Proceedings of the First International Workshop on Telegeoprocessing: TELEGEO'99. 1999, p. 23-30

# A CORBA Object-Oriented Architecture to Provide Distributed GPS data to GIS applications.

Pedro R. Muro-Medrano
Daniel Infante
Jorge Guillo
Javier Zarazaga
José A. Bañares

Computer Science and System Engineering Department
Centro Politécnico Superior
Universidad de Zaragoza
María de Luna 3
50015 Zaragoza, SPAIN

{prmuro,dinfante,jguillo,javy,banares}@posta.unizar.es
http://diana.cps.unizar.es/iaaa

## Abstract

This paper shows a distributed object-oriented architecture to provide GPS data to GIS applications. Data captured in real time by GPS units are sent via radio to the computer. Several sources of this real data in addition to simulated GPS data are distributed in a computer network. These servers of GPS data can be accessed by distributed client applications. A CORBA based infrastructure provides the integration and distribution mechanisms. Client applications range from simple GUIs for GPS remote control to GIS applications used for automatic vehicle monitoring.. The basic components to provide the radio communications and GPS data servers are presented in detail.

## 1.Introduction

The reduction in price of radio communications together with steps forward in the field of data captured in digital format (using handheld computers in the fields, and GPS units), and the development of the distributed object-oriented technology are the technologies that have revolutionised the possibilities of GIS applications functionality. Automatic vehicle monitoring (AVM) is a prototypical application that requires the integration of mentioned technologies: Radio communication that provides wireless interaction between different devices, communication software, Global Positioning Systems (GPS) [5], GIS, access to databases to integrate data with non-spatial data, etc. [6]. These technologies enable the acquisition of vehicle locations in real time and the visualisation of vehicles in a map.

The need to optimise service costs, the increase of rivalry in the transport sector, and the interest of public institutions to promote public transport have persuaded many enterprises to integrate tracking and fleet control systems with their information systems. Fleet management aid systems that incorporate these technologies permits the recompilation of real operation parameters. They offer users a real time information that allows incident control and a valuable information for end-users of public transport [1].

Although AVM applications require a set of common services (acquisition, communications, GIS, ...), requirements and needs of each client are very different in resources and functionality, and therefore, they need "ad hoc" solutions. The work in this domain with new software technologies is just beginning [3]. However, the cost of these "ad hoc" solutions may only be affordable by the design of a flexible architecture that makes use of new software technologies, such as distributed object-oriented systems, and where features as interoperability and reuse are appropriately considered. In this sense, it is required that Geographic Information Systems (GIS) provide much more than a map to provide the framework for all the major functionality of an application. Applications must be able to transform themselves intelligently by reusing useful parts and incorporating new technologies to extend their capabilities [2].

This paper shows the basic ideas of OODISMAL; an object oriented distributed information system for mobile automatic location. OODISMAL provides the basic components to integrate radio communications and real-time data captured by GPS units with GIS components. These components may be easily adapted to any kind of radio or sensor to capture data. Firstly, the technological approach to develop the OODISMAL architecture and its basic components is explained. Following, design decisions and a detailed explanation of modules and operation of main components is given. In §3 the radio component server that provides to all computers in the local net the functionality of a trunking radio connected to a PC. In §4 the data acquisition component that uses the radio to extract from received messages the data captured by different GPS units. The integration of the radio and data acquisition components with GIS to develop final applications is detailed in §5. Finally, in §6 the conclusions are presented.

## 2. OODISMAL Architecture

## 2.1 Technological approach

The technological approach that has been adopted in this work is based on the new technologies of distributed object-oriented systems. The object model makes easier the co-operative work and the reuse by means of encapsulation. However, it is not sufficient. The fast evolution of new technologies suppose the cohabitation of a great diversity of machines, operating systems and programming languages that makes difficult the maintenance of systems. It is important to make compatible

new applications with applications that use old and heterogeneous technologies. The present tendency to afford these difficulties is the development of distributed client-server applications using component software technology [7], [10].

Component Software technology makes possible the development of light applications where it is solved the specific problematic of the application, whereas the basic functionality is provided by reusable components. A distributed client-server application, where clients and servers may be resident in the same or different machines, makes possible to increase the system functionality without the modification of reused parts. Servers are used only when it is necessary, and may be shared by several light remote or local clients.

Our research group has therefore been considering international computer industry standards for developing distributed object-oriented systems. In particular the Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) [9]. CORBA has been used as the middleware infrastructure to provide interoperability and to distribute functionality. In CORBA, applications are viewed as objects, and their functionality is provided through their object interface, no matter they are programmed in an object-oriented language [VIN95]. From the implementation point of view, clients of a remote CORBA component deal with them as local objects. CORBA components may be running in the same or in different computers in a transparent way.

It has been necessary to consider other component technologies because many commercial components often use the Microsoft's Distributed Component Object Model (DCOM), which is the de facto "other standard". Finally, we have also considered Java, the third component technology to be considered, because it supposes the perfect complement to CORBA. CORBA offers an infrastructure that provides interoperability, and Java provides an infrastructure for mobile code. In this way we may do CORBA ubiquitous in the net [8]. The main advantage of this language is that a Java program may be easily transformed in a Java applet that may be downloaded in any computer connected to Internet. It makes Java the best option to develop light client that provides portable GUI to access the functionality of server components.

## 2.2 OODISMAL Overview

OODISMAL is an information system that is composed of a set of distributed components in a LAN. These components interoperate between them using the CORBA infrastructure, and provide the basic services for location data acquisition, radio telecommunications, and the storing and processing of acquired data. If every object were a CORBA object it would suppose an unnecessary overhead of communications and would produce coupled components. Consequently, each component interface is defined clearly by a reduced number of CORBA objects. Each component works as a server of its interface CORBA objects and uses, as a client, the CORBA objects that needs from other components.

In object-oriented systems, components typically interact with each other by explicitly invoking their services. OODISMAL also uses an event based, or implicit invocation mechanism. The idea is that instead of invoking a procedure directly, a component can announce one or more events. Other components in the system can register an interest in the reception of an event. When the event is announced, the registered component is notified and executes some service.

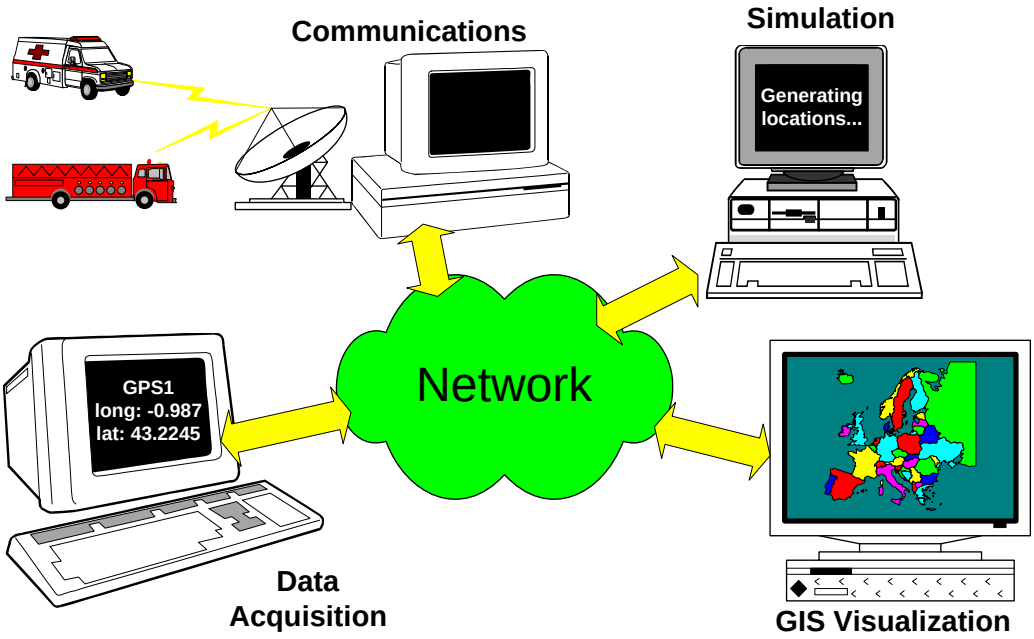


**Figure 1. OODISMAL system**

Figure 1 shows OODISMAL as a distributed system with the basic components that interoperate using CORBA to visualise in digital maps real-time location of vehicles. The working of these components is orchestrated by the reception of radio call with GPS data information, and their responsibilities are the following:

- The *radio component* works as a CORBA server that offers to the CORBA bus the messages received by the

radio connected to a PC in the LAN. It also offers the functionality to make different kind of call through the radio.

- The *data acquisition component* works as a client of the radio component to receive data from remote GPS, and as a server that offers location information to the CORBA bus.
- The *simulation component* allows the developer to tune the system simulating the reception of messages with GPS information. It has the same interface that the data acquisition component. In this way its clients do not distinguish real form simulated data. The only difference is that simulated location data are taken form a database through the persistence component.

Functionality of these components is offered to the CORBA bus. So, any new client application may be introduced in the system simply by registering it for the events provided by components. CORBA supports interoperability, and its client/server style provides strong support for reuse. For example, the simulation component and the acquisition component may be interchanged without affecting its clients because they offer the same interface. Enhanced versions of these components may be developed without affecting client applications. When a vehicle tracking application initiates, it may look for CORBA servers that provides vehicle locations in the LAN, and can choose any of them.

Previous functionality may be easily integrated in a GIS framework. It allows the visualisation in digital maps of real-time vehicle locations provided by the data acquisition component, the access to the radio of visualised vehicles, or the visualisation of analysis results of routes recorded in a database.

Finally, computers in which components reside may offer light Java clients that may be downloaded from any computer connected to Internet. It allows the user the supervision of components. For example, it is possible to access data acquisition components and show in a window the last data received from each GPS. A simple version of the tracking application has been developed in Java. It allows Internet users the visualisation in digital maps of real-time locations.

# 3. Radio Component

This component has the responsibility of handling and making accessible, to several clients, all radio typical features using a narrow and well-defined interface of allowed operations. Those features include communicating information and establishing different kinds of calls using the radio physic communication channels, such voice channel, data channel and control channel. The flow of information between the clients of this component and the radio device is in two senses. On one hand, some clients may ask to send a message, which is coded and sent through the serial port to the radio device. On the other hand, other clients may register in the component to be notified when a message arrives, and to receive it decoded.
The choice between different radio devices depends on different factors such as price and functionality. We consider trunking radiotelephony the best option due to its fixed cost (it is not charged by the number of calls, the working time, or the number of transmitted data). The component has been developed for a T500 voice terminal with GPS option developed by Teltronic. The integration of the GPS inside the radiotelephone reduces the dimensions and prize, and offers a better portability. In any case, different kind of radios provides different kind of capabilities, but a great number of this features are common to all radio devices and these are the operations offered by the radio component. This allows creating different instances of the radio component using different kind of radio devices, but maintaining the interface, so the same capabilities are offered device independent.

## 3.1 Radio CORBA Interface

The radio component has been built as a CORBA server. In this case, it is so important to reuse the radio device as the software component. By providing a CORBA server that controls the radio, it is possible to access to the same radio from every computer that is connected to the network. The entire interaction with the radio component is made through several CORBA interface objects, as illustrated in figure 2.

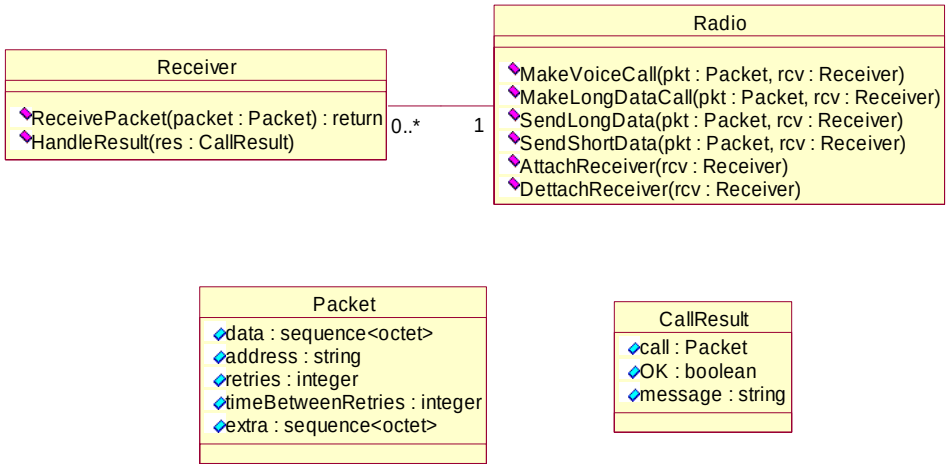


**Figure 2. Radio Interface Objects**

The *Radio* CORBA object is the interface to the radio component. It has services for making calls and registering clients for receiving incoming calls. The *Receiver* CORBA object is an interface through which the component will communicate with the client. A client must implement a Receiver object in order to use the radio component. The client can use a Receiver to interact with the radio component in two different ways. On one hand, the client may specify a Receiver when it makes a call, in order to get the result of that call through the Receiver. On the other hand, a client may register a Receiver in the component to receive any incoming message. The client can implement the Receiver to do whatever is necessary when this information arrives. In this way, any component or application which implements a Receiver can be a client of the radio component.