# TerraLib: The Architecture of an Open Source GIS Library

**Gilberto Câmara[1], Antônio Miguel Vieira Monteiro[1], Ricardo Cartaxo Modesto de Souza[1], Lúbia Vinhas[1], Karine Reis Ferreira[1], Juan Carlos Pinto de Garrido[1], Gilberto Ribeiro de Queiroz[1], Marcelo Tílio Monteiro de Carvalho, Marco Antônio Casanova[2], Ubirajara Moura de Freitas[3]**

[1]Image Processing Division – National Institute for Space Research (DPI/INPE)

[2]TECGRAF – Catholic University of Rio de Janeiro (TECGRAF-PUC-RIO)

[3]GIS Division – Research Foundation for Space Technology (GEO/FUNCATE)

{gilberto, miguel, cartaxo, lubia, karine, juan, gribeiro}@dpi.inpe.br
{tilio, casanova}@tecgraf.puc-rio.br, bira@geo.funcate.br

***Abstract.*** *This paper describes the architecture of TerraLib, and open source library for handling spatio-temporal data types in object-relational DBMS.*

## 1 Introduction

Geoinformation technology (GI) denotes the branch of information technology that develops and deploys systems for handling spatial data. GI includes geographical information systems (GIS), global positioning systems (GPS), location-based services, and remote sensing image processing. GI is a key technology for Brazil, given its vast range of applications in areas such as environmental protection, urban management, agricultural production, deforestation mapping, public health assessment, crime fighting, and socioeconomic measurements. In recent years, this area is undergoing substantial change, due to increased availability of database management systems (DBMS) that can handle spatio-temporal data types. In order to allow the open source GIS community to benefit from this opportunity, the authors are developing *TerraLib***,** a software library that supports quick development of custom-built applications using spatial databases. TerraLib also allows the development of GIS prototypes that include recent advances in spatial information science, such as spatio-temporal data models(Erwig et al. 1999), spatial statistics (Anselin 1999) and environmental modelling (Burrough 1998).

## 2 The Architecture of TerraLib

A crucial design decision in any open source project is its software architecture. All successful open source products such as Linux, PostgreSQL and Apache have a kernel whose maintenance is the responsibility of a very small team. The kernel has to be stable and to allow easy extension without internal changes. Contributions from the community occur at the external layers of the system. As an example, out of more than 400 developers, the top 15 programmers of the Apache web server contribute 88% of added lines (Mockus, Fielding, and Herbsleb 2002).

However, not all software products have well-defined paradigms as in the case of operating systems and DBMS. The challenge in GI technology is to combine object-relational DBMSes that handle spatial data structures with modules that perform functions such as spatial analysis and dynamic modeling. A typical GI application consists of three steps: (a) querying the spatial database; (b) manipulating the query results to generate new objects; (c) visualizing the resulting objects. These three capabilities (query, manipulation and visualization) should all be part of a generic GI software library. To respond to these requirements, TerraLib has four main components, as shown in Figure 1:

- **kernel**: the core of TerraLib is composed of: (a) spatio-temporal data structures; (b) support for cartographic projections; (c) topological and directional spatial operators; (d) an API for storage and retrieval of spatio-temporal objects in object-relational DBMS; (e) classes for controlling visualization of spatial data. Kernel maintenance and upgrade is the responsibility of the project core team (INPE and TECGRAF), as typical for other free software projects.
- **drivers**: modules that implement the kernel's generic database API to access DBMS products such as mySQL and PostgreSQL, and external files in both open and proprietary formats. Basic maintenance and upgrade is the responsibility of the project core team, but there is a large scope for external contribution.
- **functions**: algorithms that use the kernel structures. Typical functions include spatial analysis and query and simulation languages. The functions are designed to allow external contribution.
- **interfaces**: consist of different interfaces to the TerraLib components, to allow software development in different environments (C++, Java, COM) and also the support for Open GIS services such as WMS (Web Map Services) protocol.

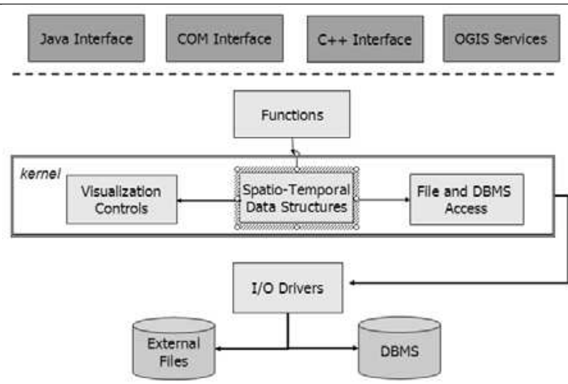


Figure 1 – TerraLib Software Architecture

## 2.1 The TerraLib Kernel

Some of the key issues involved in the design of the TerraLib *kernel* include:

- *Support for spatio-temporal applications*: In order to handle different types of spatio-temporal applications, TerraLib includes different types of spatio-temporal data types (*events, mobile objects, and evolving regions*) and supports spatio-temporal queries. Each spatial object has a unique and persistent identifier and a spatial location. The descriptive, spatial and temporal components of geographical objects are stored separately on the database and are linked by the object's id.

- *New types of data structures:* Models of spatial processes, including patterns of land use and land cover, socioeconomic and demographic characteristics, are becoming increasing important for geographical applications. To support these models, TerraLib includes *cell spaces* in a GIS database environment. This allows is an important improvement over current GIS technology, where most dynamical models based on cell spaces have a very loose coupling with a GIS. In *TerraLib*, we have implemented support for dynamic models using cell spaces, which generalize cellular automata by providing multiple attributes and flexible neighborhood definitions (Pedrosa et al. 2002)..
- *Efficiency in large-scale applications:* emerging GIS applications require storage and retrieval of hundreds of thousands of spatial objects, as well as very large satellite imagery. TerraLib supports spatial indexing of both raster and vector data, including multi-resolution and block segmentation techniques for handling satellite imagery (Vinhas, Souza, and Câmara 2003).

### 2.2 The TerraLib Drivers

The interface between the *kernel* and the various DBMS and file formats is done by the modules contained in the *drivers* component of TerraLib. When designing this component, we considered that the library should be able to handle different object-relational databases management systems transparently. This has required careful design of a DBMS interface that handles specific features of each DBMS and allows a single software interface for different systems such as PostgreSQL and mySQL (Ferreira et al. 2002).

### 2.3 The TerraLib Functions

The interface between the *kernel* and the *function* components has been designed to allow maximum extensibility. TerraLib should be extensible by other programmers, and the introduction of new algorithms and tools should not affect already-existing code. To achieve extensibility, we have adopted the principles of *generic programming*. *TerraLib* uses iterators over spatial data structures to decouple algorithms from data structures, thus simplifying extensibility. For example, for computing a histogram it is not essential to know if the elements are organized as a set of points, a set of polygons, a grid or an image. All that is needed is the ability to look into a list of values, and to obtain, for each element of the list, its values and the indexes of the elements of the list that satisfy a certain property (for example, those that are closer in space than a specified distance). In a similar way, a large number of spatial analysis algorithms can be abstracted away from a particular data structure and described only in terms of their properties. (Vinhas et al. 2002).

### 2.4 Programming Interfaces

The straightforward way to use TerraLib to develop GIS applications is to embed its C++ classes into a new system. However, the TerraLib designers have to take into account the diverse qualifications and preferences of the GIS programming community. Interfaces for Java and COM programming environments have been designed and are currently being implemented. An interface for the Open GIS web services specification is also under development.

### 2.5 Project Status: May 2004

The TerraLib version 3.0 has been released on early May 2004, with an investment of 40 man-years of work. It comprises 95.000 lines of code in C++, plus 195.000 lines of code of third-party libraries. The software is available at the website http://www.terralib.org. There are already many different applications that use TerraLib, including:

- *TerraView* is a tool for spatial data analysis, which provides the basic functions of data conversion, visualization, exploratory spatial data analysis, spatial statistical modelling and spatial and non-spatial queries. This product is being used as general visualization tool for TerraLib databases, and its source code is available together with TerraLib.
- *InfoPAE* is an automated system designed to improve the response to emergency situations for the petroleum industry. InfoPAE is developed by TECGRAF/PUC-RIO, under contract from PETROBRAS. Current plans include implementing InfoPAE at nearly 80 installations throughout 2003.
- *SIGMUN* is a system for handling cadastral applications in metropolitan areas, used by the cities of Santos, São Sebastião, Caraguatatuba, São José dos Campos, São Bernardo do Campo, Cachoeiro de Itapemirim, Vitória, and in 30 cities of the state of Bahia .

## References

Anselin, Luc. 1999. Interactive techniques and Exploratory Spatial Data Analysis. In *Geographical Information Systems: principles, techniques, management and applications*, edited by P. Longley, M. Goodchild, D. Maguire and D. Rhind. Cambridge: Geoinformation International.

Burrough, P. 1998. Dynamic Modelling and Geocomputation. In *Geocomputation: A Primer*, edited by P. Longley, S. Brooks, R. McDonnell and B. Macmillan. New York: John Wiley.

Erwig, Martin, Ralf H. Güting, Markus Schneider, and Michalis Vazirgiannis. 1999. Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases. *GeoInformatica* 3 (3):269-296.

Ferreira, Karine Reis, Gilberto Queiroz, Joao Argemiro Paiva, Ricardo Cartaxo Souza, and Gilberto Câmara. 2002. Arquitetura de Software para Construção de Bancos de Dados Geográficos com SGBD Objeto-Relacionais. Artigo apresentado em XVII Simpósio Brasileiro de Banco de Dados, em Gramado, RS.

Mockus, Audris, Roy Fielding, and James Herbsleb. 2002. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology* 11 (3).

Pedrosa, Bianca, Gilberto Câmara, Frederico Fonseca, and Ricardo Cartaxo Modesto de Souza. 2002. TerraML - A Cell-Based Modeling Language for an Open-Source GIS Library. Artigo apresentado em II International Conference on Geographical Information Science (GIScience 2002), em Boulder, CO.

Vinhas, Lubia, Gilberto Ribeiro de Queiroz, Karine Ferreira, Gilberto Câmara, and João Argemiro Paiva. 2002. Programação Genérica Aplicada a Algoritmos Geográficos. Artigo apresentado em IV Simpósio Brasileiro de Geoinformática, em Caxambu.

Vinhas, Lubia, Ricardo Cartaxo Modesto de Souza, and Gilberto Câmara. 2003. Image Data Handling in Spatial Databases. Artigo apresentado em V Simpósio Brasileiro de Geoinformática, em Campos do Jordao.