Visualización con MapObjects de la localización de móviles en un sistema de información distribuido orientado a objeto

Pedro R. Muro-Medrano (1) (2)
F. Javier Zarazaga Soria (1)
J.A. Bañares Bañares (1)
J. Guillo Pitarque (1)
D. Infante Pérez (1)
R.J. López Alejaldre (1)
F. Javier Salas Procas (3)

(1) Departamento de Informática e Ingeniería de Sistemas
Centro Politécnico Superior, **Universidad de Zaragoza**
María de Luna 3, 50015 Zaragoza
(2) muro@rioja.cps.unizar.es – Tel. 976 761950

(3) Departamento de I+D, Área de Software
Teltronic s.a.
Leopoldo Romeo 18, 50002 Zaragoza

Resumen

En este trabajo se presenta la motivación e ideas básicas de un sistema de información distribuido orientado a objeto para el seguimiento y control de flotas. El trabajo presenta la arquitectura de software adoptada y se centra en el módulo para la visualización en mapas digitales mediante MapObjects En este contexto se analizan las ventajas de disponer de jerarquías de objetos especializadas en la manipulación de mapas digitales y en las facilidades que esto aporta para su integración. Se muestra también como ejemplo el modelo de objetos básico utilizado y algunos aspectos de su implementación en C++.

Agradecimientos

Este trabajo ha estado parcialmente financiado por el CONSYD de la Diputación General de Aragón a través del proyecto P-18/96. El trabajo relacionado con la arquitectura de objetos distribuidos está parcialmente basado en resultados originados por el proyecto TAP95-0574 de la CICYT.

1. Introducción

El avance de las tecnologías de la información y de las comunicaciones está permitiendo el desarrollo de sistemas a precios muy asumibles por una creciente gama de clientes institucionales, industriales y privados, con unas utilidades que hace muy pocos años sólo podían ser desarrollados por empresas lider en tecnología y comprados por clientes con grandes presupuestos.

Dentro de este tipo de productos se encuentra una amplia gama de sistemas de navegación y ayuda a la gestión de los transportes y su logística. Si bien los precios de los componentes necesarios para su construcción van bajando paulatinamente, la dificultad técnica para su desarrollo radica en el hecho de que este tipo de sistemas exigen al equipo de desarrollo conocimiento para la integración de diversas tecnologías: tarjetas con microprocesadores, integración de sensores, comunicaciones por radio terrestre o por satélite, comunicaciones entre redes de computadores, software de comunicaciones, sistemas de información geográfica, interfaces gráficos de usuario, acceso y gestión de bases de datos avanzadas, ingeniería del software, etc.

Krakiwsky [KRAK93] clasifica los sistemas de navegación en cuatro tipos:

- sistemas autónomos, son autocontenidos sin posibilidades de comunicación;
- sistemas asesoría, reciben normalmente información de congestión de tráfico en tiempo real;
- sistemas de seguimiento y control de flotas, incluyen comunicación en dos sentidos y un centro de gestión;
- sistemas de inventario, son utilizados para recolección y seguimiento coordinado de información relacionada con las carreteras.

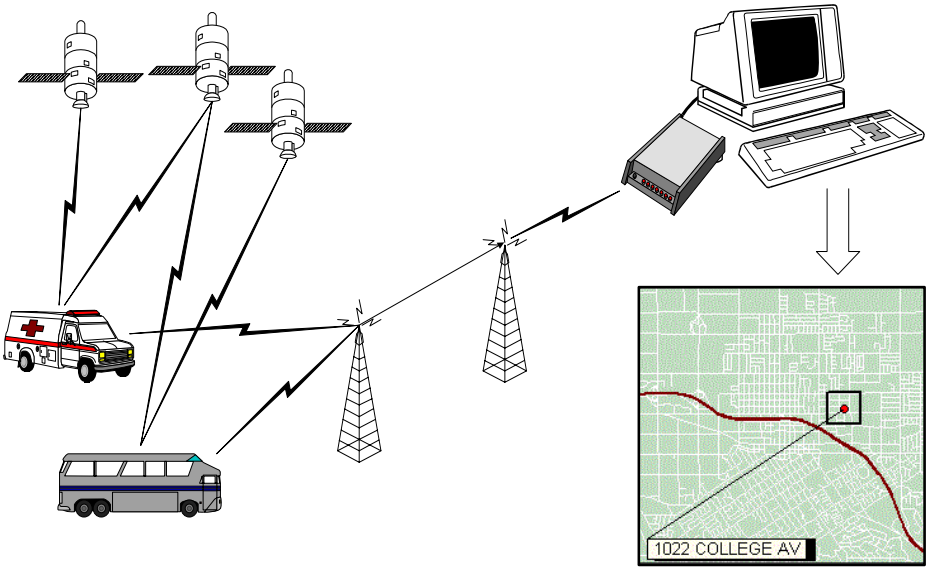

Figura 1: Sistema de navegación IVHS de tipo gestión de flotas.

De todos ellos, son los sistemas de seguimiento y control de flotas [BBH94] donde se enmarca el sistema objeto del presente trabajo (Figura 1). El corazón de dichos sistemas

es un centro de control que monitoriza los vehículos de la flota en un mapa digital de carreteras y calles. Esta arquitectura requiere un enlace de comunicaciones con los móviles y capacidades para el reconocimiento de direcciones. Suele resultar necesario también capacidades de persistencia en base de datos para poder realizar análisis de rutas y, eventualmente, dependiendo de la aplicación, la integración con información de las bases de datos corporativas. En relación con las tecnologías utilizadas para el posicionamiento, son varias las disponibles, pero es la tecnología basada en GPSs [KAPL96] la que se ha impuesto en el mercado. Según datos de 1993 era ya utilizada por un 52% de los sistemas [KRAK93] y en la actualidad es utilizada ya por encima del 66% de los sistemas de control de flotas [KRAK96].

La necesidad de optimizar los costes de los servicios y el incremento de la competencia en el sector del transporte ha inducido a muchas empresas a interesarse por estos sistemas de seguimiento y control de flotas. Así en los últimos años han surgido en España sistemas comerciales básicos (p.e. [BARR97]) y diversos proyectos de este tipo en los sectores de la paquetería, taxis, autobuses urbanos y de largo recorrido, así como en servicios públicos, fundamentalmente de protección civil (policía, bomberos, sanidad, …). Además este interés no ha estado limitado al transporte por carretera sino que ya hay sistemas en marcha para el sector naval y el transporte ferroviario [RP97].

Todo este extenso dominio de aplicaciones requiere una serie de servicios básicos comunes (adquisición, análisis, visualización, comunicaciones, …), sin embargo, los requisitos y necesidades específicas del cliente que han de ser satisfechas es muy variable en cuanto a magnitud de los recursos, máquinas, funcionalidad, etc., lo que exige generalmente soluciones “ad hoc”.

La estrategia elegida por nuestro grupo de I+D al introducirnos en este campo, ha consistido en desarrollar una infraestructura flexible que soporte las funcionalidades básicas y que posibilite la fácil integración de dichas funcionalidades en las aplicaciones. Por este motivo nos hemos planteado una arquitectura de diseño flexible donde aspectos como escalabilidad, interoperabilidad, sustituibilidad, extensividad y reusabilidad resultan de gran importancia. La aproximación técnológica adoptada se guía por los principios de sistemas abiertos en base a una arquitectura distribuida cliente/servidor utilizando las nuevas tecnologías de desarrollo de software orientado a objeto. Una elemento importante que nos ha facilitado el desarrollo del software para la visualización de los datos de localización en mapas digitales ha sido la posibilidad de utilizar las librerías de objetos de mapas de MapObjects dentro de nuestro programas orientados a objeto desarrollados en C++.

El objetivo de este trabajo es presentar la motivación e ideas básicas de OODISMAL, un sistema de información distribuido orientado a objeto para el seguimiento y control de flotas. El trabajo presenta la arquitectura de software adoptada y se centra en el módulo para la visualización en mapas digitales mediante MapObjects. En este contexto se analizan las ventajas de disponer de jerarquías de objetos especializadas en la manipulación de mapas digitales y en las facilidades que esto aporta para su integración.

2. Panorámica de OODISMAL

OODISMAL (Object Oriented Distributed Information System for Automatic Movil Location). está organizado en base a diversos módulos que hemos clasificado en:

- **Servicios**, que constituyen las utilidades básicas como:

- adquisición de datos de los móviles a través de enlaces de comunicación (actualmente se dispone del software para la adquisición de datos de *GPS* vía radiotelefonía *trunking* en base al terminal T500 voz con opción GPS desarrollado por Teltronic),
- utilidades para mejoras de precisión y de ajuste a rutas,
- análisis de rutas (relacionados con distancias, tiempos, velocidades, posiciones) y correlación entre rutas reales y previstas,
- gestión de persistencia,
- simulación de móviles para depurar otras aplicaciones y hacer demostraciones,
- interfaz gráfico de usuario para el acceso a los datos (estas visualizaciones pueden hacerse también por *internet*),
- visualización de datos en mapas digitales (estas visualizaciones pueden hacerse también por *internet*).

Estos servicios proporcionan tanto las utilidades para el funcionamiento en *tiempo real* como el basado en datos almacenados.

- **Aplicaciones de usuario**, que constituyen soluciones integradoras con objetivos más especializados del cliente. Actualmente se está trabajando en las siguientes aplicaciones:
  - Un sistema de información de ayuda a los usuarios de autobuses urbanos de Zaragoza.
  - Un sistema de ayuda a la toma de decisiones para la asignación de taxis en base a peticiones telefónicas.

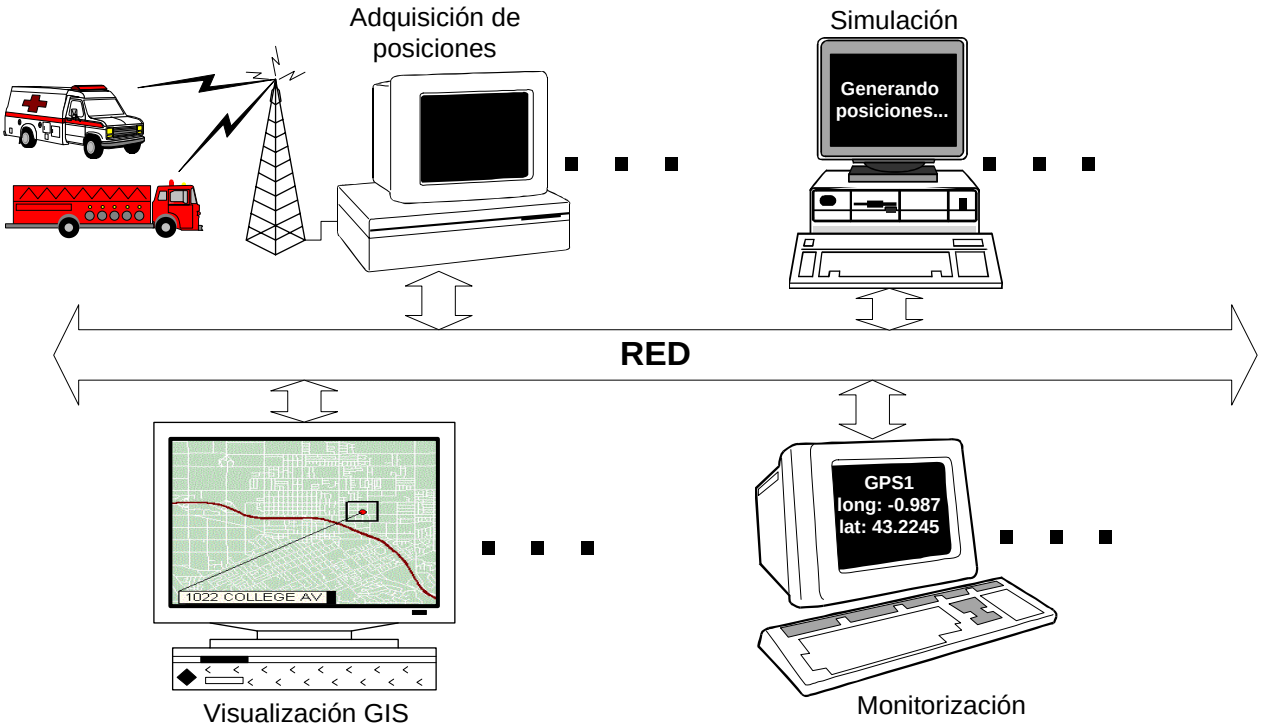

Figura 2: Arquitectura de aplicaciones

La distribución implica distintos procesos de ejecución que pueden estar residentes en la misma máquina o en máquinas distintas. De esta forma resulta fácil ajustar el sistema a

distintas magnitudes de las necesidades de los usuarios, así el sistema puede ir escalando incrementando el número de máquinas para ajustarse a las prestaciones y puestos de trabajo deseados.

Esta tecnología distribuida posibilita el desarrollo de aplicaciones ligeras donde se resuelve la problemática específica de la aplicación mientras que están distribuidas el resto de funcionalidades y servicios.

Esta arquitectura proporciona grandes posibilidades de escalamiento del sistema para ajustarlo a las peculiaridades del cliente.

Otro aspecto de interés lo constituye las facilidades para el trabajo conjunto de software heterogéneo. Dicha hetereogeneidad no sólo radica en la posibilidad de tener software corriendo en distintos sistemas operativos, lenguajes de programación y distintos tipos de máquinas físicamente distribuidas, sino que sería posible también integrar en el sistema el uso de eventuales aplicaciones ya disponibles y que no habian sido desarrolladas pensado en esta integración.

3. Arquitectura de OODISMAL como sistema distribuido orientado a objeto

El paradigma de programación orientada a objeto proporciona técnicas de análisis, diseño e implementación para la construcción de software que es extensible, reusable y menos costoso de producir y mantener que el software orientado a la función. CORBA es una especificación para una arquitectura orientada a objeto distribuida estándar para aplicaciones. CORBA fué definido por el Object Management Group (OMG) [OMG96b] y actualmente existen varias de implementaciones proporcionadas para los entornos y lenguajes más representativos.

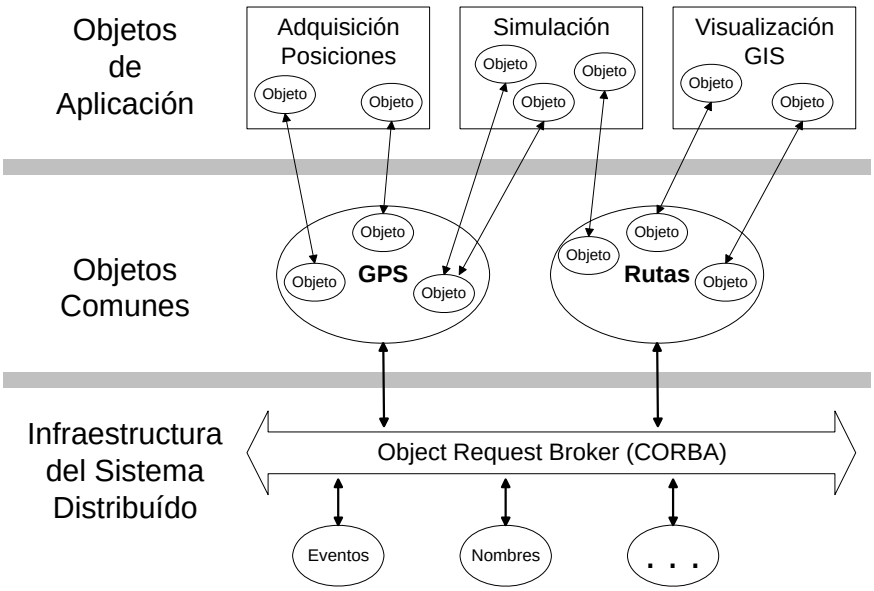

Figura 3: Arquitectura del entorno organizada por niveles

Por medio de la computación distribuida dos o más piezas de software pueden compartir información. Dichas piezas pueden estar ejecutándose en el mismo computador o en diferentes computadores conectados a la misma red. La mayor parte de la computación distribuida está basada en el modelo cliente/servidor (un software cliente, que requiere informaciones o servicios y, un software servidor, que proporciona la información o servicio).

La Figura 3 muestra las líneas básicas de la arquitectura de software de OODISMAL para soportar la distribución de objetos, en el que se ha adoptado una aproximación basada en el estándar CORBA. La arquitectura está formada en base a tres niveles de reponsabilidad:

- La **infraestructura** proporciona el entorno de computación distribuido para la aplicación. Estos servicios incluyen comunicaciones, almacenamiento persistente de datos, interfaz de usuario, distribución de eventos, gestión de excepciones, etc. OODISMAL utiliza la infraestructura de servicios y facilidades definidas por la Arquitectura de Gestión de Objetos de OMG (OMA).
- Los **objetos comunes** constituyen las entidades funcionales comunes a través de las aplicaciones. Los objetos comunes proporcionan un modelo común para estas entidades, posibilitando un desarrollo e integración más rápida. Estos objetos especifican datos y comportamiento requerido para la interoperatividad entre las aplicaciones. En el caso de OODISMAL, estos objetos proporcionan la infraestructura para compartir datos de localización en tiempo real e informaciones sobre rutas almacenadas en soportes persistentes.
- Los **objetos de aplicación** proporcionan funcionalidades de aplicación específicas. Estos objetos definen datos y comportamiento construidos sobre los datos y comportamientos comunes, lo que permite a la aplicación interoperar con otras aplicaciones. Los objetos de aplicación definen también las reglas del negocio (business rules) y el interfaz de usuario para las aplicaciones. En nuestro caso corresponden con módulos especializados en la adquisición, visualización, simulación, disponibilidad para internet y distintas aplicaciones para gestión de flotas y logística.

4. Módulo de visualización SIG basado en MapObjects

Desde el punto de vista de satisfacción para el usuario, un aspecto fundamental del sistema de localización es la posibilidad de visualizar las localizaciones de los móviles sobre mapas o planos. Generalmente, la integración en las aplicaciones de las infraestructuras para este tipo de visualización de datos SIG consiste en la utilización de entornos SIG completos (tipo ArcInfo, ArcView, MapInfo, …) a los que se pasa todo el control. Esta aproximación conlleva varios inconvenientes como el tener que utilizar (y costear) una aplicación con grandes funcionalidades para utilizar sólo unas pocas. Por otra parte, aunque estas aplicaciones disponen de lenguajes de programación específicos, éstos están especializados en aspectos relacionados con el SIG pero no resultan tan competitivos (frente a lenguajes y entornos como los basados en C++ o Basic) cuando se trata de la programación de utilidades generales. MapObjects a venido a cubrir este pequeño vacío y está siendo muy agradecido por programadores a los que les interesa tener el mayor control posible del software que desarrollan. La aproximación que hemos adoptado en nuestro sistema ha sido utilizar las librerías de objetos de MapObjects con Visual C++ accediendo a información distribuida de datos de localización vía CORBA.

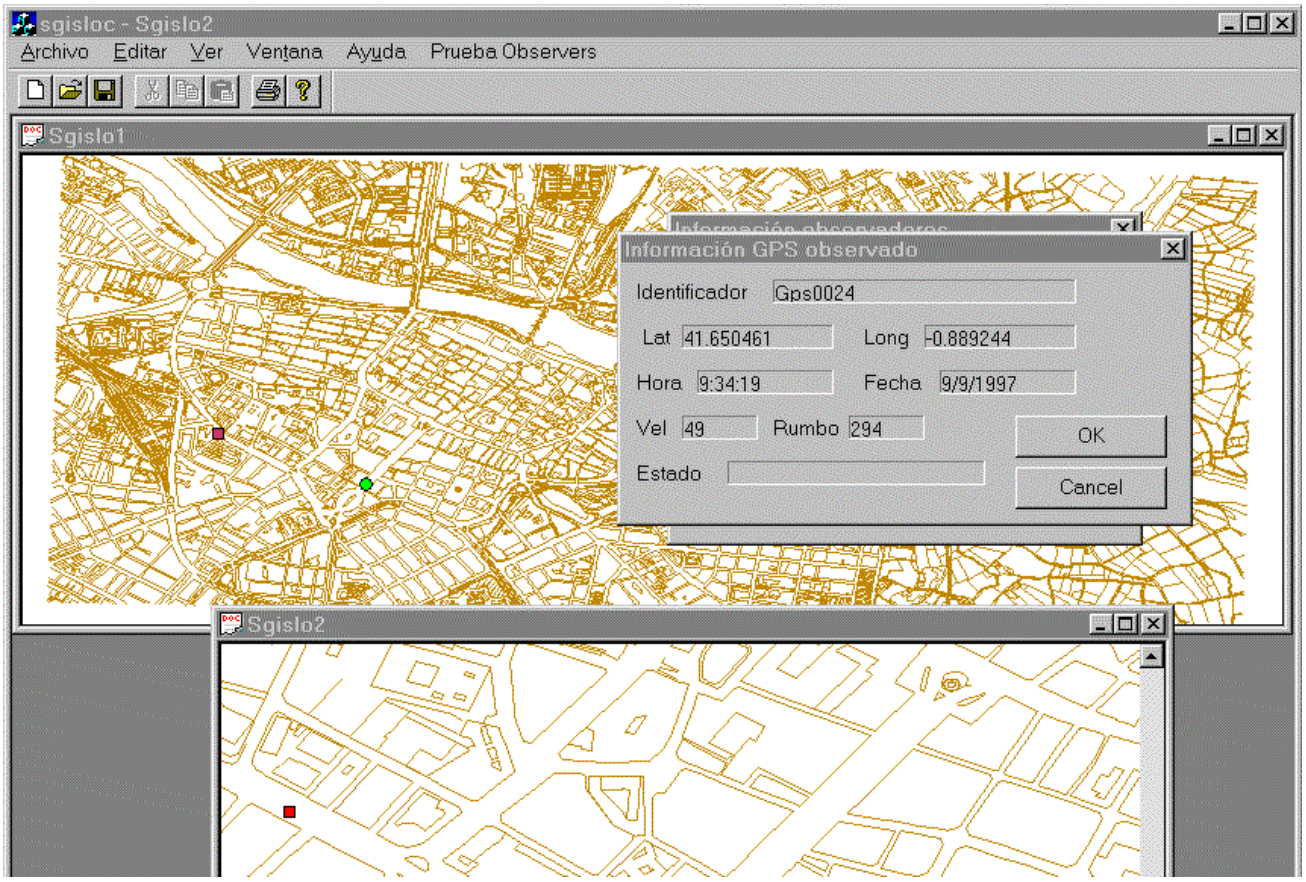

Figura 4: Ejemplo de visualización sobre un plano de la ciudad de Zaragoza

La aplicación de visualización puede ejecutarse en el mismo computador donde se adquieren las informaciones de localización por radio o en otro cualquiera. De hecho, debido al diseño distribuido del sistema, es posible ejecutar a la vez varias aplicaciones de visualización monitorizando los mismos o diferentes móviles. La aplicación está diseñada de forma que es capaz de localizar en la red qué computadores disponen de datos de localización de móviles y posibilita al usuario elegir cualquier conjunto de ellos para visualizar en tiempo real.

La Figura 4 muestra un ejemplo del módulo de visualización donde puede observarse el seguimiento de 2 móviles sobre una ventana con el mapa de Zaragoza y otra ventana con una zona del mismo mapa ampliado.

El usuario tiene posibilidad de:

- modificar formas, tamaños y colores de los móviles;
- elegir etiquetas y datos a mostras por cada uno;
- disponer de varias ventanas con iguales o distintos mapas, hacer zooms, …;
- almacenar rutas, visualizar rutas tanto en tiempo real como rutas pregrabadas;
- ahora estamos trabajando para proporcionar a las rutas capacidades de análisis.

4.1 Objetos para la visualización de los datos de localización

MapObjects es un conjunto de componentes de mapping y SIG para desarrolladores de aplicaciones sobre Windows. Consiste en un control OLE y una colección de objetos OLE

programables. MapObjects puede operar en entornos de programación que soporte controles OLE, concretamente se encuentra disponible para Visual Basic y Visual C++. Por ejemplo, para el programador de C++, MapObject proporciona objetos y clases que pueden interoperar con el resto de objetos de la aplicación accediendo a sus propiedades, requiriendo servicios y creando instancias. Aunque no están disponibles como jerarquía, MapObjects proporcionan distintas agrupaciones de clases y objetos agrupados según su funcionalidad en objetos para: acceso de datos, reconocimiento de direcciones, dibujo de mapas y elementos geométricos.

La Figura 5 muestra una vista parcial del modelo de objetos utilizado en la aplicación. El modelo muestra los objetos de dibujo de mapas de MapObjects y su relación con algunos objetos de la aplicación. **`MapControl`** es el objeto más importante que controla todos (**`LayersCollection`**) los niveles de mapa (**`MapLayer`**). MapObjects proporciona un tipo de nivel de mapa especializado en la gestión de datos en tiempo real (**`EventTrackingLayer`**) que es el utilizado para las informaciones de localización en tiempo real. Este objeto mejora la eficiencia en la visualización de datos que cambian pero por contra sus utilidades se encuentran muy limitadas en comparación con los objetos **`MapLayer`**.

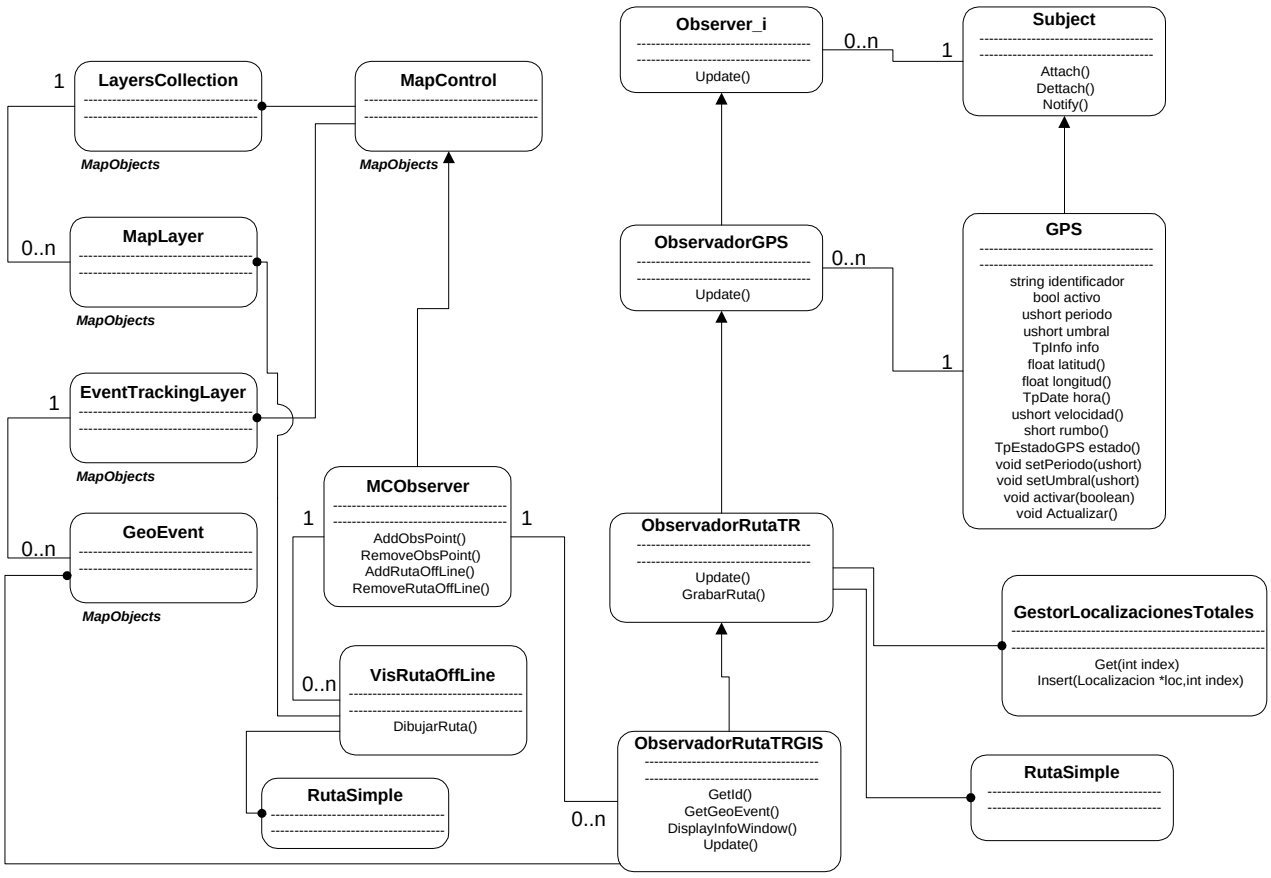

Figura 5: Vista parcial de modelo de objetos

El diseño de los objetos de la aplicación siguen un patrón sujeto/observador [GHJV94]. Los objetos de cualquier aplicación que deseen visualizar datos de localización en tiempo real se suscriben como observadores a **`GPS`**s y a partir de ese momento estos informan de cualquier cambio (los **`GPS`** son objetos CORBA).

Sin embargo la utilización de MapObjects no está exenta de problemas dentro del ámbito de la programación orientada a objeto. A pesar de que se puede trabajar con las clases y objetos proporcionados por MapObjects en los programas de C++, tienen sus capacidades relativamente limitadas. Así, por ejemplo, desde el punto de vista de la programación en C++ existen limitaciones en las posibilidades de herencia y el acceso a ciertas informaciones (por ejemplo, los objetos importantes de visualización de mapas no soportan herencia múltiple).

5. Conclusiones

El trabajo ha presentado una arquitectura distribuida orientada a objeto para el desarrollo de un sistema de localización de móviles. Esta aproximación, además de proporcionar las ventajas de ingeniería del software provenientes de las aproximaciones orientadas a objeto, proporciona un alto grado de flexibilidad y reusabilidad al permitir fácilmente un gran rango de desarrollo de sistemas, desde pequeños basados en un único computador personal, hasta la integración de varios computadores personales y/o estaciones de trabajo basadas en UNIX. El módulo de visualización de información geográfica utilizado para mostrar la posición de los móviles en mapas, se ha realizado utilizando desde C++ las librerías de objetos proporcionadas en MapObjects. Nuestra experiencia con estas librerías ha sido muy satisfactoria al permitirnos una integración muy cómoda y versátil con el resto de código de C++, aunque dichos objetos carecen de algunas de las agradables cualidades que desean disponer los programadores avanzados de C++.

6. Bibliografía

[BBH94] A. Bethmann, K. Brocke, S. Harrer. “Sistema automático de localización de vehículos”. *Comunicaciones Eléctricas*, pp. 129-135. 2º trimestre, 1994.

[BARR97] E. Barrios. “Seguimiento Competitivo de Vehículos”. *RedesLan*, pp. 110-111. Febrero, 1997.

[GHJV94] E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Desing Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley Publishing Company. 1994.

[KAPL96] E.D. Kaplan (ed.). *Understanding GPS Principles and Applications*. Artech House Publishers. 1996.

[KRAK93] E.J. Krakiwsky. “Tracking the Worldwide Development of IVHS Navigation Systems”. *GPS World*, V 4, N 10, pp. 40-47. October, 1993.

[KRAK96] E.J. Krakiwsky. “Driving ITS Development: Technology and Market Forces”. *GPS World*, V 7, N 10, pp. 50-55. October, 1996.

[RP97] J. Roca, J.L. Pérez. “All Aboard!. On track with Catalonia´s Trains”. *GPS World*, V 8, N 6, pp. 20-28. June, 1997.

[OMG96a] Object Management Group. *CORBA Services: Common Object Services Specification*.. Framingham, MA: Object Management Group, July, 1996.

[OMG96b] Object Management Group. *The Common Object Request Broker: Architecture and Specification, Revision 2.0*. Framingham, MA: Object Management Group, July, 1996.

[OMG95] Object Management Group. *Object Management Architecture Guide, Version 3.0*. Framingham, MA: Object Management Group, 1995.

[RUMB91] J. Rumbaugh et al. *Object Oriented Modeling and Design*. Englewood Cliffs, N.J.: Prentice Hall, 1991.