema, no obstante, si no es necesaria también puede agregar complejidad a los datos y a la aplicación.

Un raster incorpora algún tipo de información topológica implícita en los datos. La regularidad de las celdas permite un sistema de direccionamiento implícito que permite acceso a posiciones específicas. De esta manera se puede saber cuáles son las celdas que son adyacentes a una posición determinada, o se puede analizar una región acotada por un grupo de celdas.

Por otro lado, es muy costoso obtener información topológica mas compleja cuando no está almacenada explícitamente, en estos casos es mas eficiente usar estructuras de vector para manejar la aplicación.

Otros factores en la determinación de las estructuras a utilizar tienen que ver con: volumen de datos, eficiencia en la recuperación de la información, eficiencia

en la manipulación de datos, corrección de los datos, y las interfaces. Sin embargo algunos de estos factores son menos importantes que otros, la tendencia a decidir en función del volumen de datos a manipular es cada vez menos usada. La comparación entre el volumen de información generada por un raster o un vector depende directamente del contenido de la base de datos, la tolerancia aceptable y la precisión de la información.

Tradicionalmente se pensaba que para combinar información de distintos mapas era mas conveniente el uso del raster, sin embargo en los sistemas actuales existen maneras eficientes de determinar las ubicaciones de los polígonos manteniendo índices en la base de datos.

Por supuesto la situación ideal es la de aquellos sistemas (los menos) que aceptan almacenamiento de ambos tipos de manera que permiten el almacenamiento, recuperación, manipulación y despliegue de datos de tipo raster y de tipo vector. Estos sistemas proveen mecanismos de conversión para permitir el análisis de datos de ambos tipos.

4- Un modelo Orientado a Objetos para SIG

4.1 Introducción

En este capítulo se presenta un modelo para el diseño de aplicaciones geográficas basado en el paradigma de orientación a objetos. En particular, el modelo se basa en el uso de patrones de diseño que permiten resolver problemas recurrentes en esta área.

Como se ha visto en el capítulo 2, la tecnología de SIG involucra tratar con aspectos complejos que van desde la adquisición de datos, la corrección de los mismos, la representación de relaciones espaciales y de la topología y el diseño de la interfaz. Uno de los problemas mas graves que existen en este dominio es la carencia de un método de diseño que permita reflejar todas las características mencionadas y permitir obtener un producto con las propiedades deseables desde el punto de vista de la ingeniería de software, como por ejemplo, reusabilidad, modularidad, facilidades para la evolución y la modificación, etc.

En efecto, hoy en día la construcción de este tipo de aplicaciones se vuelve una tarea completamente artesanal en donde aquellos diseñadores experimentados utilizan su conocimiento anterior "on the fly" mientras que aquellos que no poseen experiencia realizan la tarea basándose en las "pruebas y errores". Por supuesto, es sumamente difícil encontrar documentación sobre estos sistemas, pues la preocupación pasa más por la precisión y la corrección de los datos que se están almacenando y en resolver las dificultades que presentan las relaciones espaciales, que por obtener un buen diseño.

Esta metodología lleva a lo que ya se ha conocido y vivido en otras áreas, los costos de mantenimiento, reusabilidad y evolución de los sistemas resultan altísimos. De hecho, ocurre más de una vez, que grupos que realizan desarrollos basados en el mismo tipo de aplicación (por ejemplo catastral) deben comenzar cada trabajo como si fuera completamente nuevo, cuando en realidad la mayor parte de los módulos de una aplicación catastral pueden ser utilizado para otra.

La tecnología orientada a objetos ha demostrado ser una buena herramienta a la hora de resolver problemas de gran envergadura, en los que la complejidad de la información que se manipula es alta y es imprescindible contar con un alto grado de integración de esa información. Como resultado de su uso se obtienen, no solamente soluciones en donde las características anteriores se mantienen (reusabilidad, evolución, modularidad) sino también sistemas interoperables, ya que los objetos pueden encapsular conocimiento que puede agruparse según las necesidades.

4.2 Patrones de Diseño

Basada en la tecnología de objetos, aparece en los últimos años una herramienta de diseño muy poderosa: los patrones de diseño [Alexander77], [Copilen et al.95], [Fowler97] [Gamma et al.95].

Los patrones de diseño representan buenas soluciones para problemas que aparecen recurrentemente dentro de uno o mas dominios. Cristopher Alexander [Alexander77] define a los patrones de diseño como: *"Cada Patrón describe un problema que ocurre una y otra vez en nuestro ámbito, luego exhibe el corazón de la solución al problema de manera que esta solución se pueda usar una y otra vez, sin hacer lo mismo dos veces"*

Tal cual se describe en [Gamma et al.95], un patrón de diseño se define habitualmente por la descripción del problema, sus relaciones, sus responsabilidades y sus colaboraciones. Estos elementos se describen en forma abstracta ya que los patrones son como "moldes" que pueden ser aplicados en diferentes situaciones.

Un patrón de diseño tiene cinco elementos fundamentales:

a) el **nombre** del patrón acorde con lo que representa
b) la **intención** del patrón, que es una descripción sucinta de cuál es el problema que resuelve
c) la descripción detallada del **problema** sobre el cual se va a aplicar el patrón
d) la definición de la **solución** al problema describiendo los elementos que la conforman (clases, objetos, relaciones, responsabilidades y colaboraciones)
e) finalmente, las **consecuencias** que resultan de aplicar el patrón y que se usa para la evaluación de diferentes alternativas de diseño.

La notación que se utilizará para describir los diferentes patrones de diseño definidos para aplicaciones SIG es la notación UML definida en [Rational97].

Como ejemplo, a continuación se muestra la definición del Patrón de diseño "Adapter" que se encuentra en [Gamma et al.95].

El patrón de diseño Adapter se encuentra entre los patrones estructurales, los cuales se utilizan en situaciones relacionadas con la forma en que deben componerse las clases y los objetos de manera de formar estructuras mas complejas.

Los patrones estructurales de *clases* usan la herencia para componer interfaces o implementaciones. Los patrones estructurales de *objetos* describen la forma en que se deben componer los objetos para definir nuevas funcionalidades.

Intención: el patrón Adapter convierte la interfaz de una clase en una interfaz apta para el cliente. Permite que clases con interfaces incompatibles trabajen juntas.

Motivación: Algunas clases que han sido diseñadas para ser reusadas no son realmente reusables simplemente porque sus interfaces no son compatibles con la interfaz necesaria para el nuevo dominio.

Considere por ejemplo, un editor gráfico que le permite a los usuarios dibujar y manipular elementos tales como líneas, polígonos, texto, etc.. La clave de la abstracción del editor es el objeto gráfico, el cual tiene una forma que es editable y se puede autodibujar. La interfaz del objeto gráfico está definida por una clase abstracta llamada *Shape*. El editor define una subclase de *Shape* para cada tipo de objeto gráfico: *LineShape* para líneas, *PolygonShape* para polígonos y así sucesivamente.

Las clases de los elementos geométricos simples (*LineShape, PolygonShape*, etc) son sencillas de implementar porque tanto las posibilidades de dibujarse como

de editarlos son limitadas. En particular todas estas subclases entenderán un mensaje *Draw* que será el que dibuja una figura.

De esta forma el cliente tendrá un mensaje del tipo *f draw* donde f es alguna figura.

Supongamos ahora que se le agrega una nueva jerarquía al editor de manera de poder manejar textos. Las clases de esta nueva jerarquía entienden un mensaje *view* en lugar del *draw*. Obviamente se debe realizar alguna modificación para que el cliente pueda o dibujar una de las figuras definidas o bien ver un texto. Una de las posibilidades es modificar el cliente de manera de que pueda determinar a cuál mensaje se invoca; esta no es una buena solución ya que implica modificar el código del cliente y si, eventualmente, se fueran agregando más clases el código se volvería poco claro. La segunda solución, es cambiar el código de las clases que manipulan el texto cambiando el nombre del método, llamándolo *draw*, lo cual resulta una solución bastante forzada, ya que los textos no se dibujan.

La tercera solución es definir una nueva clase que adapte la interfaz entre la clase que representa al texto y las de los gráficos. Esto se puede hacer de dos formas: heredando la interfaz de la clase *Shape* y la implementación de la clase *TextView,* o componiendo una instancia de *TextView* dentro de *TextShape* e implementando *TextShape* en términos de la interface de *Textview*. La Figura 4.1 muestra la solución a este ejemplo:

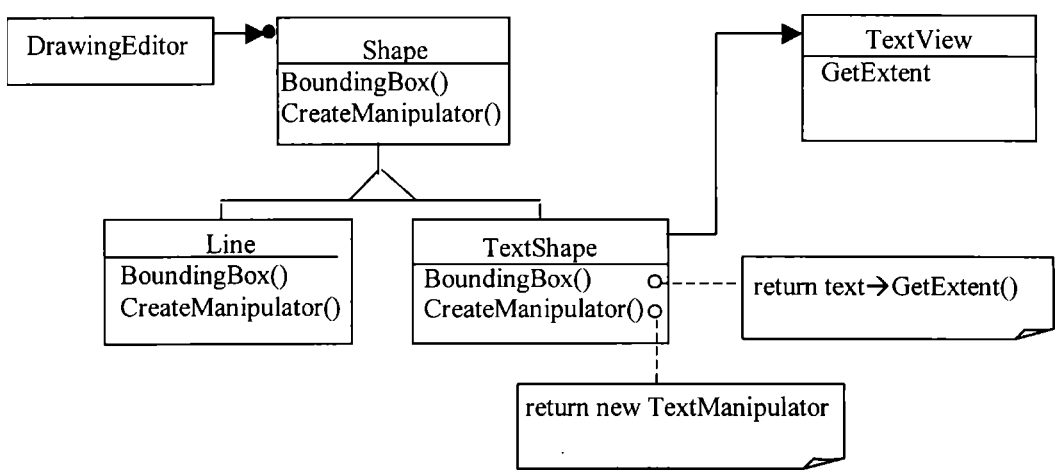


Figura 4.1: Uso del Adapter

En este caso, la clase *TextShape* es la que hace de adaptador de la interfaz, transformando el mensaje *BundingBox*, declarado en la clase *Shape* en *GetExtent* que es el mensaje que entiende la clase *TextView*.

Muchas veces el Adapter es responsable de la funcionalidad que no provee la clase adaptada. Por ejemplo el usuario podría querer copiar cualquier figura a una nueva ubicación interactivamente, sin embargo, la clase *TextView* no está preparada para eso. *TextShape* puede agregar esta funcionalidad implementando una operación CreateManipulator, que retorna una instancia de la subclase *Manipulator* que corresponda.

La Clase *Manipulator* es una clase abstracta que tiene diferentes subclases para cada forma de figura, *TextManipulator* es la que manipula los textos.

Aplicabilidad

El Patrón Adapter se puede utilizar en las siguientes situaciones:

-Se quieren usar una clase existente y su interfaz no es la que se necesita.

-Cuando se quieren crear clases reusables que van a cooperar con clases que no necesariamente tienen interfaces incompatibles.

-Cuando se necesitan usar varias subclases existentes pero es impracticable adaptar sus interfaces subclasificandolas.

Estructura

El siguiente diagrama muestra la arquitectura de los adaptadores de objetos, que trabajan con la composición.

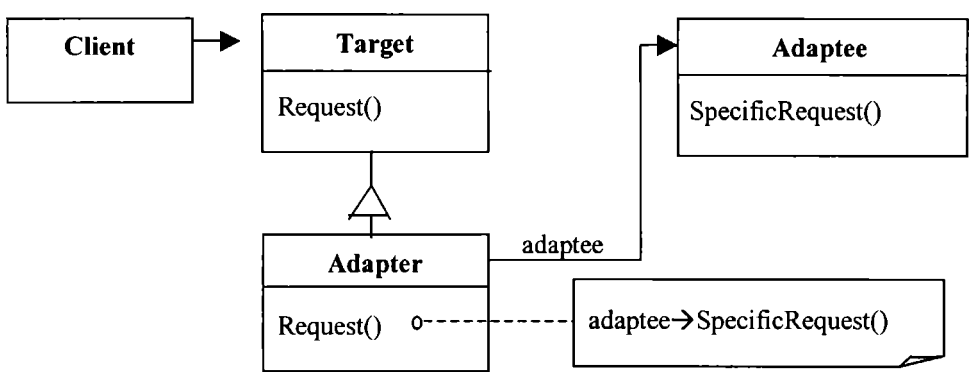


Figura 4.2: Arquitectura del adaptador de objetos

Participantes:

Target (Shape)

define la interfaz específica que usa el *Cliente*.

Client (DrawingEditor)

colabora con los objetos conformando la interfaz de *Traget*.

Adaptee (TextView)

define la interfaz existente que necesita ser adaptada

Adapter (TextShape)

adapta la interfaz de *Adaptee* a la de *Target*

Colaboraciones:

Los clientes invocan operaciones sobre una instancia de *Adapter*. A su vez, el adapter invoca las operaciones de *Adaptee* para llevar adelante el requerimiento.

Consecuencias:

Un adaptador de objetos:

- permite que un *Adapter* simple trabaje con varios *Adaptees*, es decir, el adaptado y todas sus subclases (si las hay). También puede agregar funcionalidad.
- hace mas difícil redefinir el comportamiento del *Adaptee*. Esto requerirá subclasificar la clase y hacer que el adpatador se refiera a la subclase en lugar de al *Adaptee*.

4.3 El Uso del Modelo Orientado a Objetos en Aplicaciones Geográficas

La preocupación de los diseñadores de aplicaciones geográficas por obtener información correcta desde el punto de vista de la representación (precisión, escala, etc.) genera un error bastante común en el diseño que parte de considerar las características de la representación espacial de las entidades, en lugar de pensarlas en función de su definición abstracta. El énfasis entonces está puesto en *cómo* la información se representa y no en *qué* información se debe representar. Como una consecuencia de esto, las entidades que tienen distintas representaciones (por ejemplo distintas escalas) se implementan usualmente como objetos diferentes cuando en realidad no los son.

Existen muchos trabajos que se han abocado al estudio del uso del modelo de objetos como una solución a estos problemas [Medeiros et al. 94] [Trifona et al.95] [Kosters et al.95]. Sin embargo, la mayoría de estos trabajos se basan en el modelo para representar entidades complejas pero no han estudiado cuestiones tales como la separación de las representaciones físicas de los objetos (puntos, líneas, polígonos) de su comportamiento conceptual, lo que fuerza al diseñador a asignar prematuramente características espaciales a las entidades y hace dificultoso trabajar con diferentes representaciones del mismo objeto.

Por otra parte, también se ha vuelto una práctica común extender sistemas convencionales para que manipulen características espaciales. Ejemplos de estas extensiones pueden encontrarse en [Postmesil97].

4.4 Un Modelo para Aplicaciones Geográficas

A continuación se describe un modelo orientado a objetos para diseñar aplicaciones geográficas. Este modelo está presentado en varios artículos [Gordillo et al.97], [Balaguer et al.97a], [DasNeves et al.97], [Gordillo et al.98]. El modelo está basado en el uso de patrones para resolver desde el diseño de las características espaciales de las entidades hasta situaciones particulares como la representación del sistema de referencia utilizado para las posiciones de las mismas. Este modelo es utilizado como base para el diseño, descripto en el capítulo 5, de información continua.

4.4.1 El modelo conceptual

Existen en las aplicaciones geográficas dos tipos de datos bien diferenciados. Los datos conceptuales describen a las entidades en términos de atributos descriptivos, y son los que se utilizan habitualmente en las aplicaciones convencionales. Por ejemplo, si estamos representando un país, su nombre, su lengua, la moneda que se utiliza son atributos que describen al país. Por el otro lado está la información geográfica, como la ubicación de ese país la definición de sus fronteras, etc.

Estos dos tipos de datos, definen en la mayoría de los SIG dos bases de datos bien diferenciadas: una, generalmente relacional conteniendo los atributos descriptivos, y otra espacial que almacena las características geográficas.

En el nivel funcional pasa algo parecido, las operaciones que se deben realizar se pueden considerar de dos tipos, las que operan con los atributos descriptivos y obtienen información basándose en ellos, y aquellas operaciones espaciales que no solamente van a manipular información espacial, sino que además pueden estar determinadas por diferentes fenómenos.

La discusión anterior sugiere que las características (atributos y operaciones) conceptuales y espaciales pueden ser tratadas en niveles diferentes de abstracción, sin embargo los dos aspectos deben estar cubiertos correctamente, de manera que interactúen en forma consistente.

Como la complejidad del diseño y el desarrollo reside básicamente en la definición de las características espaciales, la propuesta es realizar el proceso de diseño en dos pasos, separando el diseño de las características conceptuales de las geográficas, de manera de desacoplar los problemas haciendo posible que la tarea resulte mas fácil y clara.

Durante el primer paso, llamado Modelo Conceptual, se describe el dominio en términos del mundo real. La intención es comprender el problema proveyendo una descripción abstracta en la cual se ignoran tanto las características geográficas como los detalles de implementación. Se utilizan los conceptos de la orientación a objetos y, sólo aquellas responsabilidades que son independientes del espacio son tenidas en cuenta en esta etapa; de esta forma, el resultado obtenido no es diferente de una aplicación convencional diseñada bajo este paradigma.

En el segundo, se especifican las características geográficas de los objetos definidos en la etapa anterior y, posiblemente, aparezcan nuevos objetos que sólo contienen características espaciales. El método para adicionar estas características está descripto en el punto siguiente.

4.4.2 El modelo Geográfico

Una vez que el modelo conceptual está definido, puede enriquecerse agregándole las características geográficas. La propuesta es identificar aquellas clases del modelo conceptual que deben ser enriquecidas y definir una nueva clase que adiciona las nuevas características. Para hacer esto proponemos el uso del patrón de diseño "Decorator" [Gamma et al.95]. A continuación se describe la

arquitectura del Decorator y luego se explica cómo puede ser usado para diseñar el nivel geográfico en el modelo propuesto.

Nombre: Decorator

Intención: Adicionar responsabilidades a un objeto en forma dinámica. Provee una alternativa mas flexible que la subclasificación para extender funcionalidad.

Problema:

Se tiene modelizada una clase *Ciudad* en el modelo conceptual, que es parte de una aplicación geográfica en la cual se quieren representar aspectos climatológicos.

En la primera etapa, la clase fue pensada desde sus características conceptuales, evitando los aspectos espaciales. Desde ese punto de vista se han definido los atributos descriptivos y el comportamiento necesario. La Figura 4.3 muestra la especificación de esta clase.

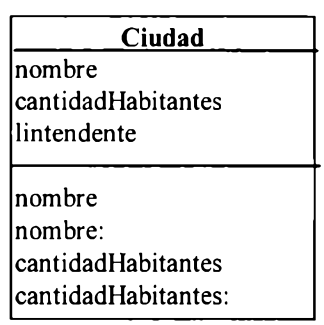


Figura 4.3: Clase Ciudad

Cuando se agrega información geográfica a un objeto, se cambia la perspectiva y se piensa cuáles operaciones espaciales se van a realizar con la información. Por ejemplo, necesitamos conocer la ubicación de la ciudad (en algún sistema de referencia, por ejemplo, latitud/longitud del conjunto de puntos que definen su contorno) su superficie, su ancho en alguna latitud, etc.. Una alternativa para agregar funcionalidad a una clase podría ser sub-clasificar, (tal cual se muestra en la Figura 4.4) pero en realidad, la solución que el Decorator provee resulta más flexible que la clasificación, ya que mientras esta última lo hace en forma estática, la primera es una solución dinámica. Esto es, cuando clasificamos y creamos un objeto de una clase particular, no existe manera de cambiar la clase del objeto. El Decorator, replica la interfaz del objeto al que decora y además define comportamiento adicional; de esta forma, delega los requerimientos que corresponden a la componente original y ejecuta las acciones adicionales.

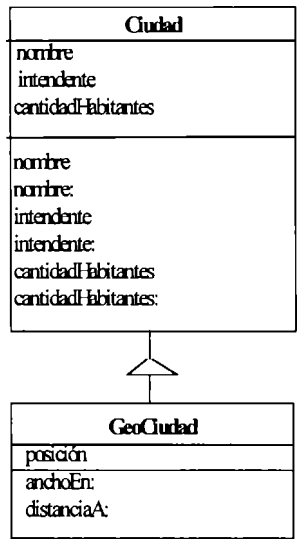


Figura 4.4: subclase que agrega características espaciales a la ciudad

A partir del Decorator, todas estas características adicionales pueden definirse y agregárselas a los objetos en forma individual. La Figura 4.5 muestra la clase *GeoCiudad* que actúa como decorador de *Ciudad.*

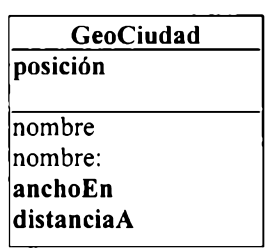


Figura 4.5: la clase GeoCiudad

Como se puede ver en la figura, la clase *GeoCiudad* replica la interfaz de *Ciudad* y define el comportamiento relacionado con los aspectos espaciales. De esta manera, cuando el mensaje *nombre* de la clase *GeoCiudad* es invocado, ésta delegará la responsabilidad en la clase *Ciudad.*

El uso de decoradores permite la modificación de la funcionalidad del sistema sin modificar las clases del esquema existente. Obviamente entre estas dos clases se debe establecer una relación que permita a la clase que actúa como decoradora, en este caso *GeoCiudad*, delegar en la clase *Ciudad* el comportamiento que corresponde a la definición conceptual. De esta manera, el esquema resultante es el que muestra la Figura 4.6.

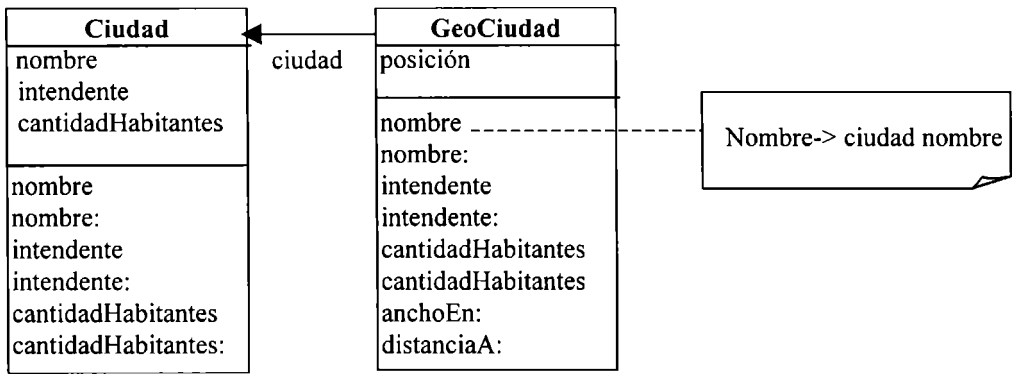
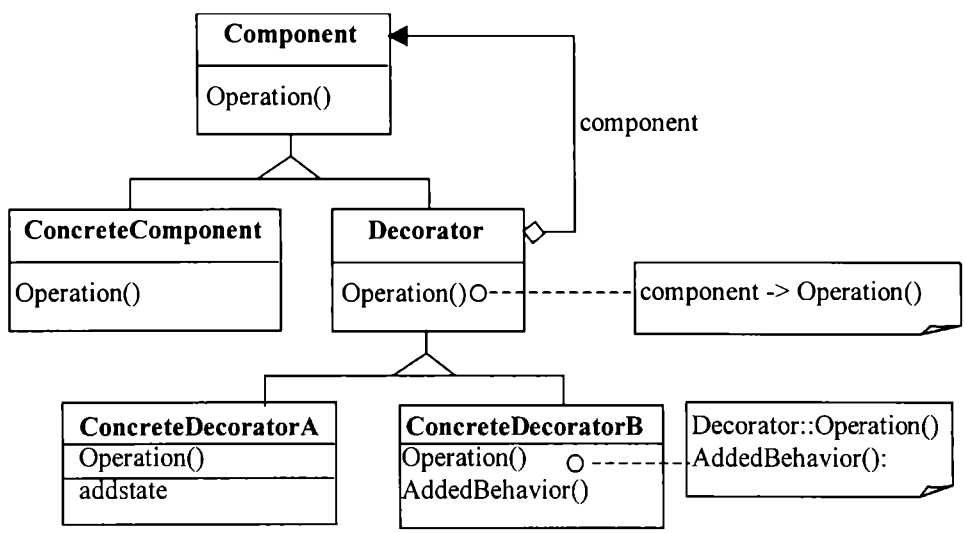


Figura 4.6: Relación entre la Ciudad y su clase decoradora

Estructura:



Aplicabilidad:

- adicionar responsabilidades a objetos individuales en forma dinámica y transparente, sin afectar otros objetos.
- para responsabilidades que deben ser redefinidas.
- cuando extender funcionalidad por subclasificación no es práctico. Algunas veces es posible definir un gran número de extensiones independientes y podría producirse una explosión de clases para soportarlas.

Participantes:

- *Component:* define la interfaz de los objetos a los cuales se les han agregado responsabilidades.
- *ConcreteComponent:* define un objeto al cual se le adicionan rsponsabilidades

- *Decorator:* mantiene una referencia a un objeto *Component* y define la interfaz que conforma la interfaz del objeto decorado.
- *ConcreteDecorator:* adiciona responsabilidades a los objetos

Consecuencias:

- Resulta una solución más flexible que la herencia. Como ejemplo se puede pensar en una aplicación en la que se ha definido la clase *Ciudad.* Para agregarle a esta clase la posición se define una subclase *GeoCiudad.* Pero para ciertas ciudades se necesita manejar el plano. Se define entonces una subclase *GeoCiudadconPlano* en la cual se especifican las características necesarias para definir el plano. Además de generar una arquitectura en la que hay mucha repetición, esto tiene un problema adicional en el caso en que las ciudades tuvieran que ser especializadas por otro criterio, por ejemplo ciudades capitales y ciudades no capitales. La Figura 4.7 muestra como queda la arquitectura planteada.

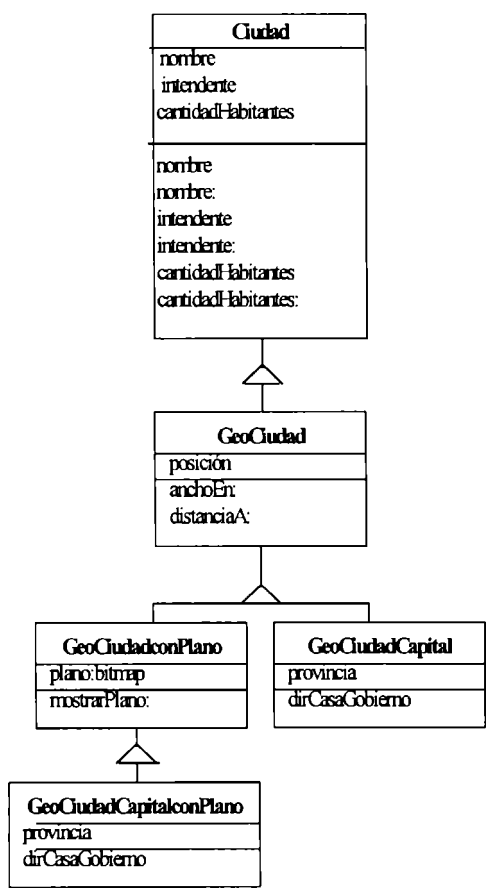


Figura 4.7: Esquema subclasificando por planos y ciudades capitales

Se ve claramente en el ejemplo cómo se repite información para poder definir ciudades con planos y sin ellos y ciudades capitales con planos y sin ellos. Además, la jerarquía resulta bastante poco clara, ya que se está especializando por criterios absolutamente diferentes: la necesidad de contar con los planos de ciertas ciudades y la clasificación por ciudades capitales o no.

Cada subclase define sus propios objetos, por lo tanto, en este caso se van a definir tres objetos ciudades que representen en realidad a la misma ciudad, sólo porque se necesita tener la ciudad conceptual, la ciudad con su posición solamente y la ciudad con la posibilidad de mostrar un plano.

El Decorator, en cambio, replica la interfaz del objeto decorado; y también define comportamiento adicional. Por esto, en lugar de definir diferentes objetos que en realidad representan el mismo con características adicionales, se los decora con distintos comportamientos (por ejemplo espacial) y es posible referirse al objeto original o a cualquiera de los decoradores.

- El uso de Decorators también permite agregar funcionalidad incrementalmente en lugar de definir clases que engloben todas las características, de esta forma, si hay características que no se usan no hay costos adicionales.
- El decorador y sus componentes son idénticas: el decorador actúa como una envoltura transparente para la componente. Pero desde el punto de vista de la identidad, una componente decorada no es idéntica a la componente en sí misma, por lo tanto no se debe asumir esto.
- Cuando se usan decoradores, a menudo se tienen sistemas con una proliferación de objetos pequeños , todos muy semejantes, aunque estos sistemas son fáciles de "customizar" pueden ser dificultosos para realizar controles y seguimientos.

Utilizando la idea de este patrón de diseño es que se construye el modelo geográfico, agregando características espaciales a los objetos en forma dinámica, tanto en aplicaciones que se están comenzando a desarrollar como en aplicaciones convencionales a las que se le quiere agregar comportamiento geográfico.

Como resultado de usar este patrón, las aplicaciones resultantes manipulan dos tipos de objetos, los conceptuales y los geográficos, que definen dos niveles diferentes. Así el modelo resulta muy flexible, ya que podemos definir varios decoradores de un objeto o aún decorar un decorador, es decir anidar especificaciones.

Si lo que estamos haciendo es ampliar aplicaciones ya existentes, el modelo conceptual que se toma como base no sufre modificaciones, es decir, ésta puede seguir siendo utilizada tal como fue concebida.

Cuando a un objeto se le agregan características geográficas, siempre se le debe asociar una posición. Esta posición no solamente tiene información a cerca de las coordenadas terrestres del objeto, sino que también posiblemente, tiene información acerca del tiempo. En particular, aquellos objetos que están en

movimiento poseen una posición que depende del tiempo, por ejemplo, para ubicar una ciudad alcanza con la posición ya que ésta es estática y será la misma en todo momento, pero la posición de un huracán depende esencialmente del momento en que se lo está analizando, entonces la misma debe estar asociada a un período de tiempo determinado. Para evitar confusiones de aquí en adelante, llamaremos Location a la información referente a la posición y al tiempo (cuando sea necesario), mas detalles sobre esta definición pueden ser encontrados en [OpenGIS96].

La definición de la Location de los objetos geográficos es uno de los puntos cruciales en este tipo de aplicaciones y uno de los elementos más difíciles de manejar. Existen varias formas de realizar proyecciones de datos que han sido tomados de la esfera terrestre para llevarlos a un sistema de coordenadas planares [Star et al.90]. El uso de cada una de ellas depende sustancialmente del tipo de datos que se está adquiriendo y de la forma en que éste será manipulado.

Una de las formas mas comunes de representar posiciones es con el sistema latitud/longitud, pero no siempre es el mas conveniente, por ejemplo, este sistema hace dificultoso el cálculo de distancias y áreas.

Otro sistema de referencias usual es el "Universal Transverse Mercator". Este sistema produce una proyección secante de la superficie terrestre y la divide en zonas de 6 grados de longitud por 8 de latitud, proveyendo un mecanismo que permite el cálculo de áreas en una grilla.

Cuando tratamos con las posiciones de los objetos es imprescindible manejar información a cerca de cuál es el sistema de referencias que se esta utilizando. Mas aún, es probable que en una misma aplicación existan objetos referenciados con distintos sistemas, si en algún momento estos objetos necesitan ser combinados, se deben tener disponibles las operaciones de conversión de un sistema de referencia a otro.

Otro elemento que se debe definir asociado a la Location es la geometría, la cual establece la "forma" que tienen los puntos definidos en la Location.

Dada la complejidad de las posiciones de los objetos y todos los elementos que están relacionados, en el modelo se define la Location como un objeto. Este objeto, tiene asociado un sistema de referencias, que es el encargado de interpretar los valores de una posición en un contexto determinado, así como trasformar posiciones de un sistema de referencias en otro; y una geometría que es la que brinda la información a cerca de la topología de los objetos.

La arquitectura resultante en el modelo es la que se muestra en la Figura 4.8

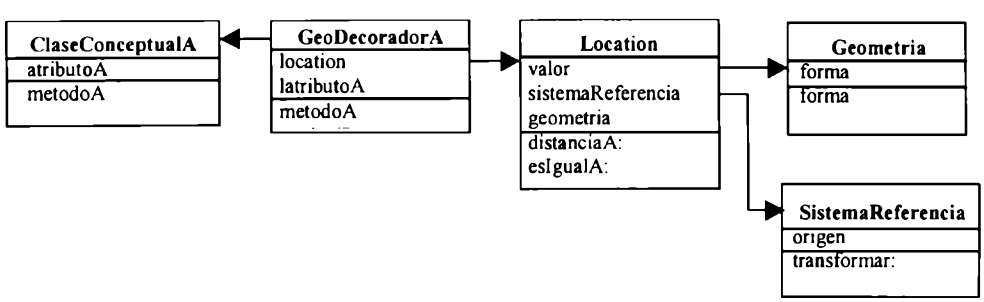


Figura 4.8: el Decorador agrega características espaciales como la Location

En el modelo geográfico pueden aparecer objetos espaciales puros, es decir que no están decorando un objeto conceptual, sino que solo poseen características geográficas. Por ejemplo, supongamos que queremos, en el contexto de una aplicación climatológica, representar el viento como un objeto. La información de interés acerca del viento está dada por su dirección y su velocidad en una posición y en un momento determinado. Está claro que este objeto no va a representar ninguna vista particular en el modelo conceptual.

Para representar este tipo de situaciones se crea una clase abstracta que agrupa el comportamiento tanto de los objetos decorados como de los espaciales puros, el esquema resultante se muestra en la Figura 4.9.

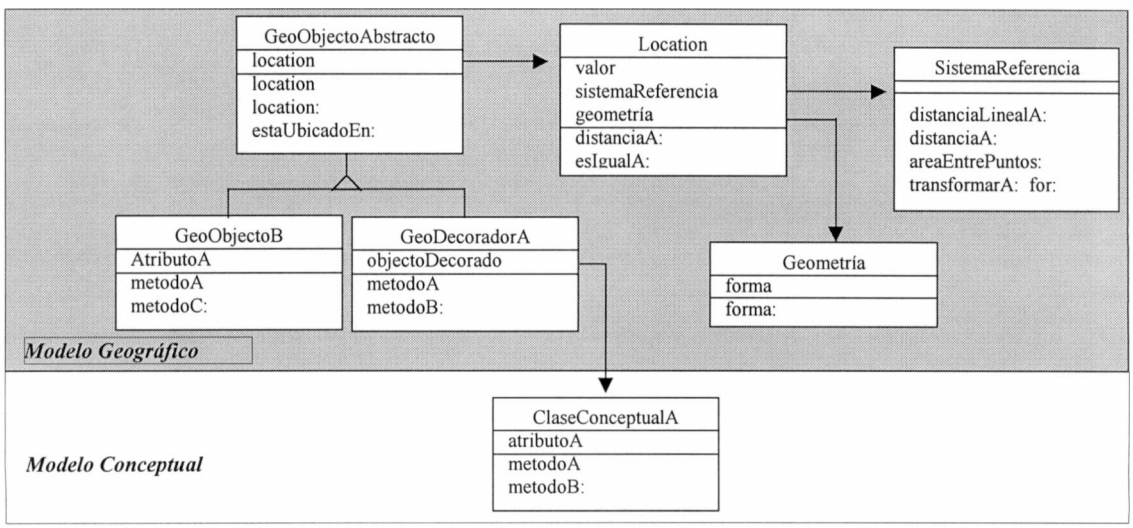


Figura 4.9: esquema básico para construir aplicaciones geográficas

Esta arquitectura es la que se usará de aquí en adelante para definir datos continuos. Es decir, asumiremos siempre que un GeoObjeto tiene una Locetion asociada que define y manipula la posición y, eventualmente la información temporal de ese objeto. La geometría es la que determina todas las características topológicas de los objetos, describirá por ejemplo a qué topología corresponde (punto, línea o polígono) y las operaciones relacionadas. No se incluye en esta tesis una descripción detallada de la definición de la topología ya que no hace a los temas que se tratan.

5- Campos Continuos

5.1 Introducción

Según la Enciclopedia Británica, "el término *campo* se emplea para designar una porción de espacio en la cual se manifiesta cualquier tipo de fenómeno físico". Como ejemplos define el de campo eléctrico el cual se origina a partir de un sistema de cargas eléctricas o el campo gravitatorio que se desarrolla como consecuencia de una distribución de masas. Desde el punto de vista matemático "el campo (también llamado dominio) es una función de las coordenadas de los puntos del espacio y del tiempo o, por analogía, el conjunto de valores que adoptan las variables de un sistema, sean aquellos finitos o infinitos".

Un campo es escalar o vectorial dependiendo del resultado de la función correspondiente. En los primeros la función retorna como resultado un valor, tal es el caso de las temperaturas de una región. En el segundo, se consideran la magnitud y la dirección del campo, por ejemplo, un campo gravitacional es vectorial ya que a cada punto del espacio le corresponde el valor de un vector de campo.

Un campo puede ser estacionario o no, los primeros son funciones que no dependen del tiempo mientras los segundos si, así las funciones que definen campos pueden ser del tipo:

f(x,y) en campos estacionarios o

f(x,y,t) en campos no estacionarios

donde x,y son coordenadas del espacio y t un valor de tiempo.

En el dominio de la información geográfica, estos campos se denominan "Campos Continuos" ("continuous fields") como una manera de reforzar la idea de la continuidad de los campos: como la cantidad de puntos que se manipula no es discreta, la representación de los valores de datos es continua.

Numerosos fenómenos geográficos están relacionados con la noción de campos continuos. En el dominio de la meteorología por ejemplo, datos como las temperaturas en una región, la densidad de lluvias, la humedad; en aplicaciones que conciernen redes de datos como por ejemplo redes eléctricas, de gas, de agua; aplicaciones de cartografía temática: vegetación, tipos de suelo, altitud, porcentaje de erosión, etc. son algunos de los atributos que se manipulan en este tipo de aplicaciones y que requieren el uso de campos continuos.

Para que un Sistema de Información Geográfica pueda manipular este tipo de información, estos fenómenos deben representarse como objetos informáticos y deben poder manipularse. Sin embargo, la implementación real de esta información es generalmente problemática. Muchos de los problemas aparecen porque la relación entre la realidad que está siendo representada por el modelo matemático y el modelo de datos utilizado para representar la información en el SIG no está rigurosamente definida. En particular, mientras que la mayoría de los modelos matemáticos asumen la continuidad de la información, los SIG sólo pueden representar los datos en forma discreta.

5.2 El Problema de la Continuidad

En el contexto de la información espacial almacenada en un SIG, hay dos formas básicas de ver la información: como entidades discretas donde uno de sus atributos es su posición en el espacio (modelo de vector), tal es el caso de un árbol o animales, los cuales por supuesto representan entidades que no son continuas; y la segunda forma es ver a las entidades como distribuidas en el espacio, y cuyas propiedades son funciones de coordenadas del espacio y, eventualmente del tiempo; esta última representa la manera de ver la información continua (modelo de raster).

Dos problemas fundamentales se presentan en este punto, el primero es el soporte y manipulación de información continua; en la actualidad no existen sistemas que manipulen información geográfica y sean capaces de definir a los campos continuos como un tipo de datos mas, con las operaciones necesarias para su manipulación. Habitualmente lo que se hace para poder manejar estos datos es procesar la información fuera del SIG, con programas matemáticos que permitan la interpolación de valores intermedios a partir de una muestra original, y luego incorporarla al SIG para poder realizar las operaciones deseadas. Cada vez que un nuevo cálculo es necesario este proceso debe repetirse.

El segundo problema aparece cuando se quiere combinar el uso de información representada en el modelo vector y en el modelo continuo. Si bien se han realizado varios esfuerzos para la conversión de estos datos, los resultados no han sido los esperados. [Harv69] opina que los modelos de vector y continuo necesitan manipular dos lenguajes diferentes, por ejemplo la noción de similaridad en uno y otro modelo son completamente distintos y produce resultados diferentes cuando se trata de establecer si dos objetos o dos posiciones son similares.

En [Peuquet88] se propone un modelo conceptual dual en donde las entidades puedan ser vistas de dos formas diferentes de acuerdo a su representación.

El dominio de los modelos semánticos puede proveer alguna solución a esta dicotomía. Una de las aproximaciones es considerar un modelo en donde se puedan representar vistas alternativas de la información permitiendo representaciones discretas y continuas alternativamente.

Evidentemente, se necesitan estrategias y técnicas para manipular la información espacial referente a fenómenos continuos. Según [Kemp97] algunas de las estrategias que deberían tenerse en cuenta son:

- Se debe permitir la definición y manipulación de datos provenientes de campos continuos en un lenguaje simbólico. Es decir, se debe poder incorporar en un lenguaje de computación el modelo ambiental que se quiere representar.
- Se debe eliminar la necesidad de considerar la manera de discretizar el espacio, cada vez que sea posible.
- Se debe proveer una sintaxis para incorporar operaciones primitivas apropiadas al modelo ambiental. Esto incluye operaciones para realizar

versiones discretas de campos y la incorporación del concepto de campo vectorial.

- Por último, se debe contar con la posibilidad de combinar representaciones discretas y continuas así como combinar dos representaciones distintas y aún transformar una en otra.

5.3 Manipulación de campos continuos

La matemática de los modelos que se utilizan para manipular datos continuos a menudo está dada en la forma de ecuaciones diferenciales. Esas ecuaciones, implícitamente reconocen la continuidad del espacio y los cambios constantes de las variables. Se han desarrollado numerosos trabajos para definir representaciones discretas del espacio y son actualmente, ampliamente usadas. Los métodos que se utilizan hoy en día en este campo, pueden clasificarse según su tipo en [Pariente94]:

- Métodos globales: tienen en cuenta todos los valores de la muestra a la hora de interpolar un valor. El método de diferencias finitas es un ejemplo de este tipo.
- Métodos locales: sólo tienen en cuenta para la interpolación los puntos mas próximos al punto a estimar. Los métodos que trabajan sobre la triangulación del espacio o las splines son métodos de este tipo.
- Métodos exactos: retornan el valor exacto de la muestra como resultado de la interpolación. La interpolación lineal es un ejemplo.
- Métodos aproximados: retornan un valor distinto del original después de la estimación. Los métodos de regresión son de este tipo.

Por otra parte, los fenómenos continuos estarán descriptos por información discreta, que pueden representarse a través de los modelos numéricos de terreno. Esta información discreta constituye los puntos de la muestra y sirve como base para la construcción de los modelos numéricos.

Existen varias formas de representar la muestra de datos a partir de la cual se realizarán las estimaciones:

Grilla regular:

El área de estudio se divide íntegramente en células del mismo tamaño. El ejemplo típico de la grilla celular es el raster utilizado, por ejemplo, para las imágenes satelitales.

El valor del atributo se representa por un único valor que se asume como el valor válido para toda la región que representa una célula dentro de la grilla. Cada célula se define por su largo y su ancho. La figura 5.1 muestra un ejemplo de esta representación.

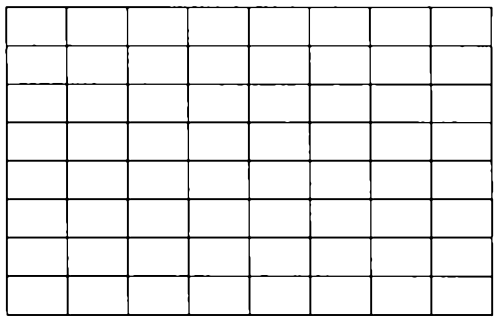


Figura 5.1: grilla regular

Tesellations:

Particiona el espacio en polígonos regulares o irregulares. El valor del fenómeno de estudio representa a toda la superficie representada por dicho polígono. La figura 5.2 muestra un ejemplo de esta representación:

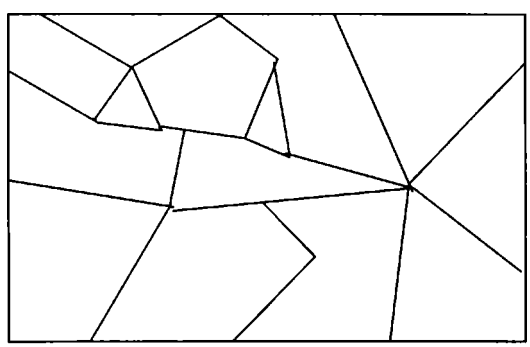


Figura 5.2: Tesellation

Redes de triángulos irregulares:

Esta estructura fue explicada en el capítulo 3. Divide al espacio de estudio en triángulos de diferentes tamaños, el valor asociado a cada triángulo representa el valor del atributo en toda la región especificada por el triángulo.

Grillas de puntos:

La grilla de puntos puede ser regular o irregular. Cada punto contiene el valor del atributo del fenómeno de estudio.

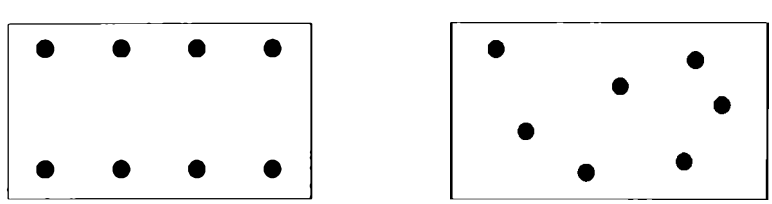


Figura 5.3: Grilla regular e irregular

Curvas de Nivel:

Las curvas de nivel son la intersección de la superficie topográfica con los planos horizontales equidistantes. Cada curva une los puntos de la superficie que tienen el mismo valor del atributo. La figura 5.4 muestra un gráfico de este tipo.

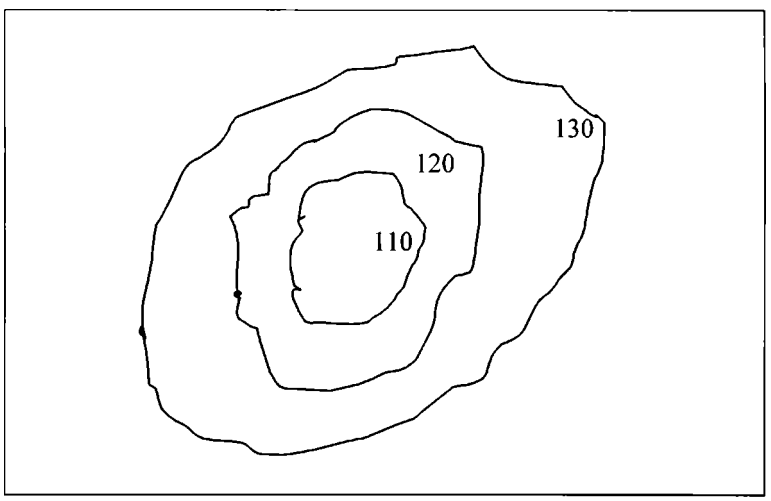


Figura 5.4: curva de nivel

5.4 Operaciones sobre campos continuos:

5.4.1 Operaciones unarias

A continuación se enumeran las operaciones principales que habitualmente se realizan sobre campos continuos. Si bien puede no resultar una lista completa seguro abarca las más importantes.

Las operaciones pueden involucrar un valor particular del campo continuo, todos los valores del campo o combinar valores de más de un campo.

Obviamente, la operación más común es, calcular el valor de un atributo en una posición dada. Para esto se conocen las coordenadas, entonces necesitamos los métodos de estimación para calcular el valor del campo. Una vez que el valor ha sido calculado es posible estimar el gradiente y la integral sobre la zona. A continuación se describen algunas de las operaciones que se realizan sobre los valores de un campo.

Estimación (F, #id_punto)
retorna un valor real que es el valor del atributo especificado, si éste es un escalar. Si el campo es vectorial devuelve un vector.

Gradiente (F, #id_punto)

retorna un vector con tres elementos que representan el gradiente del punto especificado.

Integral (F, #áreaGeográfica)
retorna la integral del dominio especificado por el área

Densidad (F, #_áreaGeográfica)
retorna la densidad del dominio especificado por el área

Cuando el campo es temporal todas las operaciones deben incluir un parámetro asociado con el tiempo, por ejemplo:

Estimación (F, #id_punto, fecha)

Gradiente (F, #id_punto, fecha)

Integral (F, #id-áreaGeográfica, período)

También deben incluirse las operaciones habituales sobre los valores de un campo continuo o sobre un campo continuo propiamente dicho, como por ejemplo *negación*, *valor absoluto*, *logaritmo*, *raíz* y *exponenciación*, las que resultan de aplicación trivial aún cuando la variable represente un campo continuo.

5.4.2 Operaciones binarias

Un de las operaciones más importantes entre dos variables que representan un campo continuo es la comparación. Existen dos conceptos fundamentales que se utilizan para comparar dos discretizaciones espaciales de campos que están definidos con distintas representaciones. Uno es el de *equivalencia espacial* y el otro el de *anidamiento espacial*. Si dos variables de campo son espacialmente equivalentes, entonces la geografía de todos sus elementos se corresponden exacta y completamente. Por ejemplo si A y B son dos grillas de celdas espacialmente equivalentes, entonces, tienen las mismas dimensiones de celdas, el mismo origen, orientación y proyección. El anidamiento espacial indica que una variable se anida espacialmente con otras. Si A se anida espacialmente con B, entonces, cada elemento de A cabe completamente en un elemento de B y el conjunto de líneas que definen los límites de B es un subconjunto del conjunto de líneas que conforman los límites de A. La Figura 5.5 muestra un ejemplo de anidamiento.

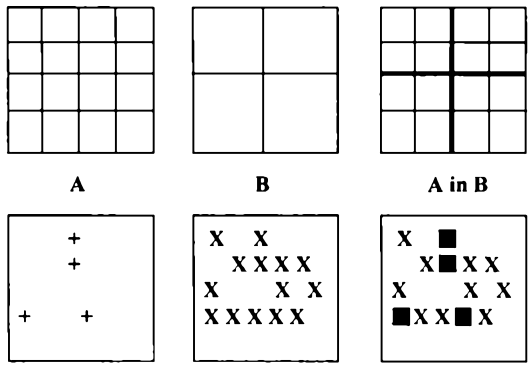


Figura 5.5: Anidamiento espacial

Para realizar operaciones como la suma de campos continuos, se debe tener en cuenta que las computadoras son máquinas discretas y por lo tanto, no tienen la capacidad de generar un nuevo campo continuo en función de dos existentes. Todo campo debe reducirse a un conjunto finito de elementos antes de proceder a realizar cualquier operación de este tipo. Aún es este caso existe una complicación, para poder, por ejemplo sumar el campo A al campo B, se deben sumar los valores que están en la misma posición. Como diferentes campos pueden estar representados de distintas maneras, las posiciones de los elementos también pueden estar expresados en distintos modelos, por lo que se debe antes convertirlos a modelos espacialmente equivalentes para poder operar.

La definición de la operación de asignación genera un problema similar al anterior, al igual que las variables escalares, si B es un campo de temperaturas y A es igual a B, entonces A es una copia del mismo campo. Cada posición de B tiene el mismo valor de esa posición en A. Pero si A y B han sido representados en diferentes modelos, probablemente las posiciones especificadas en A y en B no sean las mismas. En estos casos la asignación requiere la conversión de un modelo en otro para poder compatibilizar los campos. [Kemp97] describe una lista de los procesos necesarios para poder compatibilizar modelos diferentes y poder realizar la asignación.

Cuando las operaciones binarias incluyen un escalar y un campo continuo, el resultado es operar sobre cada una de las variables del campo. Si ambas variables son campos espacialmente equivalentes, entonces se pueden definir operaciones como:

C= A+B
C= A-B
C= A/B
C= A*B

Al ser los campos espacialmente equivalentes las operaciones se realizan sobre cada uno de los valores de los campos.

Si los campos no son espacialmente equivalentes se deben primero realizar conversiones de manera tal que la operación pueda ser resuelta y la respuesta pueda ser asignada. La cuestión es qué tipo de conversión conviene hacer cuando se está

en esta situación, para esto [Kemp97] ha desarrollado un conjunto de reglas a utilizar:

- Si ambas fuentes son espacialmente equivalentes, use ese modelo
- Si cada fuente es un TIN o un modelo de contorno, use el modelo de destino
- Si todas las variables son grillas anidadas, use la grilla mas densa
- Si el destino es espacialmente idéntico a una de las fuentes use la estructura de destino
- Si una de las fuentes es un TIN o un contorno y la otra una grilla, use la grilla
- Si todas las fuentes tienen aproximadamente la misma densidad, use el modelo de destino
- Si una de las fuentes son puntos, use puntos a menos que estén muy dispersos
- Use la estructura mas densa.

De esta forma es posible hacer las conversiones necesarias para poder efectuar las operaciones. Obviamente cualquiera de estos procesos implica costos, lo que hace que hasta la operación de asignación sea complicada cuando se trata de campos continuos. La Figura 5.6 muestra un conjunto de combinaciones de distintos modelos.

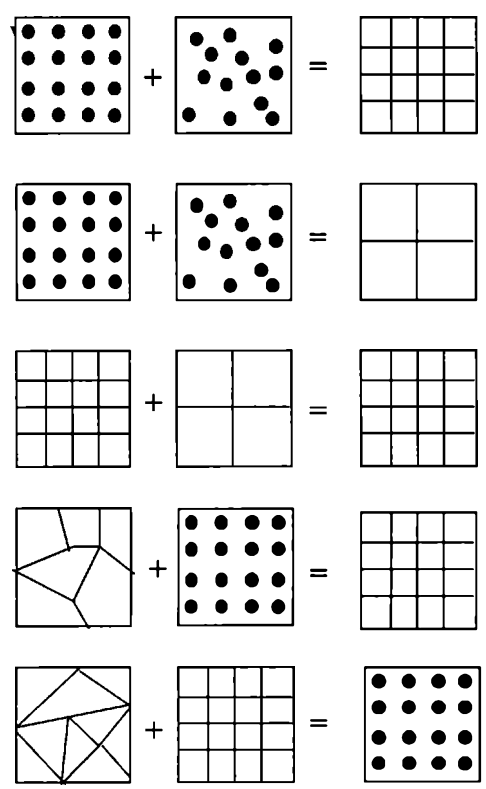


Figura 5.6: Distintas combinaciones de modelos

Una vez que la variable de campo ha sido especificada y los procedimientos de conversión establecidos, es posible implementar las operaciones matemáticas tradicionales fácilmente.

5.5 Requisitos de un modelo Orientado a Campos Continuos

Es necesario entonces definir una estrategia en la cual los diseñadores y los usuarios puedan utilizar un campo continuo como datos que representan fenómenos espacialmente continuos. Así, podrán expresar sus necesidades espaciales como representaciones de la realidad mas que como elementos de la base de datos del SIG.

También es necesario contar con un lenguaje para manipular esta información.

La compatibilidad del modelo de datos y del modelo analítico determina cuán fácil o dificultoso resulta acoplar a ambos. Es importante poder tratar a los campos continuos como un tipo de dato mas dentro del sistema con las operaciones necesarias para manipularlo.

Otra cuestión a tener en cuenta es el uso combinado de información definida como discreta e información definida como continua, de manera que se puedan operar en una forma coherente.

En el capítulo siguiente se describirá una estrategia, basada en el modelo Orientado a Objetos, que permite cumplir con los dos objetivos mencionados.

El modelo orientado a objetos permite especificar tipos definidos por el usuario, por lo que no solamente se puede definir al campo continuo como un tipo de dato, sino también a las representaciones. Esta facilidad permite encapsular las distintas funcionalidades en forma mas modular, por ejemplo todo lo que está relacionado con la conversión de las estructuras utilizadas para la implementación de la muestra estará asociado con las representaciones mas que con el campo continuo en sí mismo.

6- Modelización de Campos Continuos

6.1 Introducción

Un objetivo importante de esta tesis es definir a los campos continuos como un tipo de datos que permita no solamente la manipulación de la información por parte del diseñador, cuya implementación en tipos más primitivos desacoplar la gran cantidad de elementos que se deben tener en cuenta cuando se define información continua.

Los primeros conceptos que se asocian naturalmente a un campo continuo son la información propia del campo mas la muestra que lo define. Para representar un dato continuo entonces, necesitamos una clase en la cual especificar la información propia de una muestra, es decir, datos generales como un nombre, la fecha de la toma de información y los datos específicos como los puntos que la representan, las irregularidades (si las tuviera) las restricciones (si hubiera) [Pariente94] y las operaciones que se realizan sobre una muestra [Kemp97]. La figura 6.1 especifica la clase *CampoContinuo* y muestra las características que estos datos necesitan.

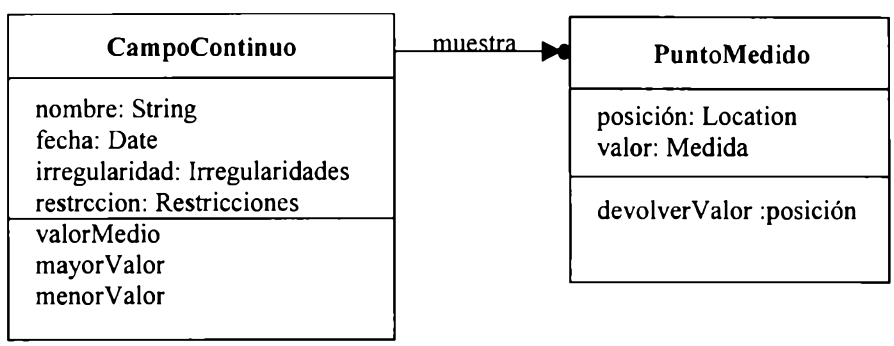


Figura 6.1: CampoContinuo es una clase que asocia los valores de la muestra con sus irregularidades y sus restricciones.

Cada instancia de *PuntoMedido* representa a cada uno de los datos específicos que pertenecen a la muestra y de los cuales se sabe su posición y su valor del atributo de interés (por ejemplo, temperatura, humedad, etc.).

Las definiciones anteriores permiten representar una muestra de un campo continuo, pero tal cual está definida no podemos realizar estimaciones de nuevos puntos a partir de la misma. La clase *CampoContinuo* debería tener asociado un método de estimación para poder realizar este cálculo y así generar valores para posiciones intermedias de la muestra. De hecho, el problema que existe en asociar un método de estimación al dato continuo es que las estimaciones podrían hacerse con diferentes métodos, según el tipo de dato que se está manipulando, la calidad de la muestra, la velocidad de cálculo, la precisión necesaria, etc. Definir el método de estimación dentro de la clase *CampoContinuo* significa fijar un único método posible. Otra solución podría ser subclasificar, de esta forma se debería tener una subclase de *CampoContinuo* por cada método de estimación deseado, tal cual se muestra en la Figura 6.2. Esta solución tiene una desventaja: en el momento que se

instancia un objeto, se debe decidir qué método de estimación se va a utilizar y si se necesitara cambiar el método, la única solución sería borrar el objeto y volver a instanciarlo desde otra subclase.

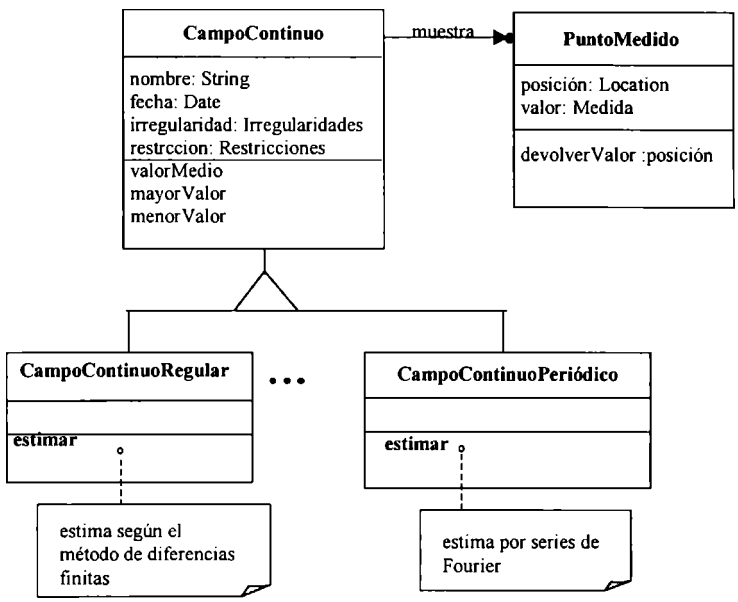


Figura 6.2: los distintos métodos de estimación se especifican en las diferentes subclases.

Existe una mejor solución utilizando el patrón de diseño "Strategy" definido en [Gamma et al.95] y que tiene como objetivo "definir una familia de algoritmos , encapsularlos y hecerlos intercambiables." Este patrón está especialmente definido para cuando diferentes algoritmos son apropiados en distintos momentos, y así, los mismos se pueden intercambiar en forma dinámica; también permite agregar nuevos algoritmos, o eliminar los no utilizados de una manera sencilla. Esta es una estrategia de diseño muy poderosa en este caso particular, ya que se evita que el usuario quede "atado" a un método particular. Utilizando este patrón, el nuevo esquema para la representación se muestra en la Figura 6.3.

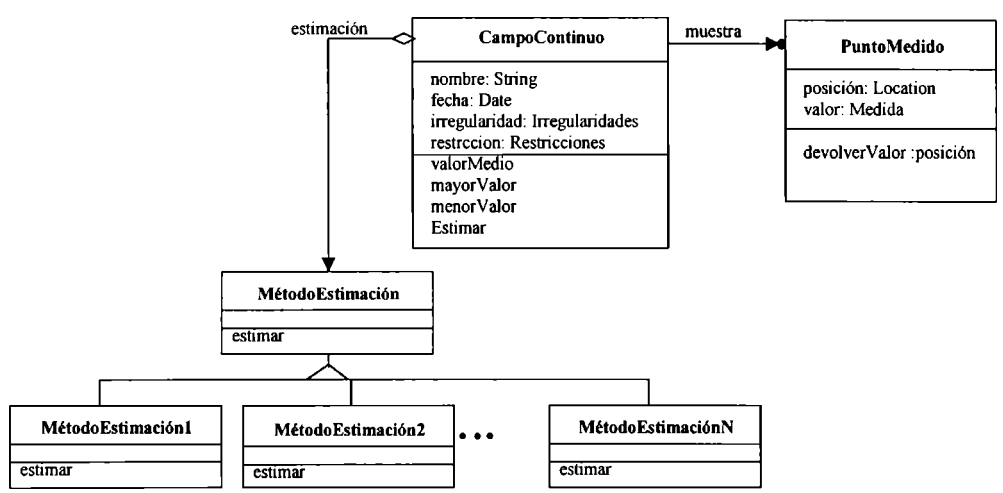


Figura 6.3: El patrón Strategy utilizado para modelar los métodos de estimación

Como vemos en la figura un campo continuo tiene como una de sus partes los posibles métodos de estimación de puntos intermedios. Esta relación es la que permite al objeto cliente crear un objeto del tipo *CampoContinuo* y asociarle un objeto de alguno de los sub-tipos de *MetodoEstimación*. Eventualmente, si es necesario cambiar el método se modifica el valor del atributo estimación en tiempo de ejecución.

Pero aún existen ciertos problemas en la representación del campo continuo; como hemos mencionado existen varias formas de implementarlo: definir la muestra como un conjunto de puntos medidos no nos permite manipular diferentes representaciones. Una posible solución es subclasificar, pero tal cual el caso de los métodos de estimación, la subclasificación ofrece una solución estática al problema, ya que una vez que un objeto se instanció bajo una representación es imposible cambiarla dinámicamente. Una mejor solución es usar el patrón de diseño "Bridge" [Gamma et al.95]. El objetivo de este patrón es "desacoplar la abstracción de su implementación, de manera que ambas puedan evolucionar independientemente". De esta forma se evitan asociaciones permanentes entre un objeto y su implementación, de manera que la última puede ser modificada en tiempo de ejecución. Por otro lado, las modificaciones en la implementación no tienen consecuencia sobre los clientes, e incluso es posible compartir implementaciones entre diferentes tipos de objetos.

En la Figura 6.4 se muestra el esquema utilizando el patrón Bridge para modelar las implementaciones de la muestra.

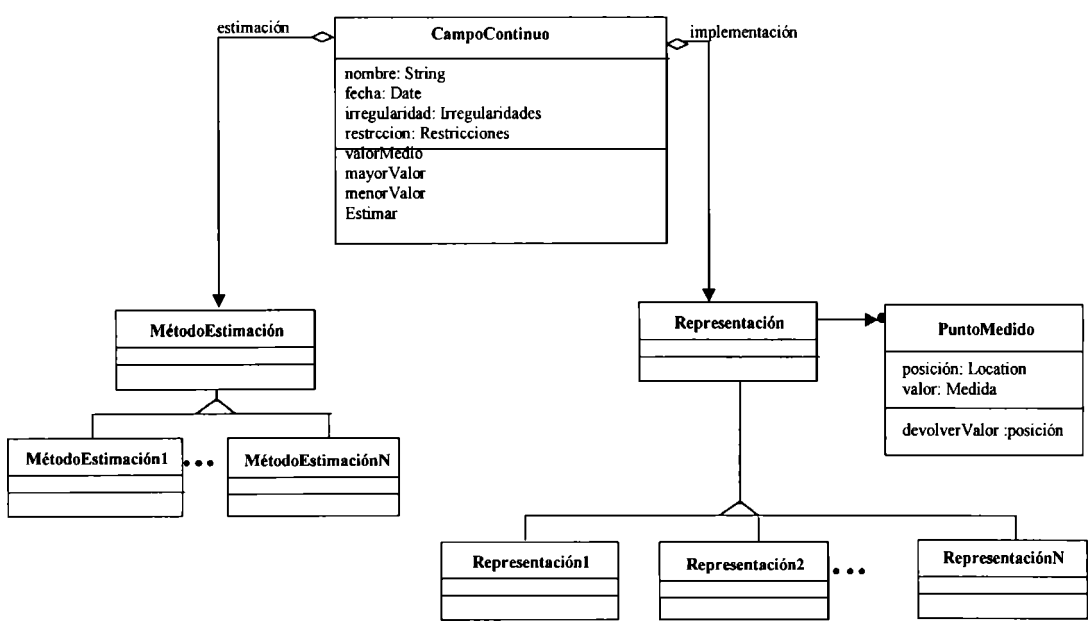


Figura 6.4: La aplicación del patrón Bridge permite múltiples implementaciones

Como se puede ver en la figura anterior el atributo que representaba los puntos de la muestra fue eliminado y, en su lugar, se define una relación entre el campo continuo y las diferentes representaciones de la muestra, las que a su vez son quienes "conocen" los puntos que conforman cada muestra, ya que las

representaciones serán las que en definitiva recuperarán y manipularán la información.

6.2 Definiendo campos continuos

Existen aplicaciones en las que se trabaja con objetos geográficos (referenciados espacialmente) que poseen como una de sus características un dato que pertenece a un campo continuo. Por ejemplo, una aplicación que trabaja con las temperaturas en distintas ciudades de una provincia, algunas de ellas medidas en una muestra y otras que se determinarán con algún proceso de estimación. Los "GeoObjetos" en este caso tendrán información específica (en el caso de las ciudades por ejemplo nombre, población, etc.), una posición que determina la ubicación geográfica de cada objeto y, además uno o más datos continuos asociados (en el caso del ejemplo, la temperatura de la ciudad). Podemos definir a los GeoObjetos como se muestra en la Figura 6.5.

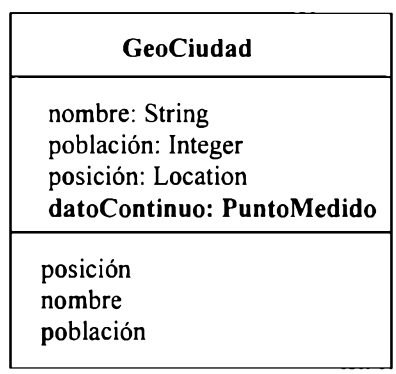


Figura 6.5: El atributo datoContinuo de la clase define el valor de la temperatura

A través de un atributo se le asocia un valor de un campo continuo, como por ejemplo la temperatura a una ciudad.

Un objeto del tipo PuntoMedido es un punto perteneciente a un dato continuo. Cuando se instancia un objeto existen dos posibilidades: el valor del dato continuo (la temperatura en el caso de la ciudad) es uno de los puntos de la muestra, o bien no existe un punto medido para la posición correspondiente a la del objeto que se está creando y es necesario estimar el punto correspondiente para poder establecer la asociación. Es necesario entonces, para poder implementar este procedimiento que al instanciar un atributo que representa un valor de un campo continuo se realice el análisis antes descripto. Para ello, es necesario que el cliente que genera la creación de un objeto que contiene un atributo que corresponde a un valor de un campo continuo, cuente con los métodos necesarios. La Figura 6.6 muestra el esquema que permite realizar el proceso descripto.

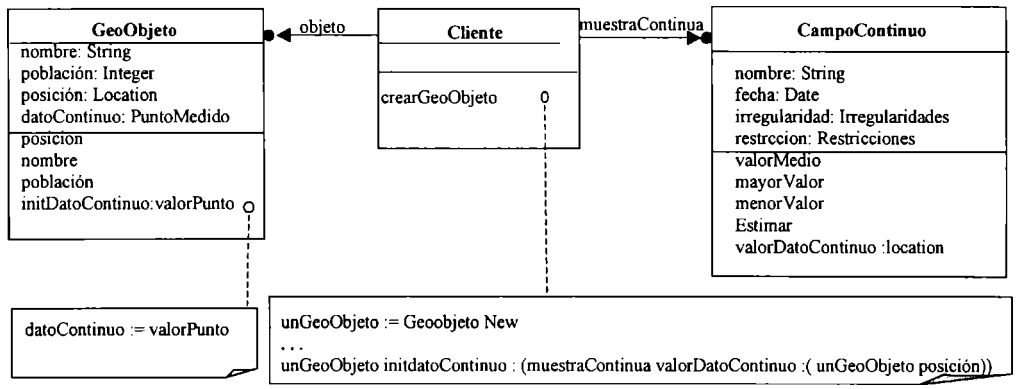


Figura 6.6: El cliente y sus relaciones con las ciudades y el campo continuo.

De esta manera, cada objeto tendrá asignado el valor correspondiente ya sea porque la misma ha sido medida o porque se ha estimado a partir de la muestra. Sin embargo, existe todavía cierta redundancia en el esquema presentado que puede evitarse. Por una parte, cada objeto, tiene una posición que lo ubica geográficamente; por otro lado cada punto medido también tiene una posición dentro de la muestra, por lo tanto es redundante tener las dos posiciones cuando ambos objetos se asocian. La única información que el geo-objeto necesita es el valor del atributo del campo continuo. Podemos desacoplar entonces la información que corresponde a la posición de los valores propiamente dichos. De esta forma, el objeto geográfico conocerá solo el valor y la unidad en la que este valor fue medido. El nuevo esquema se muestra en la Figura 6.7, donde por razones de claridad no se ha representado la jerarquía correspondiente a los métodos de estimación.

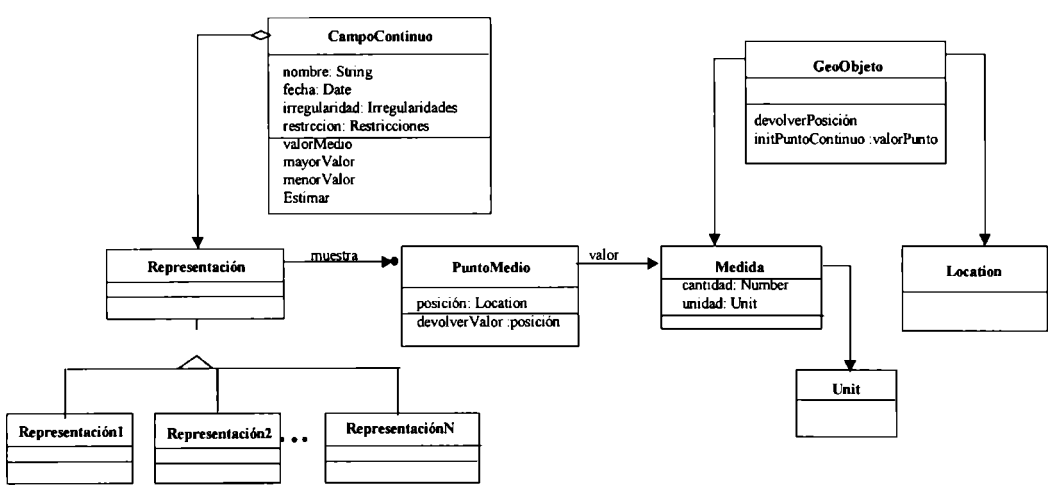


Figura 6.7: El dato continuo del objeto es de tipo Medida.

Con este esquema se pueden crear objetos y asignarles los valores que correspondan, valores que (como ya se ha mencionado), o pertenecen a una muestra o se estiman con el fin de poder manipularlos. Como se puede observar en la figura 6.7 se especifican los tipos *Medida* y *Unidad*; este patrón de diseño, está definido en [Fowler97] y permite especificar las cantidades de una medida en distintas

unidades. El uso de las medidas con sus unidades asociadas utilizando como base este patrón, puede ser ampliado de manera de convertir unidades para hacer comparaciones entre medidas que se han tomado usando como base distintas unidades o utilizar medidas complejas. La especificación de esta definición ampliada se obvia en este texto dado que no es el centro del tema que se está atacando.

Este esquema permite recuperar el valor de un dato continuo asociado a un objeto geográfico para, eventualmente, hacer algún tipo de análisis o simplemente conocerlo. Por otro lado, a partir de la definición de un campo continuo se pueden realizar operaciones que conciernen a todos los datos de la muestra, como por ejemplo determinar valores promedio de toda la región censada o, también, actualizar la muestra cuando sea necesario, operación que, en aplicaciones donde los datos varían continuamente, como es el caso de las temperaturas, es crítico para poder realizar un análisis válido.

Sin embargo, hay todavía una discusión que se debe tener en cuenta. La decisión de si los puntos que se estiman se guardan junto con la muestra o se mantienen como puntos fuera de la muestra original cambia ciertos aspectos en la representación. En muchas ocasiones, agregar a la muestra original los puntos estimados, no es una buena solución ya que en este caso se pierde la posibilidad de realizar nuevas estimaciones, eventualmente con otros métodos para después poder comparar resultados; lo mismo ocurre cuando los datos de la muestra original deben ser modificados constantemente (como en el caso de las temperaturas) y éstos se toman en puntos fijos; en este caso habría que especificar cuáles son los puntos a tener en cuenta en una nueva estimación. En función de lo anterior la solución es tener los puntos estimados, pero no como parte de la muestra original. De esta manera, el campo continuo, no solamente tiene asociada la muestra(a través de la representación) sino que también "conoce" los puntos estimados. La Figura 6.8 muestra la arquitectura teniendo en cuenta esta última discusión.

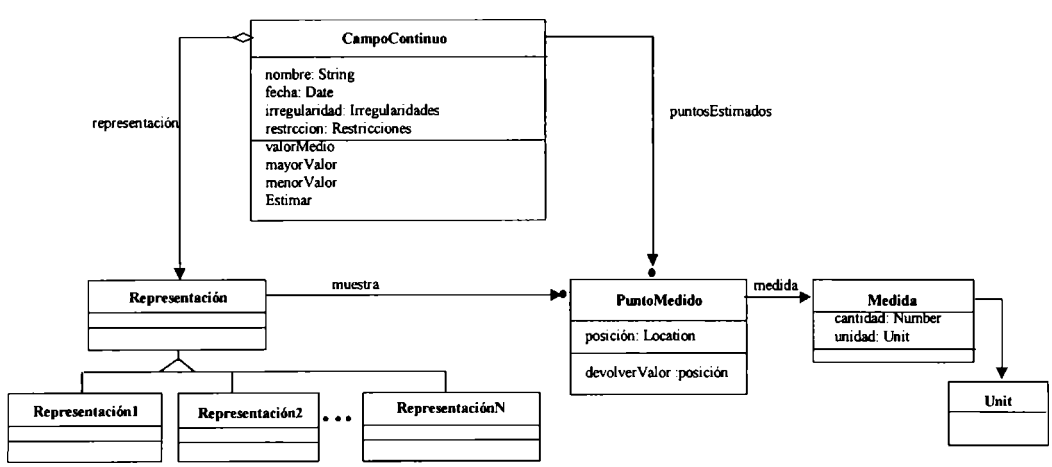


Figura 6.8: la relación entre el campo continuo y uno de sus valores

De esta manera, la arquitectura definida permite diseñar y manipular un dato continuo, asociar a objetos geográficos datos continuos pertenecientes a una muestra determinada y realizar operaciones con los datos individuales y con la muestra en

general. También es posible modificar los valores de la muestra produciendo la actualización automática de los valores de los atributos continuos de un objeto geográfico. Esto es posible pues estos atributos no tienen asociados un valor fijo, sino el objeto medida que es mismo objeto que manipula el dato continuo.

6.3 Representando atributos continuos en objetos geográficos

Es esta sección se mostrará cómo utilizar la arquitectura definida para manipular atributos continuos en los objetos geográficos. Esto significa que los geoObjetos definidos en el modelo geográfico de una aplicación pueden definir atributos que toman sus valores de aquellos definidos en un campo continuo.

Por ejemplo, supongamos que se quiere desarrollar una aplicación para trabajar con las temperaturas de las distintas ciudades de una provincia. Para ello se cuenta con las temperaturas exactas tomadas en ciertas ciudades donde hay observatorios climatológicos.

El primer paso es modelar las ciudades y luego agregarles las características geográficas. La figura 6.9 muestra las clases necesarias para esta definición.

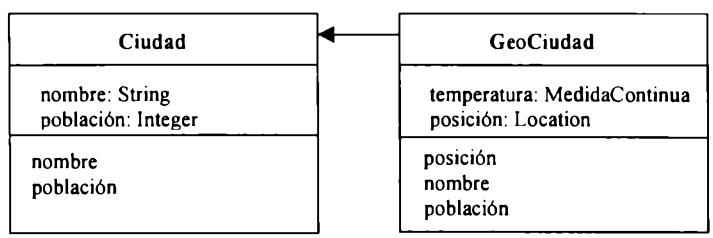


Figura 6.9: la clase GeoCiudad tiene un datos continuo

GeoCiudad define un atributo temperatura que tendrá el valor de las temperaturas en las distintas ciudades. A su vez, la temperatura de cada una de la ciudades es uno de los datos que conforman la muestra de una región completa y en función de los cuales se estimarán las temperaturas en aquellos lugares en donde el dato no fue medido.

La muestra, que es parte del campo continuo de las temperaturas tendrá entonces valores medidos en aquellas posiciones que coinciden con las de las ciudades. Sin embargo podría darse el caso de que las Locations de las ciudades estén definidas como polígonos y no como puntos, mientras que las posiciones en el campo continuo son puntos. En esta situación deben compatibilizarse las posiciones, trabajando, por ejemplo, con el centroide del polígono de las ciudades. El centroide se obtiene a partir de las operaciones definidas en las clases que describen la topología de los objetos definidos en el modelo de vector, específicamente de la topología de polígonos [Gordillo et al.98]. De esta manera, es posible compatibilizar ambas posiciones sin necesidad de conversiones.

La figura 6.10[a] muestra el esquema resultante para la representación de las ciudades con su temperatura.

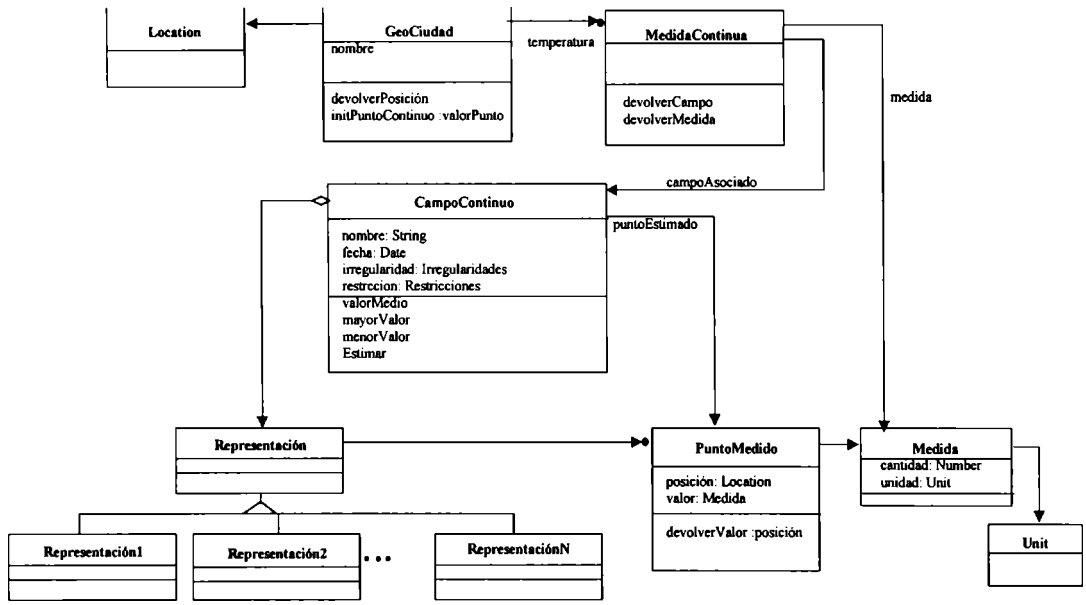


Figura 6.10[a]: esquema de una aplicación de ciudades y temperaturas

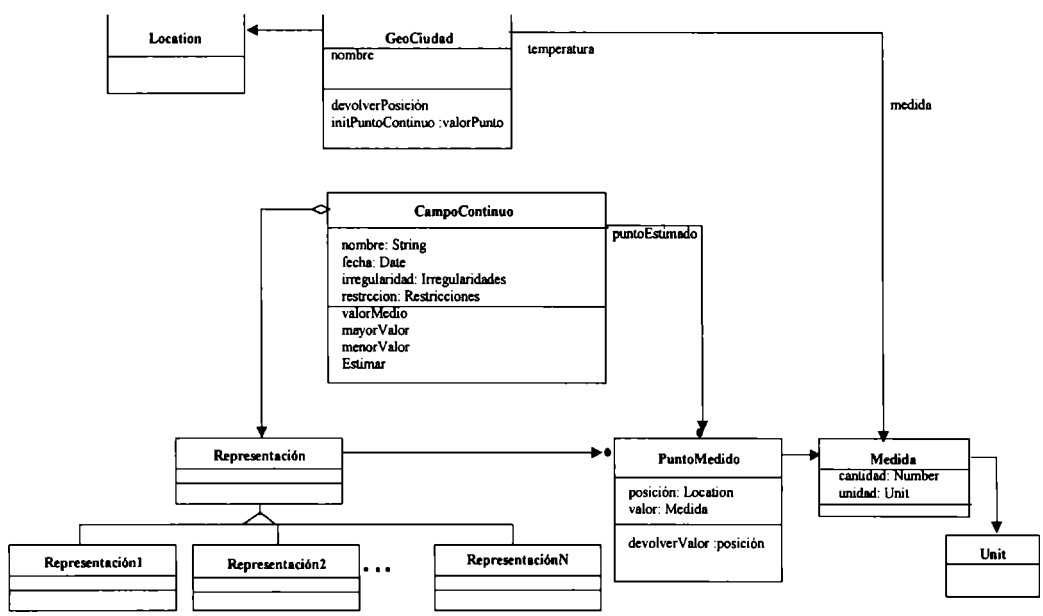


Figura 6.10[b]: Las Geociudades sólo conocen el valor de la temperatura

La clase *MedidaContinua* es la que asocia el valor del atributo (por ejemplo, *temperatura)* y el campo continuo del cual se está obteniendo con el objeto geográfico correspondiente. La necesidad de que las ciudades "conozcan" el campo continuo del cual están obteniendo la temperatura, está directamente relacionada con las características de la aplicación. Si la única operación que se va a disparar

desde las ciudades es la de conocer sus temperaturas, mientras que las actualizaciones y/o modificaciones del campo están pensadas para ser realizadas desde un cliente que manipule todo lo referente al campo, la asociación entre las temperaturas de las ciudades se podría obviar, generando un esquema como el que muestra la Figura 6.10[b].

De esta forma, cada ciudad tiene asociado el valor de la temperatura y del campo continuo en caso de ser necesario. En la figura 6.11 se muestra el código de los métodos correspondientes a la instanciación del atributo *temperatura* para la clase *Geociudad.*

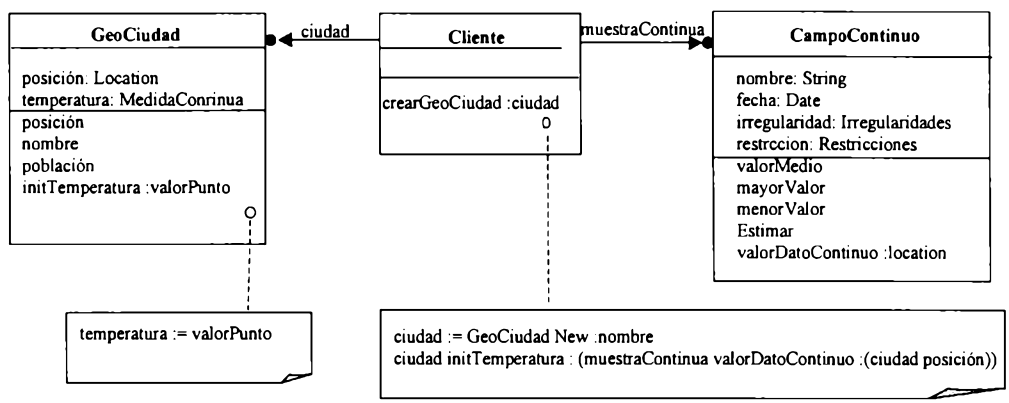


Figura 6.11: los métodos para crear y manipular un dato continuo

De esta forma cuando el cliente crea una ciudad con un nombre determinado, el método que es responsable de la creación interactúa con el campo continuo para solicitarle el valor de la temperatura en esa posición. El objeto *CampoContinuo* a su vez determinará si el valor correspondiente a esa posición está en la muestra, en cuyo caso lo recupera y lo devuelve. Si no existe en la muestra lo estima y luego lo retorna. Así, se asegura que cada ciudad tiene un valor de la temperatura asociado.

6.4 Definiendo Campos Continuos como Objetos

En el ejemplo anterior mostramos un caso en el que objetos discretos toman información de los valores que almacena un campo continuo. Aquí, el campo continuo, tal cual está mostrado, solo sirve como un proveedor/almacenador de valores ubicados en posiciones determinadas.

Sin embargo, hay veces que los objetos que se modelan representan en sí un campo continuo. Supongamos por ejemplo que estamos representando una aplicación en la que queremos estudiar fenómenos como las crecidas de los ríos en una región. Algunos de los objetos que se modelizarán serán los descriptos en la Figura 6.12.

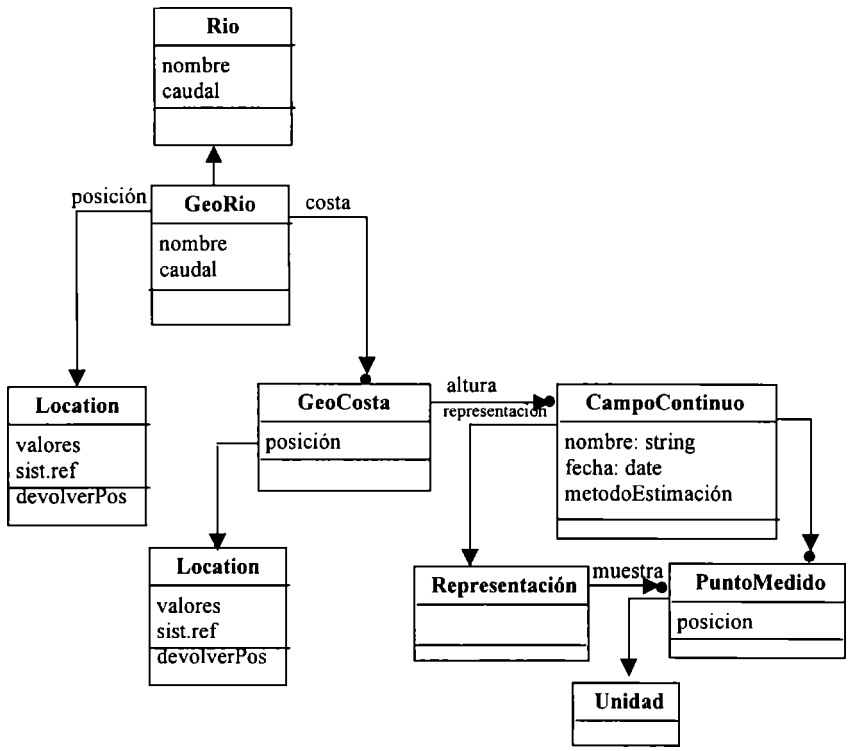


Figura 6.12: Representación de la altura de la costa como un campo continuo

En la clase *Costa* de la figura 6.12, se definen diferentes características (atributos) asociadas a campos continuos. La posición de la costa definirá un conjunto de puntos que coincidirán con los puntos de la muestra, a partir de los cuales el campo continuo estimará el resto de los valores. Es importante notar que esto no implica que las posiciones estarán repetidas en el sistema, sino, simplemente que el campo continuo compartirá las posiciones de la muestra con la costa.

Esta representación desacopla las características de la costa del río permitiendo definir, si fuera necesario, varios campos continuos relacionados a la costa. El sentido de este desacoplamiento es evitar que la clase *Río* tenga demasiadas responsabilidades y se convierta en una clase monolítica y, por lo tanto, difícil de extender o modificar. Obviamente, este desacoplamiento puede evitarse definiendo las características directamente en la clase *GeoRio*. En este caso se omite la definición de la clase *Costa*. La figura 6.13 muestra el esquema resultante.

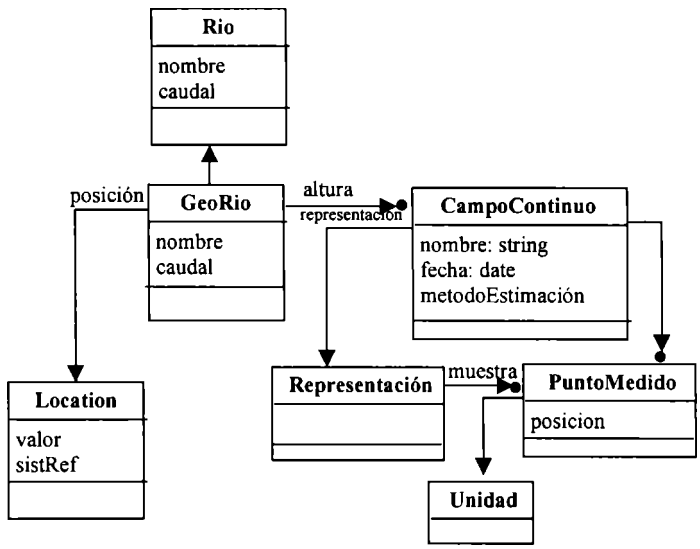


Figura 6.13: El campo continuo asociado al río

En la figura 6.13, el río en sí mismo es tratado como un dato continuo y no se propone una separación entre el río y la costa. Ahora, son las posiciones del río las que representan los puntos de la muestra, y así, la costa no es tratada como un objeto.

Para operar con datos continuos es necesario tener en cuenta lo discutido en el capítulo 5. Si bien los problemas que se plantean respecto de la ejecución de las operaciones continua existiendo, también es claro que la modularidad del esquema presentado, mas las características del paradigma orientado a objetos permiten focalizar cada una de las situaciones que se pueden presentar y buscar soluciones específicas en cada caso.

La clase *CampoContinuo* tendrá definida todas las operaciones que conciernen a este tipo de datos, como por ejemplo aquellas mencionadas en el capítulo 5. Cada operación binaria deberá, antes de la ejecución, determinar la equivalencia espacial en función de la información de los campos a operar; si ésta no se diera, se deberá aplicar algún método de transformación. En función de las representaciones de los operandos deberá determinar cuál será la representación del resultado. Ambos problemas pueden resolverse asociando clases o jerarquías de clases que definan los métodos necesarios para hacer las transformaciones.

Para ejecutar una operación entre dos campos continuos, la clase cliente contendrá los mensajes necesarios para recuperar el resultado de las mismas. La figura 6.14 muestra un ejemplo de una de las operaciones binarias que se definieron en el capítulo anterior.

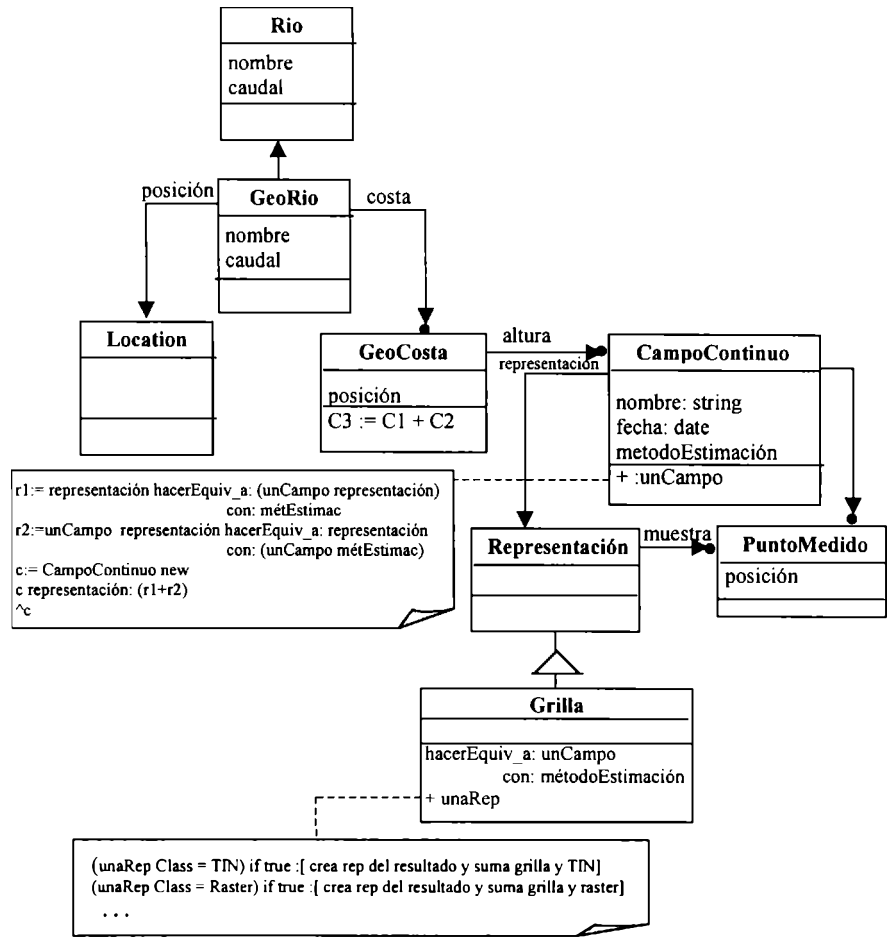


Figura 6.14: operaciones en un campo continuo

De esta forma los campos continuos definen sus propias operaciones permitiendo que sean utilizados como un tipo de dato más en el sistema. La jerarquía de representaciones debe estar proveer los elementos necesarios para las transformaciones de manera de poder compatibilizar diferentes implementaciones de una muestra, así como obtener la equivalencia espacial de dos campos continuos. Esto implica, no solamente definir el mismo espacio para ambos campos, sino también calcular los mismos puntos. Para ello, se calculan dos campos temporarios (r1 y r2 en la figura 6.14) que no solamente conocen las representaciones de ambos campos, sino también los métodos de estimación utilizados por cada uno. La invocación doble al método *hacerEquiv_a* tiene como objetivo obtener, por estimación, los puntos medidos que la otra posee. Luego, se crea un nuevo campo continuo y se le asigna la representación correspondiente a la suma de los dos campos. Para determinar cuál es la representación apropiada, se puede utilizar la clasificación de [Kemp97]. El método + de la clase *Grilla* debe encargarse de determinar la representación del campo argumento y de acuerdo a ella creará la nueva representación apropiada. Cada subclase de representación deberá definir el método *hacerEquiv_a* y el método + de manera de compatibilizar los campos.

La arquitectura definida permite definir y manipular campos continuos, y las operaciones que se puedan realizar con ellos. Si bien esta arquitectura no está

enteramente implementada, ya que no se han implementado los métodos matemáticos que implican tanto la estimación de los puntos, como la transformación de dos campos en espacialmente equivalentes y muchas de las operaciones definidas, la misma provee una serie de ventajas que permiten una aproximación a una modelización completa y eficiente de esta información. La primera tiene que ver con la posibilidad de modelar situaciones intrínsecamente complejas, y que está directamente relacionado con el uso del paradigma de objetos. La segunda es la posibilidad que provee la arquitectura definida de desacoplar los problemas que plantea el uso de campos continuos y que permite tratar a cada uno en forma individual. Estas dos posibilidades definitivamente brindan la posibilidad de pensar en situaciones en el modelo con las que no se ha trabajado hasta ahora, por ejemplo manipular diferentes representaciones de campos, distintos métodos de estimación e incluir las operaciones entre los campos continuos en las clases que los definen.

7- Implementación de la Arquitectura

7.1 Introducción

Si bien no es el objetivo central de esta tesis llegar a una implementación del modelo de campos continuos, presentamos en este capítulo una análisis de las posibles alternativas de implementación en ambientes orientados a objetos. Para acotar la discusión al sub-problema de los campos continuos, obviamos analizar la implementación de todo el modelo, aunque las conclusiones que se presentan aquí pueden extenderse fácilmente a los otros aspectos (sistemas de referencias, decoración de objetos conceptuales, etc.).

La arquitectura definida para los campos continuos se puede pensar como un sistema de clases, donde la clase *CampoContinuo* describe a cualquier tipo de campo, mientras que los objetos representan muestras con distinto atributos (por ejemplo el objeto temperatura, el objeto presión etc.).

Así, una implementación inicial podría consistir en definir una biblioteca de clases que implementen la funcionalidad expresada en esta tesis. Sin embargo, desde el punto de vista del manejo de una base de datos esta aproximación presenta una desventaja respecto del proceso que implica la recuperación de los objetos y probablemente su indexación, ya que no hay distinción entre distinta información representada como un campo continuo. Esto genera espacios de búsqueda más grandes y, por lo tanto, cualquier operación que involucre la recuperación de dos objetos que representan la misma información (como por ejemplo temperatura) o la indexación de la base, será más costosa. La generación de información de salida que involucre el mismo tipo de atributos también será más cara; por ejemplo si queremos construir curvas de nivel de alturas de las regiones, o la construcción de layers con distinta información (por ejemplo tipos de suelos con temperaturas).

Una segunda opción es subclasificar la clase *CampoContinuo* de manera que cada tipo específico de campo aparezca como una subclase; esto es podría ahaber una subclase *Temperatura*, otra *Presión*, etc.

Sin embargo esta última alternativa agrega una dificultad adicional para un usuario de la arquitectura, ya que es imposible suponer que podremos proveer una jerarquía de clases que abarque todos los posibles tipos de datos continuos y, en caso de no contemplarse, la aprición de cada tipo nuevo genera la necesidad de modificar el esquema de la aplicación para definirlo; al mismo tiempo es cierto que configurar una aplicación que requiere de datos continuos puede resultar difícil ya que deben customizarse aspectos tales como implementación de la muestra, métodos de estimación, etc.; además (y como se explicó en el capítulo 6) la arquitectura resultante puede resultar compleja.

Otro aspecto a tener en cuenta durante la implementación es cuál es la mejor opción para trabajar los datos estimados. Básicamente existen dos posibilidades: los datos estimados se calculan y se utilizan, pero no se almacenan; o, por el contrario, la estimación implica el almacenamiento de los resultados, los que no deberían estar dentro de la muestra original.

El primer enfoque, implica un costo de cálculo ya que cada vez que se necesita un dato que no está en la muestra se debe estimar; el segundo requiere, no solamente costo de almacenamiento (recordemos que la cantidad de datos estimados puede ser grande), sino también un esfuerzo considerable en el mantenimiento, ya que podría ocurrir que se utilicen distintos métodos de estimación para comparar los resultados y luego seleccionarse alguno, o podría ser necesario re-estimar una muestra, en cuyo caso tendríamos igualmente una sobrecarga. En este caso, es cierto también que disponer de los distintos tipos de campos continuos explícitamente puede facilitar el proceso.

Proponemos en este trabajo que la funcionalidad de datos continuos sea implementada como un framework orientado a objetos (más precisamente como un sub-framework del soporte completo de nuestro modelo).

7.2 Hacia un framework Orientado a Objetos para datos continuos

Un Framework "*es una aplicación reusable, semi-completa que puede ser especializada para producir una aplicación adaptada a una necesidad particular*" [Fayad97], [Johnson97]. En un framework se especifican usualmente aquellos aspectos de un dominio de aplicación que permanecen constantes en todas las aplicaciones dentro de ese dominio y se provee básicamente la posibilidad de expresar las variaciones ya sea mediante la sub-clasificación de las clases del framework para adaptarlo a un sub-dominio nunca considerado, o mediante la composición de los objetos de la aplicación con los del framework para "instanciarlo" para una aplicación en particular. Un framework posee tanto la definición de las clases que lo componen como el comportamiento abstracto que permite que objetos de dichas clases colaboren entre sí y con los objetos de la aplicación. Ejemplos de frameworks orientados a objetos pueden encontrarse en [ACM97].

Un framework orientado a objetos para datos continuos consistirá de las clases abstractas definidas en el capítulo 6, más algunas sub-clases para tipos de campos continuos más comunes, implementaciones y métodos de estimación estandar. Una aplicación que utiliza campos continuos instanciará las clases correspondientes indicando (mediante mensajes apropiados) como deben configurarse de manera de llegar a una red de objetos como en la figura 6.7. La funcionalidad de administración de campos continuos, entonces, se trasladará de los objetos clientes a los objetos instanciados del framework quienes controlarán aspectos tales como pertenencia a una muestra, estimación de nuevos valores, persistencia, etc. El framework podrá ser extendido mediante el agregado de nuevos métodos de estimación, tipos de muestra o nuevos tipos de campos continuos.

Un paso adicional imprescindible para este tipo de sistemas será construir herramientas de instanciación visual que simplifiquen la customización del framework para una aplicación concreta, permitiendo que el diseñador pueda escoger los aspectos particulares de su aplicación minimizándose el código. En el capítulo 9 comentamos algunos detalles respecto a la implementación del framework para datos continuos.

8- Trabajos Relacionados

8.1-Estado del Arte en GIS orientados a objetos

Desde hace alguno años, se están haciendo varios esfuerzos para definir un modelo adecuado para construir aplicaciones geográficas basadas en el modelo orientado a objetos [Triffonaet al.95] [Kosters95] [Oliveira et al.97] y [Pariente94]. A continuación se describen algunos de estos trabajos y se discuten las diferencias existentes entre ellos y la aproximación descripta en esta tesis.

8.1.1 El modelo UAPE [Oliveira97]

En este trabajo los autores describen un ambiente para la modelización y el diseño de aplicaciones geográficas (UAPE) , especialmente pensado para usuarios que son expertos en el dominio pero no necesariamente en el área de sistemas de información y, por lo tanto, no están en condiciones de explotar las herramientas que provee un SIG. El ambiente ha sido diseñado de manera que actúe como una capa que pueda ser acoplada a un SIG, pero que a su vez es independiente de un producto particular. Una de las mayores ventajas de este ambiente es que le permite a los usuarios abstraerse de los detalles de implementación y tratar solamente con la vista conceptual de la realidad geográfica.

UAPÉ provee facilidades al usuario para el diseño de la aplicación y la modelización de datos geográficos basándose en la combinación de un modelo de datos semántico GMOD [Camara et al.94] con una metodología de diseño de aplicaciones. Soporta básicamente dos tipos de actividades: el diseño de las aplicaciones geográficas y las bases de datos, y la manipulación de los datos almacenados en la base. Las facilidades de diseño guían al usuario a través de una serie de pasos que garantizan la documentación de las decisiones de diseño y el reuso e integración de datos existentes. Estos pasos están dados por una metodología de diseño de aplicaciones ambientales que ha sido desarrollada en la Universidad de Campinas (Brasil) y validada por diversos usuarios.

UAPÉ soporta dos tipos de interacción:

- En el modo de modelización, los usuarios pueden diseñar y documentar la planificación de estrategias, entonces son guiados por la metodología que está soportada por un modelo geográfico orientado a objetos llamado GMOD. Luego este modelo es mapeado a la base de datos subyacente.
- En el modo de salida, los usuarios pueden utilizar una interfaz gráfica para navegar a través de los datos geográficos y el esquema.

El modelo de datos GMOD utiliza todos los conceptos básicos del modelo orientado a objetos: objetos, clases, herencia, composición, etc. y es la base de la comunicación entre el usuario y UAPÉ. Es una extensión del modelo presentado en

[Camara et al.94] que permite además la modelización de fenómenos temporales y la descripción de relaciones entre las entidades.

Presenta cuatro niveles de abstracción: el del mundo real que comprende los elementos de la realidad geográfica que serán representados. El nivel conceptual que permite modelar los elementos en un nivel muy alto de abstracción. El nivel de representación que introduce detalles de la representación geométrica y topológica de las propiedades espaciales de los elementos geográficos. Por último el nivel de implementación que corresponde a las estructuras internas y a la implementación de las operaciones.

Desde el punto de vista del usuario, UAPÉ puede ser visto como un conjunto de módulos integrados (ver Figura 8.1). Pueden explorar la base de datos subyacente a través de la interfaz, además ésta le provee facilidades para modelizar y visualizar actividades en UAPÉ. El módulo de Modelización y Diseño les permite definir la base de datos en un alto nivel, así también como el comportamiento de las entidades. El módulo de la Metodología lo asiste en las actividades de modelización, registra conocimiento sobre la metodología utilizada y guía al usuario en los siguientes pasos.

El módulo de Recuperación y Manipulación se usa para recuperar y manipular datos dentro de UAPÉ. Finalmente, el módulo Geo-Base es el que se encarga de abstraer la base de datos subyacente proveyendo transparencia a los usuarios.

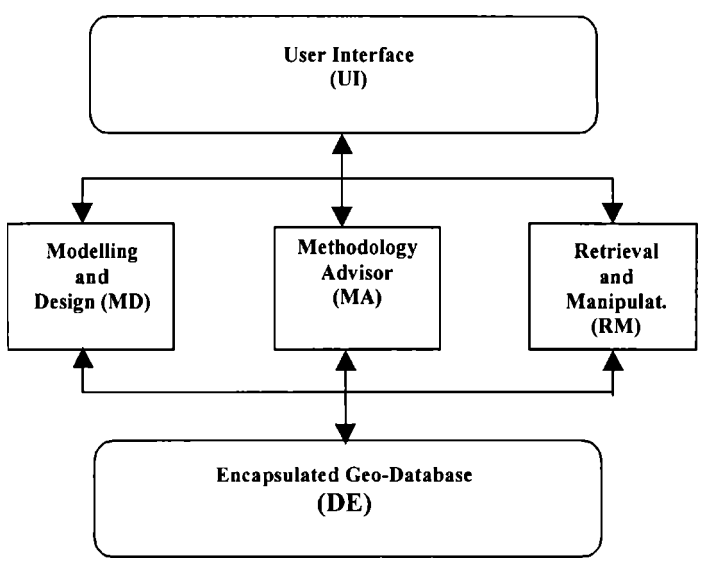


Figura 8.1: Organización Conceptual

La estructura de clases propuesta para el modelo conceptual es la que se describe en la Figura 8.2. Hay dos clases básicas *GeoClass* y *Conventional*, las cuales especifican respectivamente, los objetos que poseen características espaciales y los que no. Esta distinción permite a las aplicaciones compartir clases no espaciales y ayuda en el diseño y reuso.

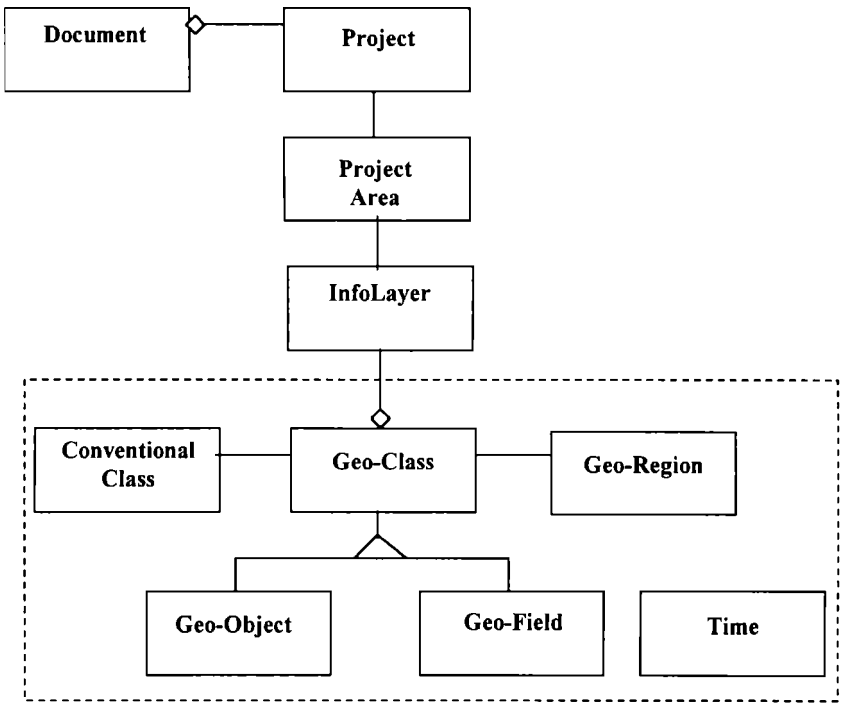


Figura 8.2: El modelo de datos GMOD

En el nivel de representación, el modelo introduce otras clases que pertencen a dos jerarquías con raiz en las clases *RepObject* y *RepField.* La primera permite representaciones usando puntos, líneas y polígonos y la segunda incluye subclases como *Grid, Trinetr, Contour, PlanarSubdivision.*

8.1.2 El modelo GeoIFO [Trifona et al.95]

Los autores definen un modelo espacial para aplicaciones geográficas que integra la visión orientada a campos y a objetos. La intención es presentar un modelo que no emplee metáforas computacionales (como por ejemplo, registros, procesos, etc.), que resulte comprensible para profesionales que no pertenecen al área de la informática y que ademas sea formal y completo de manera que, sin ambigüedades pueda ser convertido a un modelo de datos lógico.

Proponen aumentar modelos semánticos (como Entidades/Relaciones, IFO [Abiteboul et al.87] y SDM [Hull et al.87] de manera de soportar los conceptos necesarios de la información espacial y mantener la filosofía particular de cada modelo. Los principales aportes del trabajo son la integración del modelo de campos y el de objetos y mostrar cómo unos pocos conceptos (básicamente el concepto de *posición* y el de atributos *variables en el espacio*) permiten la modelización espacial y calzan perfectamente en estos modelos.

Los autores definen dos nuevos constructores, la *agregación espacial* y el *agrupamiento espacial* y con ellos manejan conocimiento geográfico como por ejemplo: propiedades que varían con el tiempo, límites indefinidos, relaciones topológicas y generalizaciones.

Definen la posición como la especificación del centroide, la forma, el tamaño y la orientación, y está definida sobre todos los objetos geográficos. Todas las operaciones geográficas (longitud, área, distancia) y las relaciones topológicas están predefinidas en el dominio de la posición.

Los *atributos variables en el espacio* se utilizan para definir información espacial que no pertenece a un objeto particular sino que son propiedades que se definen en el espacio, por ejemplo, la altura del terreno en una aplicación catastral, que si bien podría verse como una característica de cada una de las parcelas, está claro que podría definirse aunque las parcelas no existieran y además si una parcela se moviera y cambiara su posición debería heredar el valor de la altura según la posición a la que ha sido movida. Informalmente, definen atributos de este tipo, como propiedades del espacio que devienen en propiedades de objetos cuando éstos se sitúan en ese espacio, es decir el valor de estos atributos depende únicamente de la posición y no del objeto en sí mismo. Formalmente, es una función cuyo dominio es el espacio.

El modelo **IFO** es un modelo semántico estándar cuyo esquema es un grafo dirigido con varios tipos de vértices y aristas. Los vértices representan tipos (clases) que pueden ser atómicos o complejos. Los tipos atómicos pueden ser no imprimibles (rombos) o imprimibles (rectángulos), los tipos atómicos que representan una relación Es-Un se llaman tipos atómicos libres (círculos). El concepto de agregación (círculo con una X) define una composición de objetos y el agrupamiento (círculo con un *) denota la colección de elementos homogéneos de un tipo existente.

GeoIFO extiende el modelo anterior con un tipo especial de objetos, *Posición* que representa la posición de un objeto en el espacio y que se define a través de un atributo is-located-at y la función *SPA* que modeliza los atributos que varían según el espacio, es decir SPA: SPACE→DA. La Figura 8.3 muestra este esquema.

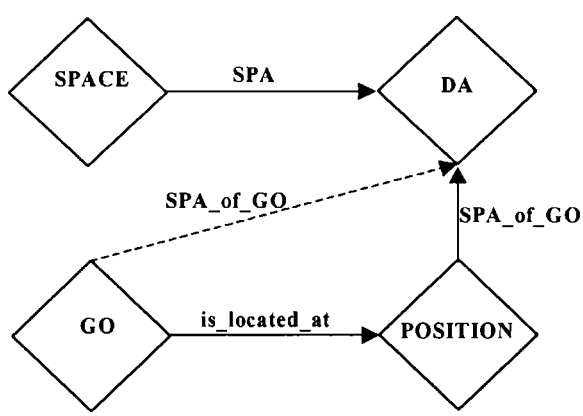


Figura 8.3: Modelización de atributos variables en el espacio

La Figura 8.4 muestra una instanciación del esquema anterior donde el tipo GO son montañas y se quiere conocer el tipo de suelos.

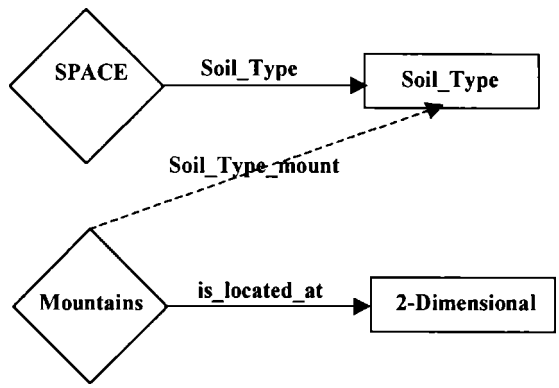


Figura 8.4: modelización de la posición de un objeto y su atributo variable

De esta manera si se quiere saber cual es el tipo de suelo del monte Olimpo, será el tipo de suelo restringido a la posición de ese monte (Soil_Type_mount).

La implementación de este modelo ha sido realizada en Kappa [IntelliCorp93], que es un lenguaje de cuarta generación con un shell para sistemas expertos y que provee un ambiente de desarrollo orientado a objetos. Para las aplicaciones geográficas tienen una clase *Geometría* con subclases 0-Dimensión, 1-Dimensión, 2Dimensiones y GeoNulo que representa el conjunto vacío. Las posiciones también se representan en una jerarquía cuya raíz es *Position* y que incluye subclases para manipular 1-Dimensión, 2-Dimensiones etc.

Los atributos variables en el espacio se definen como atributos multivaluados en Kappa que toman valores en tuplas de la forma (posición, valor), de esta manera los atributos pueden tener diferentes valores en el área de un objeto.

8.1.3 El modelo GeoAA [Kosters et al.95]

Los autores resaltan la importancia de contar con herramientas que permitan hacer una buena ingeniería de requerimientos para poder desarrollar software en gran escala en general, y en particular de aplicaciones geográficas.

Sostienen que los métodos de análisis orientados a objetos como el de Coad/Jourdon (OOA) no son capaces de reflejar loa requerimientos de este último tipo de aplicaciones apropiadamente y proponen un modelo GeoOOA que lo complementa agregando geo-primitivas y permitiendo un tratamiento mas adecuado para las mismas.

Las primitivas de GeoOOA soportan abstracciones de clases espaciales, estructuras Parte_de topológicas y estructuras de redes. No consideran aspectos tales como información espacio-temporal y atributos gráficos.

GeoOOA distingue entre tipos de clases convencionales, que representan objetos que no tienen características espaciales; y tipos de geo-clases que están definidas a través de las clases *Punto*, *Línea*, *Región* y *Raster* que proveen las operaciones que corresponden. También define restricciones topológicas, como por ejemplo si los objetos de una clase se pueden intersectar o no. También definen un conjunto de relaciones Parte_de que se muestran en la Figura 8.5 para modelizar relaciones topológicas entre el objeto compuesto y sus partes. Estas relaciones se

dividen en *cubrimiento*, *contención* y *partición,* cuya característica común es que la geometría de cada parte tiene una intersección no vacía con su contenedor.

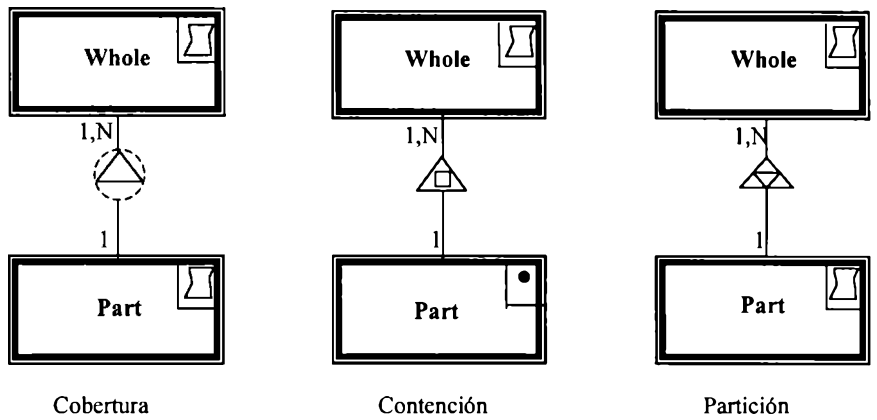


Figura 8.5: Topología de las estructuras Parte_de

En una cobertura el contenedor y sus clases pertenecen a la misma geo-clase y la geometría del contenedor es cubierta por la geometría de las partes.

En una estructura de contención la geometría del contenedor contiene a la geometría de las partes. Mientras que la partición es una estructura de contención con dos restricciones adicionales, el contenedor y sus partes pertenecen a la misma geo-clase y la geometría de las partes forma una partición de la geometría del contenedor.

Estas estructuras proveen los servicios básicos para el manejo espacial tales como conectar y desconectar objetos, consultar por la vecindad entre objetos sobre el borde de la partición.

También definen una serie de extensiones para el manejo de redes. Una de las cuestiones que se plantean en el trabajo es la necesidad de conocer algunos datos a cerca de la red que habitualmente no están dados en forma directa. Por ejemplo, cuales clases de nodos están conectadas por un link o cómo ciertos links de una clase *Link* específica inciden en ciertos nodos. Las primitivas para una estructura de red en GeoOOA pueden verse como un patrón genérico para tres roles que se presentan habitualmente en la semántica de un grafo, el rol nodo, el rol link y el rol red. Una red cosiste de una clase *Network*, al menos una clase *Node* y al menos una clase *Link*. Una geo-clase puede aparecer en mas de una red y puede jugar roles diferentes. La clase *Network* provee servicios tales como testeo de conectividad o ciclos.

La estructura de la red puede verse en la Figura 8.6 y consiste, como ya se ha mencionado, de una clase *Network* que está relacionada con todas las clases de nodos y de links que aparecen en la aplicación.

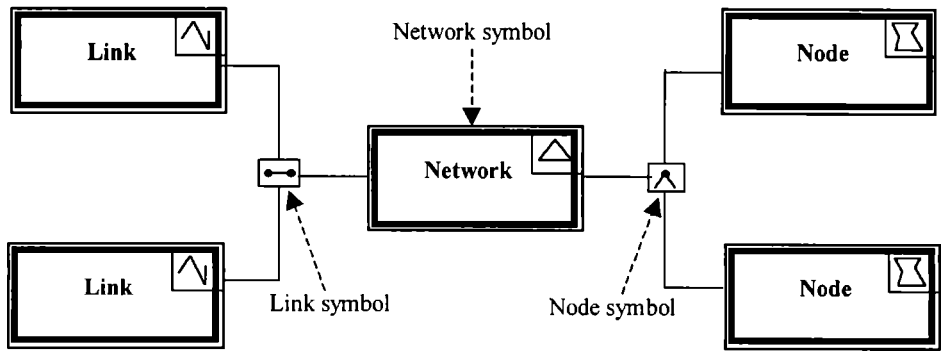


Figura 8.6: Estructura de una red en GeoOOA

Los símbolos de link y de nodo son los que especifican el rol de las clases asociadas. Para abstaer las características comunes de los nodos y los links se le asocia una clase abstracta *Node* y *Link* respectivamente a cada símbolo.

8.1.4 El método Pyrenees [Pariente94]

En su tesis doctoral, Dillon Pariente realiza un estudio sobre los campos continuos. Explica la semántica de los mismos, define la especificación de restricciones sobre los campos, critica los métodos de interpolación y extrapolación espacial existentes y, en función de estas restricciones, define un nuevo método de estimación que permite la especificación de restricciones y describe una modelización orientada a objetos que soporta el método a partir de la definición de un tipo abstracto *Campo Continuo*. Finalmente, especifica un lenguaje de declaración y manipulación de campos continuos.

El autor propone definir la estimación a partir del método de Diferencias Finitas que permite la estimación de campos continuos preservando la integridad de los puntos de la muestra y asegurando un máximo de regularidad de los valores. Este método propone la discretización del tiempo y del espacio a través de la resolución de la ecuación de Laplace. Sin embargo, este método presenta algunos problemas que deben ser aligerados o eliminados; por ejemplo, la provisión de los valores en el borde del área de estudio (condiciones de bordes) y el tiempo de cálculo; además es conveniente adaptar el método de manera que permita la introducción de restricciones estadísticas, la especificación de irregularidades, la implementación de una nueva estructura lógica para el cálculo del sistema de ecuaciones que genera.

La forma en que resuelve el último punto, es decir el cálculo del sistema de ecuaciones es a través del uso de una red de autómatas celulares y utiliza en particular las redes neuronales de Hopfield.

Sobre esta base y con una definición completa del método de resolución construye un modelo orientado a objetos para su implementación. La arquitectura de este modelo se muestra en la Figura 8.7. En el esquema modeliza tres elementos fundamentales, la muestra, las restricciones y las irregularidades.

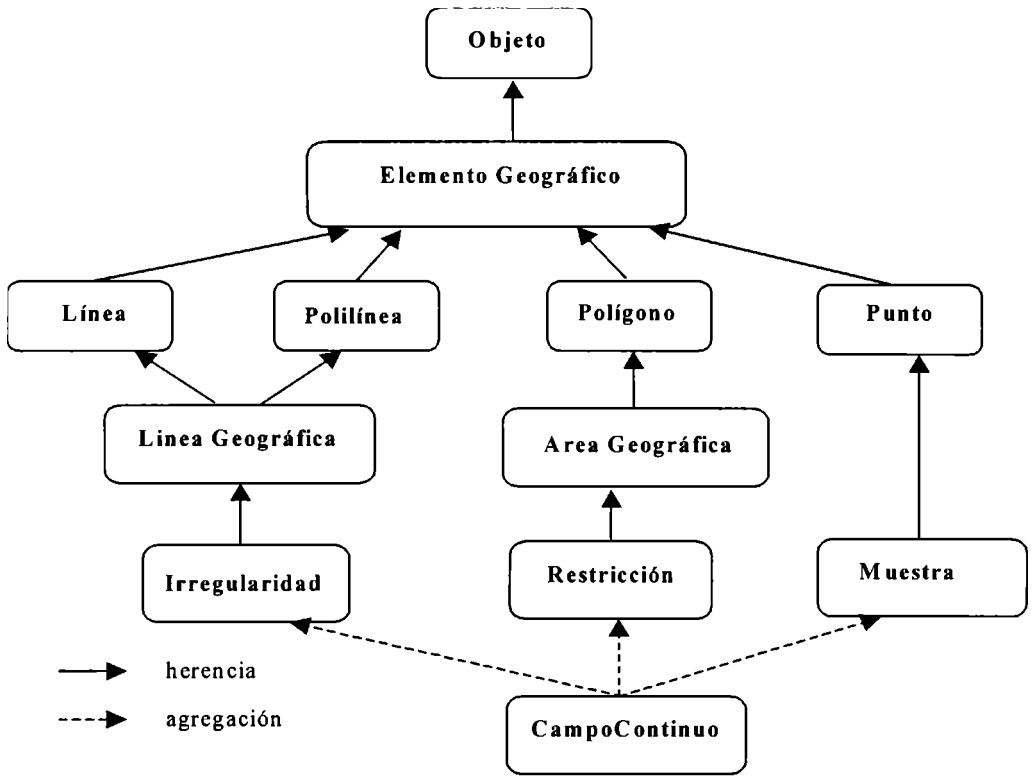


Figura 8.7: El modelo Orientado a objetos para representar campos continuos

La muestra está formada por los puntos de la muestra, estos objetos contienen las coordenadas espaciales y temporales, así como el valor del atributo. Las restricciones describen la información de la que disponemos dentro del campo de estudio. Las irregularidades, finalmente dan información a cerca de los límites en los cuales la estimación es mas precisa.

La clase *Campo Continuo* se define por:

- El nombre y la naturaleza del campo estudiado.
- La resolución a partir de la cual se determina el número de neuronas necesarias para representar el espacio de estudio.
- El error que define la diferencia tolerable para la activación de las neuronas, define la condición de fin del proceso.
- El intervalo en X y en Y que define el área de estudio.
- Un intervalo de fechas que determinan el intervalo de tiempo en el cual la muestra, las irregularidades y las restricciones son válidas.
- Los puntos de la muestra, las irregularidades y las restricciones.
- Un método que explica cual es la influencia de otros parámetros en el fenómeno (y representa la función correspondiente al miembro derecho de la ecuación de Poisson).

La muestra, por otro lado se define como una tupla que tiene la identificación del punto, un vector con los valores de los atributos y la fecha de la muestra.

La arquitectura general del sistema es la que se muestra en la Figura 8.8, en la que se pueden ver los módulos que lo componen y cómo interactúan.

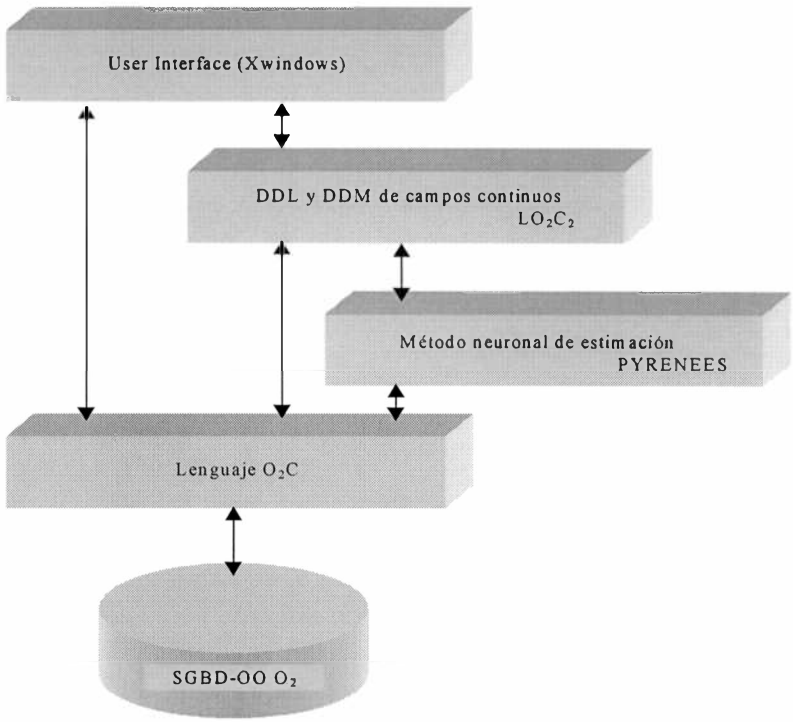


Figura 8.8: Arquitectura del sistema LO_2C_2 y Pyrenees

El usuario puede utilizar directamente O_2C y manipular los datos gerenciados por O_2. Si escribe o manipula datos continuos, o va a requerir de ellos, accede al lenguaje LO_2C_2, que se encarga de traducir y manipular los campos en lenguaje O_2C. Si se trata del requerimiento de un dato que ya ha sido estimado por el método PYRENEES recupera el dato de la base de datos existente, si no ha sido estimado, le requiere al módulo correspondiente la estimación del dato, enviando un mensaje con los parámetros necesarios para hacer la estimación. A su vez el módulo encargado de estimar (PYRENEES) accede a la base de datos para recuperar la información necesaria (puntos de la muestra, restricciones, etc.) usando O_2C, luego interpola y extrapola retornando el resultado al módulo LO_2C_2; además guarda el resultado de la estimación en una base de datos O_2.

9- Aportes del modelo. Factibilidad de su utilización

9.1 Aportes del modelo

Todos los modelos descriptos se basan en la tecnología de objetos para la definición de datos geográficos.A continuación se definen los aportes que el modelo descripto en los capítulos anteriores presenta respecto de los mismos.

Por un lado la introducción del paradigma de objetos en los SIG tiene un impacto enorme debido a la posibilidad de encapsular operaciones y datos complejos (como los que existen en estos sistemas) en objetos, en vez de separarlos en estructuras de datos y almacenamiento diferentes como es usual en los SIG "convencionales". Al mismo tiempo el paradigma de objetos soporta un mayor grado de interoperabilidad entre aplicaciones tal cual se discute en [OpenGIS96]; este aspecto es crítico debido a la necesidad de integrar este tipo de sistemas con otros de tipo "legacy" y que involucran aspectos no-geográficos.

Estas ventaja, compartidas con los otros enfoques discutidos en el capítulo anterior es potenciada por la utilización sistemática de patrones de diseño y por la definición de "micro-arquitecturas" que enfatizan el desacoplamiento y el reuso.

Una de las principales contribuciones esta dada por la separación de las características conceptuales y las geográficas inducida por nuestro modelo. La especificación de éstas últimas a través del uso de los decoradores, permite agregar características espaciales a las clases de una aplicación existente sin que ésta deba ser modificada. Esta facilidad redunda positivamente, no solamente en los costos de la ampliación de la aplicación sino también en la posibilidad de que la ya existente siga siendo utilizada sin sufrir cambios ni reales ni para los usuarios. Este aspecto además ataca fuertemente el problema de la interoperabilidad ya sea con sistemas "legacy" o con otros SIG debido a que los decoradores representan una buena estrategia para encapsular otros sistemas (ver por ejemplo [Balaguer et al 97b]); al mismo tiempo al separarse claramente los aspectos geográficos de los conceptuales se potencia positivamente el funcionamiento del grupo de desarrollo, debido a que pueden diferenciarse los roles de los expertos en el dominio geográfico los cuales pueden interactuar con otros diseñadores mediante contratos precisos (los establecidos por los geoObjetos decoradores).

Si bien el trabajo presentado en [Medeiros94] propone relaciones entre clases conceptuales y geo-clases que permiten definir la funcionalidad espacial, las relaciones especificadas requieren que las clases conceptuales tengan conocimiento de las geo-clases agregandose un nivel mayor de acoplamiento.

El desacoplamiento de las posiciones de los objetos, del sistema de referencia y la geometría también producen ventajas desde el punto de vista de la implementación, ya que hace posible la asignación dinámica de esta información a los objetos geográficos. Por ejemplo, la posibilidad de reasignar posiciones o sistemas de referencias es posible pues la arquitectura definida desacopla las posiciones de los sistemas referencia en los cuales deben interpretarse. La modelización separada del sistema de referencia y no como una características mas de los objetos hace posible manipular conversiones entre distintos sistemas para

realizar operaciones entre objetos que han sido referenciados con diferentes métodos.

Otro aspecto interesante de esta arquitectura es que permite el desacoplamiento también a nivel de representación; esto significa la posibilidad de definir la geometría de los objetos independientemente de sus otras características. En la mayoría de los modelos, en el nivel de representación la geometría de los objetos se define como clases que representan puntos, líneas y polígonos y los objetos se definen como subclases de éstas. La subclasificación, como ya fue explicado, genera la imposibilidad de modificar las características de los objetos en forma dinámica (en este caso la geometría). En nuestro modelo es posible variar dinámicamente la geometría de un objetos geográfico mediante el cambio de valor de una relación.

En el caso particular de los campos continuos, el modelo permite como característica básica, la definición de distintas estructuras de representación y la implementación de distintos métodos de estimación. Por otra parte, permite que los objetos de la aplicación utilicen datos de los campos continuos como características propias. Esto facilita no solamente el estudio de una región como un campo, sino también la asociación de objetos concretos a datos relacionados con un campo continuo.

En este sentido, se define al campo continuo como un tipo abstracto de datos, cuestión que ya ha sido discutida y recomendada por varios autores [Kemp97]; sin embargo la diferencia respecto a arquitecturas específicas, como la de [Pariente94] es que en su caso (tal como queda reflejado en la arquitectura de la figura 8.7) la implementación de los objetos (puntos, líneas, polilíneas, polígonos) no es independiente de los objetos en sí mismos y una vez que uno de ellos fue instanciado, su representación no puede ser modificada a menos que el objeto se destruya y se vuelva a crear. Además en [Pariente94] el campo continuo se modela como un tipo abstracto que no se conecta con objetos concretos dentro de la aplicación por lo cual la integración de los dos tipos de datos en un SIG (modelo de objetos y de raster) difícilmente se pueda obtener.

Otro aspecto digno de destacar respecto a nuestro modelo, tiene que ver con la definición de operaciones entre campos continuos. No realizaremos aquí un análisis detallado por razones de espacio y de complejidad del tema, que involucra desde analizar las diferentes representaciones y los métodos para hacer compatibles a las muestras mediante métodos de equivalencia espacial. Sin embargo es interesante comentar que existen dos tipos de operaciones que involucran campos continuos: aquellas que operan con los valores de la muestra de solo un campo tales como promedio, mayor valor, etc. y operaciones que generan nuevos campos o computan valores a partir de análisis más complejos tales como intersección de campos, obtención de subcampos, etc.

Si bien un primer análisis como el realizado en el capítulo 6 muestra que las operaciones pueden definirse como métodos del Campo, cuando ellas son muy complejas ameritan un diseño más cuidadoso para no complicar la estructura de la clase; en particular, en nuestra arquitectura solo se definen como operaciones del campo aquellas que presentan menor complejidad. La implementación de otras operaciones ha sido realizada sobre una jerarquía separada que encapsula la

complejidad de aspectos como la equivalencia espacial, la conversión de representaciones, etc.

9.2 Aspectos de costos y factibilidad vinculados con la arquitectura propuesta

Resulta interesante analizar de qué manera impacta la arquitectura aquí propuesta en la factibilidad de realizar un proyecto de SIG.

Si bien, el objetivo de esta tesis es solamente reflexionar sobre aspectos del diseño vinculados a campos continuos y no atacar la problemática general de desarrollo de aplicaciones SIG, ciertos temas considerados en la tesis tienen un impacto innegable en el proceso de desarrollo; los analizamos seguidamente en forma somera.

9.2.1 Aspectos "externos" al proceso de diseño.

La sola introducción de tecnología SIG requiere de un análisis cuidadoso de ecuaciones de costo beneficio relacionadas a aspectos tales como adquisición, conversión, visualización y presentación de datos y a la posibilidad de encarar nuevos sectores de mercado. Un análisis interesante de esta perspectiva se presenta en [Aronoff 95]. Discutimos aquí ciertos aspectos relacionados a la factibilidad de aplicar los métodos descriptos en esta tesis y los costos/beneficios involucrados.

Si bien la aplicación sistemática de técnicas de la ingeniería de software tiende a modificar todo el entorno en el que se desarrollan y usan aplicaciones, el campo de sistemas de información geográfica tiene una característica distintiva: el problema de captura, digitalización e ingreso de datos. Estos aspectos, que generalmente involucran buena parte del esfuerzo económico y en recursos humanos en la construcción de una aplicación SIG suelen no verse afectados por la estrategia de desarrollo utilizada; en particular la utilización de objetos en todos los módulos de sistemas SIG tiende a simplificar el proceso de entrada de información pues las abstracciones que se manejan reflejan más "naturalmente" el dominio y no se precisa usar herramientas "extra" SIG como administradores de bases de datos relacionales donde es preciso superar el "mismatch" entre la naturaleza de los datos y las estructuras de representación (tablas). Por ejemplo mientras que las herramientas "convencionales" no soportan programación "visual" y el proceso de vinculación entre los datos geográficos y los no-geográficos se realiza manualmente, el uso sistemático de arquitecturas orientadas a objetos permite naturalmente resolver este problema mediante la utilización de ambientes integrados con interfases amigables.

9.2.2 Introducción del modelo orientado a objetos en el proceso de diseño

Como se dijo en esta tesis, la mayoría de los sistemas comerciales de SIG no están basados en el paradigma de objetos y utlizan combinaciones de bases de datos relacionales comerciales con sistemas propietarios. Si bien existen esfuerzos no solo para desarrollar ambientes de SIG orientados a objetos sino tambíen para

estandarizar aspectos tales como representación, intercambio y operaciones sobre objetos geográficos [OpenGIS96], es razonable decir que este campo está aún en su infancia. La utilización de objetos en sistemas SIG presenta un conjunto de ventajas similares a las que se encuentran en el uso del paradigma de objetos en otros dominios: mayor modularidad, extensiones menos costosas, mejores interfaces, etc. Sin embargo, la mayor dificultad para incorporar objetos en el proceso de desarrollo no proviene solamente de la falta de madurez de los diseñadores y/o programadores respecto al paradigma, sino al hecho de que la comunidad SIG tiene una cultura en la cual aún se le suele dar mayor importancia a aspectos tales como representación o performance respecto a los costos asociados al desarrollo. En este sentido, tanto los trabajos de [Medeiros et al.94] y [Pariente94] como los de nuestro grupo [Gordillo et al.97],[Gordillo et al.98] son los primeros en plantear que la problemática de desarrollo de sistemas SIG debe considerar además las dificultades de desarrollo y evolución del software y no solamente los problemas de performance o elección de representaciones.

En consecuencia el mayor esfuerzo en la re-educación de la comunidad SIG no pasa solo por la introducción del paradigma de objetos sino fundamentalmente por la concientización de la misma respecto a la necesidad de utilizar aspectos de la ingeniería de software. Esta transición no es muy diferente a la existente en otras areas (como diseño de sistemas de información con bases de datos) donde las tecnologías convencionales (como las bases de datos relacionales) son prevalentes. Es necesaria una inversión importante en capacitación y entrenamiento. La reutilización de software existente asi como de datos "legacy" es un tema que está siendo considerado en los consorcios internacionales que trabajan sobre el tema [OpenGIS96]. Consideramos que este es el aspecto más crítico respecto a la factibilidad real de introducir los aspectos aquí discutidos en la industria que desarrolla sistemas SIG. Nuestra experiencia en transferencia de conocimiento y de tecnología SIG a empresas que desarrollan sistemas SIG de gran porte ha tenido un feed-back muy semejante a la de la realizada respecto a la introducción del paradigma de objetos en otros dominios. Si bien pensamos que la principal dificultad de introducir estos conceptos en la industria tiene que ver con el cambio cultural necesario, y si bien los mayores costos de esta adopción tienen que ver con el proceos de re-entrenamiento de los planteles existentes, no tenemos dudas respecto al éxito de este emprendimiento. Los recursos que están invirtiendo los consorcios internacionales son enormes. Los ambientes de soporte de SIG orientados a objetos son cada vez más comunes y existe una tendencia muy perceptible en esa dirección. Al mismo tiempo, la creciente necesidad de hacer estos sistemas accesibles en plataformas como la Internet [Postmesil 97] hacen la elección del paradigma de objetos una necesidad imperiosa debido al uso extensivo del lenguaje Java en este contexto.

9.2.3-Impacto del uso de patrones de diseño y otros aspectos arquitecturales

La contribución más importante de esta tesis es el análisis de la problemática de campos continuos desde una perspectiva de diseño; mientras que la bibliografía tradicional de SIG limita el problema de los campos continuos al desarrollo de métodos de estimación y representación, practicamente ninguna importancia se le ha

dado al análisis de cuales son los problemas de diseño involucrados en la construcción de aplicaciones que involucren campos continuos. Esto no debe sorprender dada la cultura de esta comunidad tal como se explicó en el punto anterior. El objetivo de esta tésis ha sido reflexionar sobre estos problemas y plantear algunas soluciones que derivan de la aplicación sistemática de ciertos patrones de diseño así como el descubrimiento de otras "micro-arquitecturas" específicas de los sistemas SIG. Para evaluar la factibilidad y el costo/beneficio de aplicarlas en sistemas concretos se pueden usar varias estrategias. Hemos elegido un camino con dos enfoques concurrentes: por un lado implementar las construcciones aquí descriptas y por otro lado analizar el impacto de su uso en sistemas que evolucionan.

La arquitectura aquí presentada se encuentra parcialmente implementada en un conjunto de clases Smalltalk (aunque sin utilizar aspectos de Smalltalk diferentes a otros lenguajes OO lo cual permitiría implementarlas en forma inmediata en C++, Java u otro lenguaje orientado a objetos). Aún las operaciones más complejas de adminsitración de sistemas de referencia sofisticados han requerido poco tiempo de implementación y la estructura de las bibliotecas de clases resultantes es clara y facil de comprender por otros diseñadores.

Para cada una de las construcciones arquitecturales hemos comprobado que se cumplen con exactitud las consecuencias enunciadas en [Gamma et al.95] respecto a la utilización de determinados patrones como por ejemplo Strategy, Bridge o Decorator. Por ejemplo, hemos analizado de qué manera se comunican GeoObjetos, con o sin intervención de los objetos conceptuales decorados, observando ciertas diferencias importantes con los conceptos básicos de "decoración" de [Gamma et al.95] pero que no invalidan el enfoque.

Ciertos aspectos de la arquitectura podrían haber involucrado la utilización de otros patrones; por ejemplo la granularidad de objetos de posiciones (Location) y medidas (Punto Medido) sugieren la utilización de alguna variante del Flyweight; si bien el universo de objetos es suficientemente amplio para desalentar en principio el uso de este pattern (en comparación con un universo más restringido como un alfabeto de caracteres), existe un uso "limitado" de este patrón debido a que cuando varios GeoObjetos comparten una ubicación, el objeto Location utilizado es el mismo aunque sus características "extrínsecas" se mantienen fuera de el tal como descripto en el Pattern; desde este punto de vista, el desacoplamiento de la posición de un GeoObjeto respecto al GeoObjeto (un aporte novedoso de esta tesis) permite la utilización de esta estrategia.

Otro impacto innegable de la estructura aquí definida en la construcción de sistemas SIG (y en consecuencia en la factibilidad de su uso) radica en la separación de aspectos conceptuales respecto a los geográficos, así como en la mayor granularidad en la capa geográfica debido a la separación de la representación de la topología. Es habitual que los diseñadores de sistemas de información geográfica, así como buena parte de los roles existentes en el proceso de desarrollo, privilegien la representación sobre lo conceptual tal como explicado en el punto 9.2.2; este es el punto más crítico en el proceso de re-entrenamiento: la clave como dijimos anteriormente no es solamente la utilización de objetos sino la formación de una cultura diferente que ataque el desarrollo desde una óptica más ingenieril.

Resumiendo estos últimos puntos podemos expresar las siguientes conclusiones:

*La aplicación de las ideas desarrolladas en esta tesis tiene un impacto muy positivo en la implementación de micro-arquitecturas dentro de un SIG. Dicha aplicación es factible y la eficiencia de los algoritmos no difiere de la que se obtiene usando enfoques convencionales.

*La factibilidad real de aplicación de la arquitectura global (y en particular la de los campos continuos) en la industria está fuertemente condicionada por la adopción del paradigma de objetos para sistemas SIG. Este proceso como se explicó aquí es sumamente lento y dificultoso (no diferente al que se percibe en otras areas). Sin embargo se percibe un cambio muy pronunciado en los proveedores de ambientes de desarrollo para SIG en la dirección de usar objetos en este tipo de software. A pesar de ello nuestras presentaciones acerca del desacoplamiento de características geográficas de las conceptuales, y a la necesidad de separar algoritmos de estimación de los datos, etc. en grupos de desarrollo de SIG han sido en general muy bien recibidas y han sido utilizadas (aún con restricciones) en casos reales de uso.

*El uso sistemático de patrones de diseño en problemas complejos ha sido fuertemente recomendado en la comunidad internacional (con independencia del area de aplicación). Los costos de su uso son nulos más allá de entender correctamente los trade-offs que lleva el uso de cada patrón y los beneficios recibidos son en general notables.

10- Conclusiones y Trabajos Futuros

Se presentó en esta tesis un enfoque original para modelización de aplicaciones SIG usando objetos. Este trabajo es parte de un esfuerzo más abarcativo que involucra desde la extensión de aplicaciones "legacy" con nuevas características geográficas hasta la definición de un enfoque para acceder a datos geográficos en forma distribuida (por ejemplo en el contexto de aplicaciones Internet). El enfoque consiste en la utilización de la filosofía orientada a objetos, que aplicada sistemáticamente al área de SIG, posibilita describir las entidades y relaciones relevantes de los dominios en análisis. Además, la aplicación sistemática de un conjunto de patrones de diseño orientados a objetos permiten obtener soluciones flexibles a varios problemas que se identifican en los modelos orientados a objetos existentes (ver capítulo 8).

Esencialmente este enfoque consiste en decorar objetos de la aplicación con GeoObjetos, los cuales encapsulan la información geográfica; dichos GeoObjetos a su vez se caracterizan por su posición la cual puede estar expresada en distintos sistemas de referencia. Aspectos vinculados a la topología de los objetos geográficos se presentan en otro trabajo [Gordillo et al.98].

La parte central de esta tesis trata con el problema de modelización y manejo de datos continuos. Tal como se expresó en los capítulos 4 y 5, este es un problema con incidencia creciente en las aplicaciones SIG debido a la necesidad de representar datos que reflejan información no discreta, tales como temperaturas, crecientes de ríos, corrientes marinas, etc. El aporte principal de este trabajo es la caracterización de los problemas que aparecen en la modelización de datos continuos y una propuesta de solución usando conceptos modernos de la orientación a objetos. Así, y tal como se discute en el capítulo 6 definimos una arquitectura de objetos en la cual se desacoplan los diferentes problemas que aparecen en aplicaciones que administran datos continuos, tales como manejo de las muestras, métodos de estimación, relación entre objetos geográficos y muestras, etc.

En este momento se está trabajando en la validación de la arquitectura en dos direcciones simultáneas: por un lado se está diseñando un framework orientado a objetos que soporte el modelo completo, y en particular la administración de los datos continuos; el framework será implementado usando Visual-Works Smalltalk. Al mismo tiempo, se está modelizando un sistema de control y predicción de inundaciones como un ejemplo concreto de aplicación de la arquitectura.

En esta aplicación aparecen numerosos elementos relacionados con campos continuos, como por ejemplo las temperaturas, la presión, la densidad de lluvias, el viento, los tipos de suelos de las zonas de influencia, la modelización de los niveles de agua del río, etc. Esta aplicación servirá como feed-back tanto para validar las características fundamentales de la arquitectura como para servir de caso de prueba para el framework que se está diseñando.

Se van a analizar en lo sucesivo, distintos métodos de estimación para manipular datos en toda la esfera terrestre y se estudiarán a partir de distintas aplicaciones, nuevas operaciones que se requieren sobre estos datos, de manera de definir un sistema con los servicios necesarios para manipularlos Se estudiarán

nuevas maneras de combinar objetos concretos y datos continuos, así como la indexación de campos continuos, especialmente para grandes campos.

Como se ha mencionado, este enfoque permite construir sistemas de información geográfica siguiendo prácticas de la ingeniería de software moderna, obteniendo como resultado aplicaciones modulares, más faciles de modificar y/o extender.

11- Bibliografía

[Abitebul et al.87] Abitebul S. and Hull R., "IFO: A Formal Semantic Database Model", ACM TODS, 4, PP 525-565.

[ACM97] Comunications of the ACM. Object-Oriented Application Frameworks. October 1997, Vol. 40, Number 10.

[Alexander77] K. Alexander, S. Ishikawa, M. Silverstein, " A Pattern Language", Oxford University Press, 1977.

[Aronoff95] S. Aronoff, "Geographic Information Systems: A Management Perspective", WDL Publications, Otawa, Canada, ISBN:0-921804-91-1.

[Balaguer et al.97a] F. Balaguer, S. Gordillo, F. Das Neves : "*Patterns for GIS Applications Design*". In the proceedings of Patterns Language of Programming 1997.

[Balaguer et al. 97b] F. Balaguer, S. Gordillo: "Using Design Pattern to Define Interoperable GIS Models". International Conference and Workshop on Interoperating Geographic Information Systems. Santa Barbara, USA. Diciembre 3-4, 1997.

[Bancilhon92] Bancilhon F., Delobel C. and Kanellakis P. editors, "Building an Object Oriented Database System. The story of O2", Morgan Kaufmann Publishers, California, 1992.

[Bonfatti et al.95] Bonfatti, A. Dallari and P.D. Monari: *"Capturing more Knowledge for the Design of Geological Information Systems"*. Proceedings of ACM-GIS'95. Baltimore, Maryland, USA, pp 1-7.

[Camara et al.94] Camara G., Freitas U., Souza R., Casanova M., Hemerly A. and Medeiros C., "A Model to Cultivate Objects and Manipulate Fields", in Proc. 2nd ACM Workshop on Advances in GIS", pp 20-28, 1994.

[Carsjens et al.97] G. Carsjens and J.F.J.M. Smits, "Topological Relationships Integrated in Land Use Allocation", in proceedings of Geogeraphical Information'97: Joint European Conference and Exhibition on Geographical Information, Vol. 1, april 1997, pp 758-767.

[Cook et. al.96] P.D.Cook and P.O'Packi, "Urban Regional Planning. Applications Usses: Linking Transportation Models with GIS Updating Networks within Transit Agencies" Proceedings of First International Conference on Geographic Information Systems in Urban Regional and Environmental Planning, Ed. by Timos Sellis and Dimirtri Giorgoulis, 1996, PP 30-43.

[Copilen et al.95], Coplien and Schmidt (editors) : "*Pattern Languages of Program Design*". Addison Wesley, 1995.

[Cova et al.97] T.J.Cova and R.L. Church, "Modelling community evacuation vulnerability using GIS" , Inernational Journal of Geographic Information Science, Taylor&Francis, vol.11, number 8, december 1997, PP 747-761.

[DasNeves et al. 97] F. Das Neves, S. Gordillo, C.Mostaccio, A. Levato: "Toward a Foundation for Object Oriented GIS Design", Joint European Conference on Geographical Information, Vienna, Austria, Abril 17-18, 1997, pag. 58-63.

[Fayad97] M. Fayad, "Lessons Learned Building Reusable OO Frameorks for Distributed Software", Communications of the ACM, vol. 40, number 10, October 1997, PP 85-87.

[Floriani et al.97] L. Floreani, P. Maguillo, E. Puppo, "VARIANT – Processing and Visualizing Terrains at Variable Resolution", in proceedings of the ACM-GIS97: 5th International Workshop on Advances in Geographic Information Systems, De. R. Laurini, P. Bergougnoux, K. Makkiand N. Pissinou, november 1997, PP 15-19.

[Fowler97] Fowler, M : "Patterns : Reusable Object Model". Addison Wesley, 1997.

[Gamma et al.95] E. Gamma, R. Helm, R. Johnson, J. Vlissides, "Design Patterns. Elements of Reusable Object Oriented Sosftware", Addison Wesley Professional Computing Series, 1994.

[Goodchild92] Goodchild et al, "Integrating GIS and Spatial Data Analysis: Problems and Possibilities", International Journal of Geographical Information Systems, 6(5):407-424, 1992.

[Gordillo et al.97] S. Gordillo, F. Balaguer, F. Das Neves, "Generating the Architecture of GIS Applications with Design Patterns", in proceedings of the ACM-GIS97: 5th International Workshop on Advances in Geographic Information Systems, Ed. R. Laurini, P. Bergougnoux, K. Makkiand N. Pissinou, november 1997, PP30-34.

[Gordillo et al. 98] S. Gordillo, F. Balaguer: "Refining an object-oriented GIS design model: Topologies and Field Data" 6th ACM Workshop on Geographic Information Systems.Maryland, USA.November 6-7, 1998.

[Harvey69] D. Harvey: "Explanation in Geography". Ed. Edward Arnold, 1969

[Hull et al.87] Hull R., and King R., "Semantic Data Modeling: Survey, Applications and Research Issues", ACM Computing Surveys, vol. 19, Nro 3.

[IntelliCorp93] IntelliCorp Inc. "Kappa. Manual Set, IntelliCorp Inc. , El Camino Real West, Mountain View, CA 94040-2216, USA.

[Johnson97] R. Johnson, "Frameworks= (Components + Patterns)", Communications of the ACM, vol. 40, number 10, October 1997, PP 39-42.

[Kemp97] K. Kemp, "Fields as a Framework for integrating GIS and environmental process models. Part I and II: Representing spatial continuity", Transactions in GIS, 1997, Vol.1, number 3.

[Kim95] W. Kim: "Modern Database Systems: The Object Modelo, Interoperability and Beyond" ACM Press, 1995.

[Kosters95] G. Kosters, B. Pagel and H. Six, "Object-oriented requirements engineering for GIS applications", in proceedings of the ACM-GIS95: 3th International Workshop on Advances in Geographic Information Systems, De. R. Laurini, P. Bergougnoux, K. Makkiand N. Pissinou, november 1995, PP 61-68.

[Laurini et al.96] R. Laurini and Derek Thompson, "Fundamentals of Spatial Information Systems", Academic Press, 1996, ISBN 0-12-438380-7.

[Leung et al.97] Y. Leung and J. Yan, "Point-in Polygon Analysis Under Certainty and Uncertainty", GeoInformatica, Kluwer Academic Publishers, vol. 1, number 1, april 1997, PP 93-114.

[Medeiros et al.94] C. Medeiros, M. A. Casanova and G. Camara: *"The Domus project. Building an OODB GIS for environmental control"* Proceedings of IGIS'94. International Workshop on Advanced Research in GIS, Springer Verlag LNCS, N. 884, pp 45-54.

[Neugebauer91] L. Neugebauer: *"Optimization and Evaluation of Database Queries Including Embedded Interpolation Procedures"* Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data,May. 29-31, Denver, Colorado, pp 118-127.

[Oliveira et al.97] J. Lopes De Oliveira, F. Pires, C. Bauzer Medeiros, "An environment for Modeling and Design of Geographic Applications", GeoInformatica, Kluwer Academic Publishers, vol. 1, Numer 1, april 1997, PP 29-58.

[OpenGIS96] Consortium (OGC), 1996B, *The Open GIS Guide-A Guide to Interoperable Geo-processing*, Available at http ://ogis.org/guide/guide1.htm.

[Papadopoulos et al.97] A. Papadopolous and Y. Manolopoulos, " Nearest Neighbor queries in Shared-Nothing Environments", GeoInformatica, Kluwer Academic Publishers, Vol. 1, number 4, december 1997, PP 369-392.

[Pariente94] Dillon Pariente, "Estimation, Modelisation et Langage de Declaration et de manipulation de Chamos Spatiaux Continus", Tesis doctoral presentada en L'Institut National Des Sciences Appliquees de Lyon, 1994.

[Peuquet88] D.J. Peuquet: "Representations of Geographic space: Toward a Conceptual Sysnthesis". Annals of the Association of American Geographers, 78:375-94.

[Plumer et al.97] L. Plumer and Gerhard Groger, "Achieving Integrity in Geographic Information Systems-Maps and Nested Maps", Geoinformatica, Kluwer Academic Publishers, vol. 1, number 4, december 1997, PP 345-367.

[Postmesil97] Postmesil, M : "*Maps Alive : Viewing Geospatial Information on the WWW*". Proceedings of the six International World Wide Web Conference, 1997. Available at http ://www6.nttlabs.com/Hypernews/get/PAPER130.htm.

[Rational97] "UML", www.rational.com/uml.

[Rmbaugh et al.91] Rumbaugh, M. Blaha, M. Premerlani and W. Lorensen: *"Object-Oriented Modeling and Design"*. Prentice Hall, Englewoods Cliff, New Jersey, 1991

[Samet89] H. Samet, "The Design and Analysis of Spatial Structures", Addison Wesley Publishing Company, 1989.

[Schmid et al.97] H. Schmid, "Systematic Framework Design by Generalization", Communications of the ACM, vol. 40, number 10, October 1997, PP 52-59.

[Schmidt95] D. Schmidt, *"Using Design Patterns to Develop Reusable Object-Oriented Communication Software"*. Comm. of the ACM, October 1995, pp 65-74.

[Seeger88] B. Seeger and H.Kriegel, "Techniques for Design and Implementation of Efficient Spatial Access Methods", in proceedings of the Fourteenth International Conference on Very Large Databases , aug.29/sept.1 1988, PP 360-371.

[Serra97] P. Serra, "Urban Land Use Data Spatio-Temporal Modelling", in proceedings of Geogeraphical Information'97: Joint European Conference and Exhibition on Geographical Information, Vol. 1, april 1997, PP 779-788.

[Star et al.90] Star J. and Estes J: *"Geographic Information Systems. An Introduction"*. Prentice Hall. 1990.

[Tryfona et al.95] Tryfona N. and Hadzilacos T., "Geographic Applications Development: Models and Tools for the Conceptual Level", in Proc of the 3rd ACM International Workshop on Advances in Geographic Information Systems, PP 19-28, 1995.

[Vet97] R.M.A. de Vet, "The Application of Objexct Technology in GIS", in proceedings of Geogeraphical Information'97: Joint European Conference and Exhibition on Geographical Information, Vol. 1, april 1997, PP 197-200.

[Ware et al.97] J. Mark Ware and C. Jones, "A data Model for Representing Geological Surfaces", in proceedings of the ACM-GIS97: 5th International Workshop on Advances in Geographic Information Systems, De. R. Laurini, P. Bergougnoux, K. Makkiand N. Pissinou, november 1997, PP 20-23.

[Yuan97] m. Yuan, "Use of knowledge acquisition to build world fire representation in Geographical Information System", International Journal of Geographic Information Science, Taylor&Francis, vol.11, number 8, december 1997, PP 723,745.

[Zaniolo et al.97] C. Zaniolo, S. Ceri, C. Faloustsos, R. Snodgrass, V. Subrahmanian and R. Zicari, "Advanced Database Systems", Morgan Kaufman Publishers, 1997.