The Architecture of the CommonGIS System (ESPRIT Project 28983)

Natalia Andrienko[1], Gennady Andrienko[1], Hans Voss[1], and Frank Tuijnman[2]

[1] GMD – German National Research Center for Information Technology,
AiS.KD – Knowledge Discovery research group,
http://ais.gmd.de/KD/
[2] PGS – Professional Geo Systems
Amsterdam, Netherlands
http://www.pgs.nl/

Abstract. The paper describes the object-oriented architecture of the first prototype of the system developed within CommonGIS project. The system combines knowledge-based map design, interactive cartographic displays, distributed database access, and efficient Internet GIS functionality.

1 Introduction

The project CommonGIS[1] started in October 1998 with the goal to develop an intelligent system supporting end users in visual analysis of thematic data in a geographic context. The primary focus of the project is automated visualisation of spatially referenced data. The partners in the project are 3 research institutes (GMD[2], IGD[3], and JRC[4]), 2 commercial software providers (PGS[5] and Dialogis[6]), national center for geoinformation (CNIG[7]), and international GIS association (GISIG[8]). The project is funded by an ESPRIT programme (thematic call "Information Access and Interfaces", project 28983), the duration is 30 months. Within the project the existing system for knowledge-based visualisation design Descartes [1] should be further elaborated and combined with the commercial Internet GIS, Lava/Magma [2].

2 Current state: Lava/Magma and Descartes

2.1 Rationale for the integration

The first prototype of the CommonGIS software system is created on the basis of two existing systems, Lava/Magma and Descartes. Both systems are designed to provide mapping facilities in WWW. The main difference between the systems is that Lava/Magma is oriented mostly to presentation of geographical features whereas the focus of Descartes is visualisation of thematic (attribute) data associated with geographical objects or locations. The purpose of the integration is to combine advantageous and unique features of the systems. These features are:

- Lava/Magma:
  - opportunity to work with distributed data stored on multiple WWW servers;
  - possibility of connection to various relational databases;
  - efficient work with large databases due to optimisation of access to data;
  - relatively easy extension through adding user-defined types of spatial objects.
- Descartes:
  - large variety of visualisation techniques for attribute data associated with spatial objects;
  - fully automatic design of thematic maps with intelligent selection of visualisation techniques according to characteristics of data and relationships among data components;

[1] CommonGIS project, URL http://commongis.jrc.it/
[2] GMD - German National Research Center for Information Technology, http://ais.gmd.de/KD/
[3] IGD – Fraunhofer Institute for Computer Graphics, http://www.igd.fhg.de/
[4] JRC - Joint Research Center of European Commission, http://www.jrc.org/
[5] PGS - Professional Geo Systems, http://www.pgs.nl/
[6] Dialogis Software and Services GmbH, http://www.dialogis.com/
[7] CNIG - Portuguese National Center for Geographic Information, http://www.cnig.pt/
[8] GISIG – European GIS Association, http://gisig.ima.ge.cnr.it/

- interactive, dynamic maps, i.e. maps changing their appearance in response to user's manipulation. The manipulation devices are designed so as to support visual exploratory analysis of data represented on the maps.

The interface parts of both systems are implemented in Java, which facilitates their integration. Considered below are functions and architectures of both systems.

2.2 Lava/Magma

2.2.1 Function. The main function of Lava/Magma is to display a map on a computer screen and to support basic zooming, panning, and translation operations. A map consists of a number of layers drawn in a sequence that may be specified by the user. A layer is a collection of geographical objects of the same type and meaning. For example, there may be a layer of roads having the type polyline. Usually objects belonging to the same layer are drawn with the use of the same colour, line or filling style, etc.

Besides displaying maps, Lava/Magma gives the user facilities to change visual properties of layers (colours, line or filling styles, whether labels are drawn or not, etc.) and the order of their drawing and to switch on/off drawing of individual layers (see Figure 1).

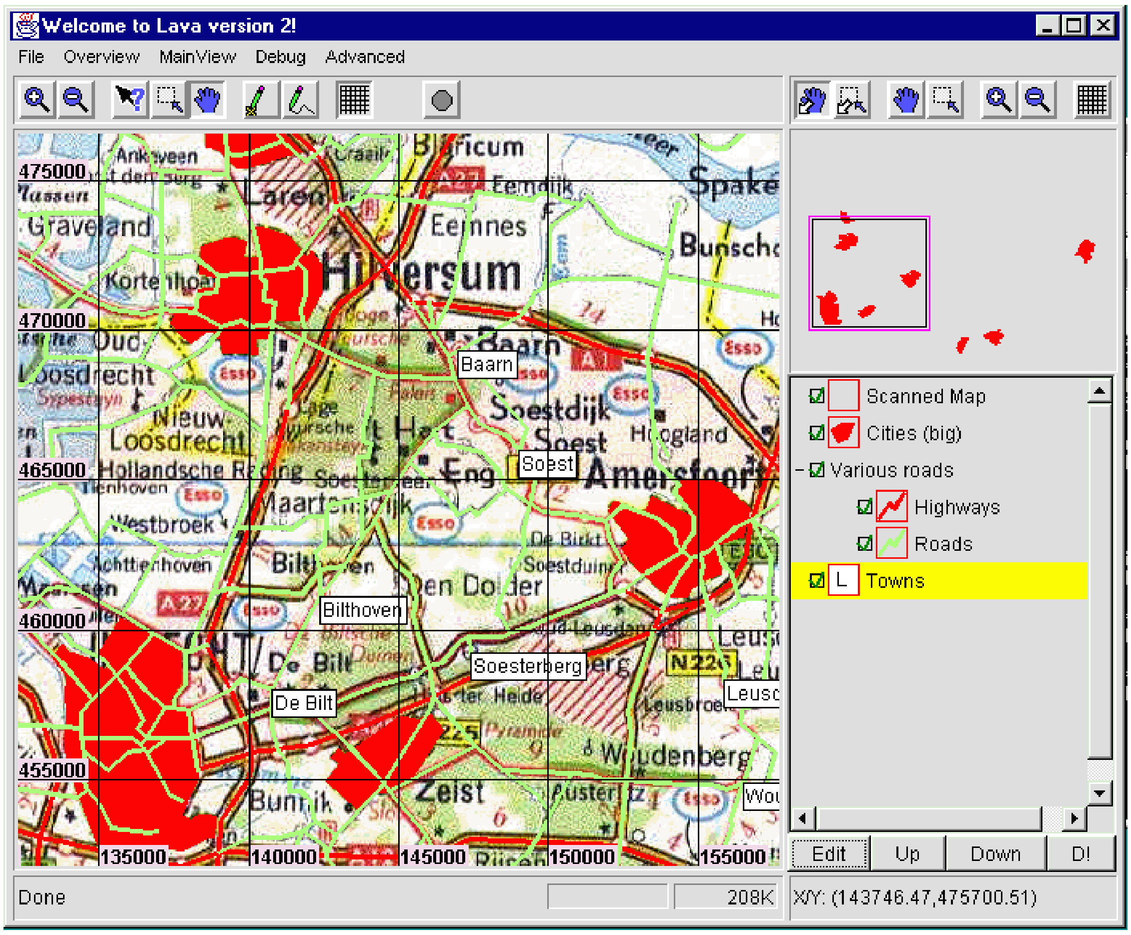


Figure 1*. The user interface of Lava/Magma*

2.2.2 Architecture. A map layer in Lava/Magma represents a relation in some relational database. Each geographical object is a graphical presentation of a tuple in this relation. Drawing of a geographical object is performed by a piece of Java code called LavaShape. An individual LavaShape is created for each geographical object to be shown in the map. There are various types of LavaShapes, according to the types of objects (point, polygon, polyline, etc.). It is possible to extend the set of object types recognised by Lava/Magma through defining a new type of LavaShape. Technically this means writing a Java class that extends the base class LavaShape and implements the desired peculiarities of drawing or/and specific reaction on mouse click.

Lava/Magma has client-server architecture. Its server, Magma, works on top of a relational database and supplies data to the client Lava. Lava is implemented completely in Java and therefore can be started from a WWW page viewed in some Internet browser. It includes two main parts called MapView and Vulcan.

MapView displays the map and provides the user interface facilities. Vulcan is a small relational database working on the client side. It acts as a data cache to optimise data transfer between the server and the client. MapView addresses all requests for data to Vulcan. Vulcan checks whether the requested data tuples are present in the cache. Only if they are absent it redirects the request to Magma and gets the data tuples from the server. For each data tuple to be sent to MapView Vulcan creates a LavaShape responsible for drawing of corresponding geographical object. So, MapView receives from Vulcan a collection of LavaShapes ready to draw. Constructing a LavaShape, Vulcan passes it the data necessary for its functioning, according to settings in a special configuration file (see Figure 2).

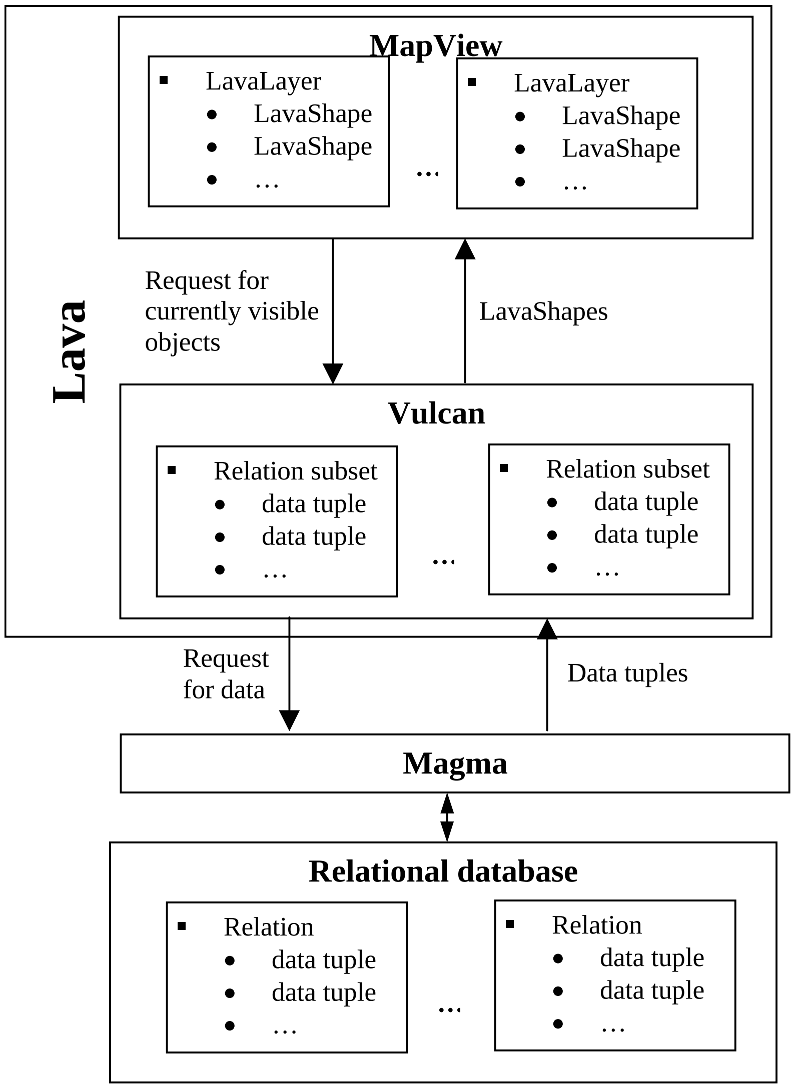


Figure 2. *The software architecture of Lava/Magma*

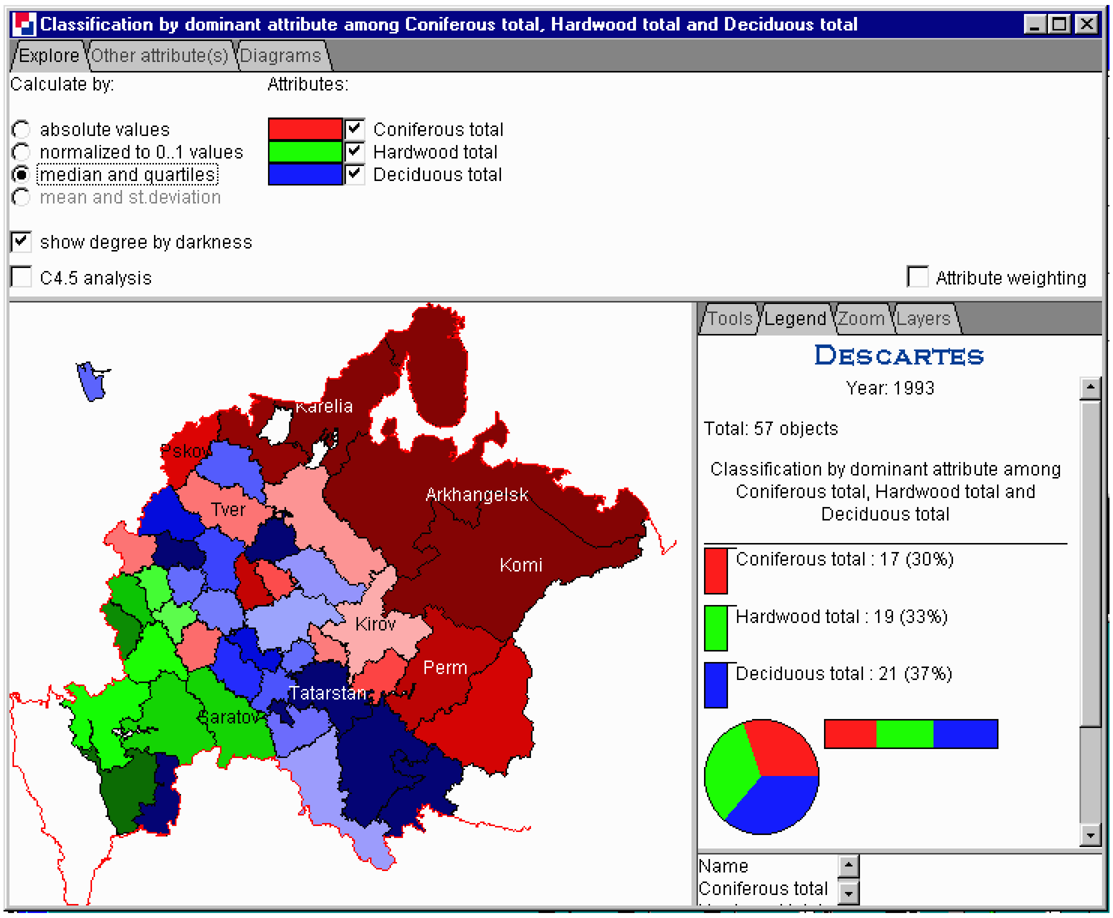
2.2.3 Some details about LavaShapes. A LavaShape is an instance of a Java class that should be a descendant of the base class LavaShape. The methods defined in this Java class can be divided into 2 groups: (1) initialisation of internal variables and (2) behaviour functions. Vulcan calls functions of the first group in order to pass to instances of the LavaShape the data necessary for their functioning. These may be, in particular, values retrieved from the corresponding tuples in the database. The second group of methods provides drawing of corresponding objects on the map, handling mouse clicks on these objects, and changing visible properties of the objects after user's manipulations.

2.3 Descartes

2.3.1 Function. Descartes is a system for automated visualisation working over a kind of domain model called "notion base". The notion base enumerates notions of an application domain essential for interpretation of data to visualise. Important relationships among the notions are indicated. For example, the system can "understand" that female and male are parts of the whole population, and that female from 0 to 14 years, female from 15 to 64 years, and female of the age 65 years and more make together female population in total. This kind of knowledge allows grounded selection of particular presentation techniques such as maps with pie charts or segmented bars.

Like Lava, Descartes is also able to display maps consisting of multiple layers. However, such a map always contains one layer drawn in a special way: geographical objects belonging to this layer look differently (e.g. painted in different colours) depending on attribute data associated with them. Descartes can present attribute data in maps using various methods of thematic cartography (see Figure 3).

The main functionality of Descartes can be described as the following scenario. The user selects one or more attributes of geographical objects to be visualised in a map. Examples of attributes are “population number”, “gross national product”, “membership in European Union”, “dominant religion”, etc. defined for European countries. Depending on the number of selected attributes, their types and some other characteristics, and relationships among them the system automatically selects one or more suitable cartographic presentation techniques. For each of the techniques Descartes creates a map implementing it. The user can view any of these maps, like a map in Lava/Magma. In addition, s/he can do various manipulations changing the visual appearance of the map in real time. For this purpose almost each map displayed in Descartes is supplemented with specially designed interactive facilities. Which facilities are applied to a map and what effects they produce on its appearance depend on the visualisation technique implemented in the map. The role of the tools for map manipulation is to support visual analysis of the represented data.



Classification by dominant attribute among Coniferous total, Hardwood total and Deciduous total

Calculate by:

- ( ) absolute values
- ( ) normalized to 0..1 values
- (●) median and quartiles
- ( ) mean and st.deviation

Attributes:

- 🟥 ✓ Coniferous total
- 🟩 ✓ Hardwood total
- 🟦 ✓ Deciduous total

✓ show degree by darkness

☐ C4.5 analysis

☐ Attribute weighting

Year: 1993

Total: 57 objects

Classification by dominant attribute among Coniferous total, Hardwood total and Deciduous total

- 🟥 Coniferous total : 17 (30%)
- 🟩 Hardwood total : 19 (33%)
- 🟦 Deciduous total : 21 (37%)

Figure 3. A screenshot of Descartes

2.3.2 Architecture. To present a geographical object in a map, Descartes needs two kinds of data: (1) geometry, i.e. whether the object is represented by a point, a polyline, or a polygon, and co-ordinates of the point or vertices of the polygon; (2) values of the user-selected attributes associated with this object. Geometry is stored separately from attribute information. The association between them is done through identifiers: a unique string identifier is assigned to each geographical object in a layer, and each tuple of attribute data should contain the identifier of the object the data refer to.

Attribute data for Descartes are stored in tables. Tables in Descartes are equivalent to relations in Lava/Magma. The only difference is that Descartes does not use any database management system and performs all data retrieval operations by itself. For the sake of unification, we will further use the terminology of Lava/Magma.

For automatic data mapping Descartes needs some knowledge about the available relations like semantic relationships among the attributes. Even if a relational database were used, it would be impossible to get such knowledge from the database schema. Therefore this information is provided additionally. To view and analyse data in Descartes, one should create a Descartes *application* that contains a *semantic description* of the relations to work with.

Like Lava/Magma, Descartes also has client-server architecture (Figure 4). The server reads and interprets the semantic description of the database, performs map design, and does all operations with data. The client written in Java provides the user interface for selection of data to be visualised, displays the generated maps, and supports their manipulation. Selection of data includes:

- selection of a relation (table) if the application contains several relations;
- selection of attributes in the chosen relation;
- construction of a query to allow visualisation of only those data tuples that satisfy certain conditions;
- specification of an arithmetic formula to derive a new attribute by calculations over existing attributes.

The server of Descartes consists of the following principal blocks: (1) data semantics management, i.e. reading and interpretation of the semantic description of data, including logical inferences; (2) data management; (3) map design.

The client and the server keep a permanent contact with each other communicating through a socket connection. When data or performing of some function is required from the server, the client sends a command and receives in response the results of its processing.

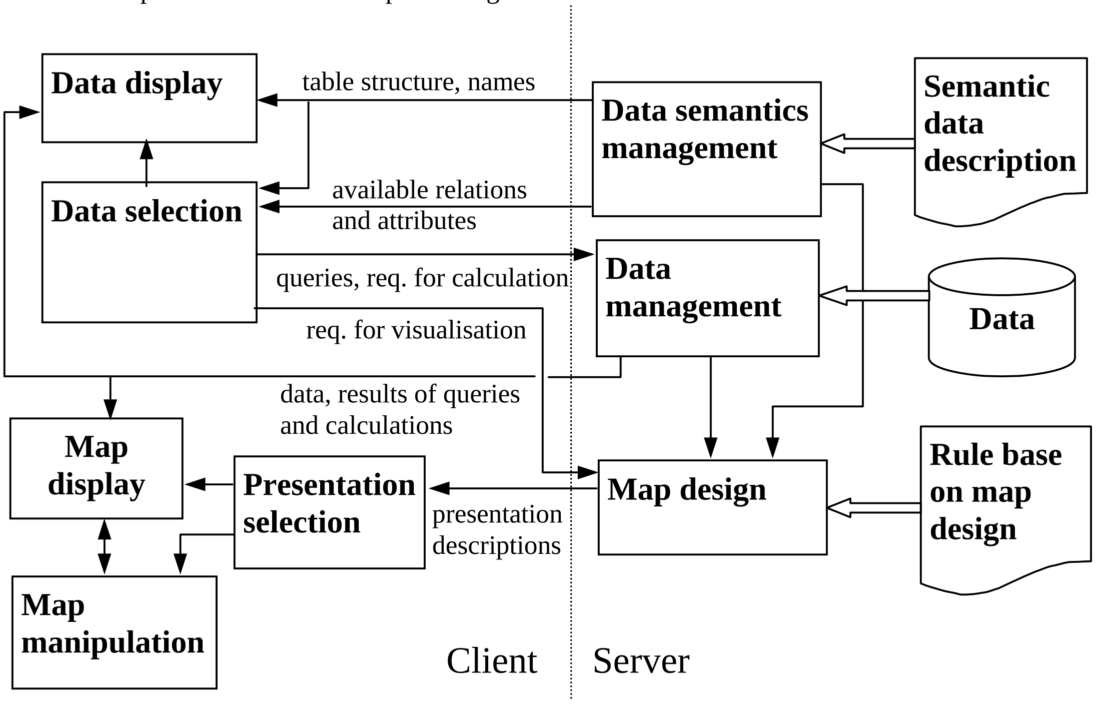


Figure 4. *The architecture of Descartes*

3 Integration

3.1 Scenario

Lava/Magma differs from Descartes in that the former can be characterised as map-centred whereas the latter as data-centred. This difference is very well seen from the scenarios of use supported by the systems. Starting Lava/Magma, the user immediately receives on the screen a map and then can perform operations on it. In Descartes the user receives a table with attribute data. Only after data selection (see §2.3.2) s/he gets a map presenting the selected data.

In the integrated system we have chosen to realise the map-centred scenario, like in Lava/Magma. Starting the system, the user receives a map display. Only geographic information is shown on the map initially, no attribute data. So, this is a usual Lava/Magma map. However, for some of the layers attribute data may exist in the database. We shall further call such layers DescartesLayers.

As was described above, in Lava/Magma the user can change certain visual properties of layers. For a DescartesLayer the user may instead select attributes (or, more generally, attribute data) to be visualised. In other words, when the user activates the function of changing properties on a DescartesLayer, the data selection interface appears. This interface is taken from Descartes.

After the data to visualise are selected, the map design component of Descartes finds suitable visualisation techniques. If there are several appropriate presentations of the data, the system proposes the user to select the one s/he prefers. The variants are represented by icons indicating the visualisation method used and by brief text descriptions, like in current Descartes. After the user has selected the presentation to view, the system redraws the map. The DescartesLayer for that the visualisation of attribute data was requested is drawn in a special way: visual properties of each geographical object are determined by the data associated with it and the visualisation technique chosen. Thus, the objects may be painted in different colours or shades, or different diagrams may be drawn at the locations of the objects.

From now on, when the user again wishes to change the visual properties of the DescartesLayer, s/he may either do data selection, as described above, or choose from previously generated variants of presentation that are stored.

A map with presentation of attribute data is viewed using the currently existing facilities for zooming, panning, and shifting. Besides, a window with manipulation tools for the DescartesLayer appears. These are the currently existing in Descartes tools to support visual exploration of attribute data. Operations with these tools affect only the visual appearance of the corresponding DescartesLayer.

3.2 Implementation

To implement drawing of geographical objects depending on values of selected attributes associated with them, we create a special kind of LavaShape further called DescartesShape. More exactly, we need two types of DescartesShapes, one for drawing point objects and another for area objects. However, both these types will be further referred to as DescartesShape. From the programming point of view, these will be classes extending the base class LavaShape and implementing a common interface DescartesShape.

A DescartesLayer consists of DescartesShapes. Since DescartesShape is going to be a descendant of the base class LavaShape, it will have all the functions defined in LavaShape though the implementation of some of them will be redefined. Therefore MapView, the component of Lava responsible for map display, will be able to treat DescartesShapes just as usual LavaShapes. The operations of switching on/off drawing and changing the sequence of drawing that are available for LavaLayers will be equally applicable to DescartesLayers.

As compared to a LavaShape, a DescartesShape will need more data to be passed to it on the stage of construction, besides co-ordinates of points and some other information needed to draw the geographical object itself. These additional data are:

- values of the attributes associated with this object;
- which visualisation technique is currently applied;
- necessary settings for the current technique, such as the coefficient to transform numeric values of an attribute into heights of bars.

Values of the attributes are obtained from Vulcan. All other information comes from the components of Descartes depending on results of visualisation design and user’s selection of variant of presentation. Settings for the current technique may be changed in result of user’s actions on the map manipulation tools. The map manipulation tools should be able to transmit the user-induced changes to all currently visible DescartesShapes.

The function of drawing of the class DescartesShape is implemented so that visual appearance of an object is determined by the associated values of the selected attributes and the current visualisation technique and its settings. The function of reaction on mouse click is defined so that the map manipulation tools are notified (after this some change of settings may be done as in the case when the user manipulates the tools themselves).

3.3 Architecture

The integrated system includes all components of Lava/Magma. The data management part of Descartes is no more needed because Magma and Vulcan perform nearly all functions that are necessary (however, an extension is needed to make them able to provide some statistics about data: number of different values, minimum, maximum, average etc.). Lava’s MapView replaces the map display component of Descartes. The data display module (table view) is, in principle, unnecessary for visual analysis of spatially referenced data. Therefore it can be eliminated. The remaining components of Descartes together with those of Lava/Magma form the architecture shown in figure 5.

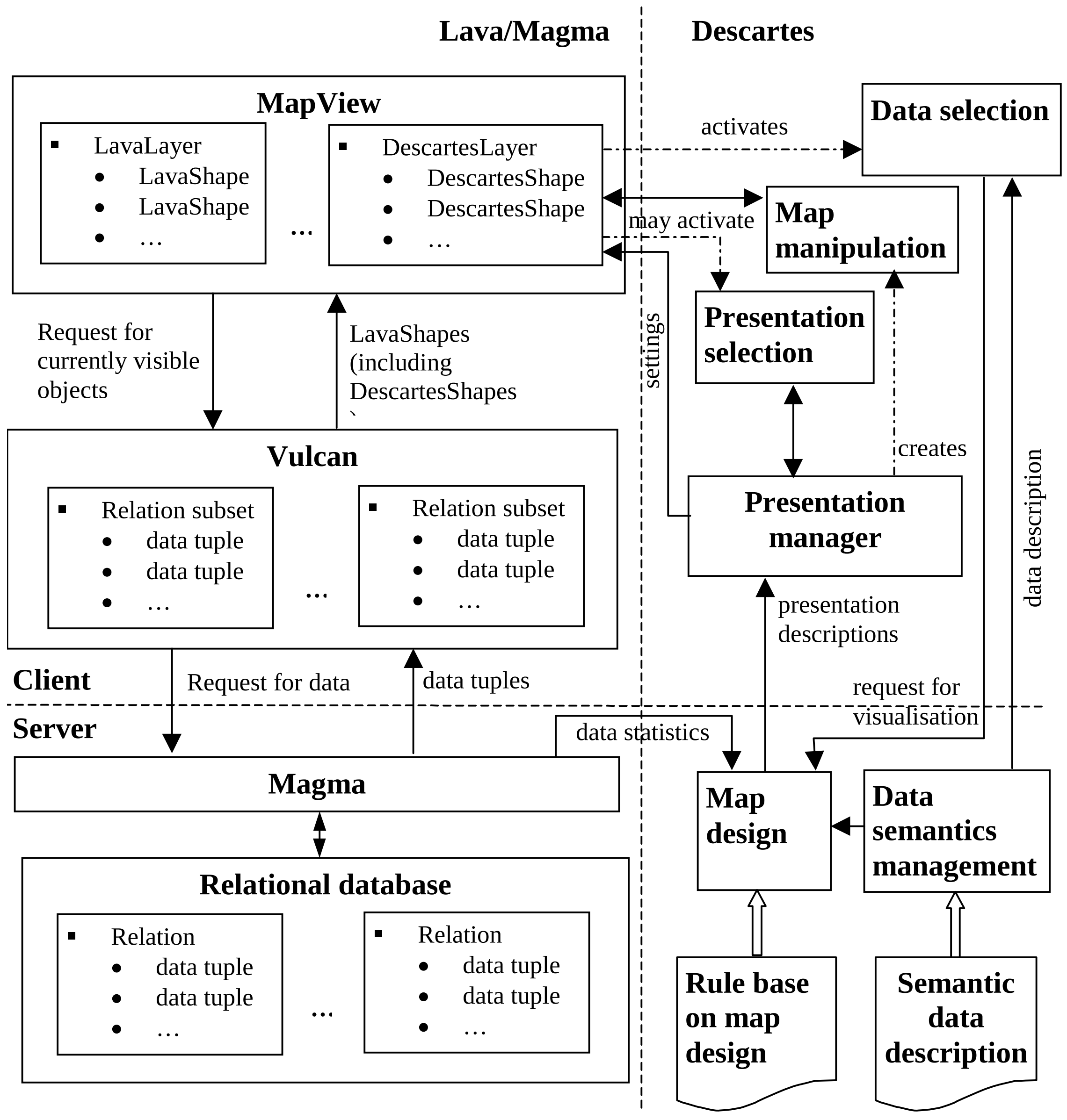


Figure 5. *The architecture of the integrated system*

In this scheme, the bottom part including Magma, the map design module and the data semantics manager, is the server part of the system, and the rest of the components constitute the client part. The upper modules provide user interface, and the middle part contains non-interface client components. The left part of the scheme includes components from current Lava/Magma, and on the right are blocks from Descartes.

The new component, Presentation Manager, is a non-interface component, the tasks of which are:

- to receive, from the map design module, descriptions of presentation variants and store them;
- to transmit these descriptions to the presentation selection module;
- depending on the current selection of the presentation variant received from the presentation selection module makes corresponding settings in DescartesShapes.

To implement this architecture, we had to make some extensions to the existing systems:

- The data server Magma needs to be extended to be able to provide data statistics:
  - for qualitative data: number of different values of the attribute;
  - for numeric data: minimum, maximum, average, median, quartiles.
- All drawing of attribute data currently implemented in Descartes are to be encapsulated in DescartesShapes.
- Map manipulation facilities of Descartes, in addition, should show the legend regarding the presentation of attribute information.

- Legend of MapView (Lava/Magma) was extended with a button activating data selection and presentation selection dialogs.
- The client and the server of Descartes are being reconfigured according to the new architecture

Figure 6 shows the user interface of the integrated system. It is built on the basis of map-centred metaphor. A user can activate attribute mapping of Descartes by selecting a DescartesLayer in the legend window of MapView. After selection of data and presentation method through dialogs taken from the user interface of Descartes the integrated system displays the thematic map inside the map display window of Lava/Magma. Additionally, dynamic manipulation controls and, possibly, non-cartographic statistical displays appear. The controls allow visual exploration of geographically referenced attribute data within the system.

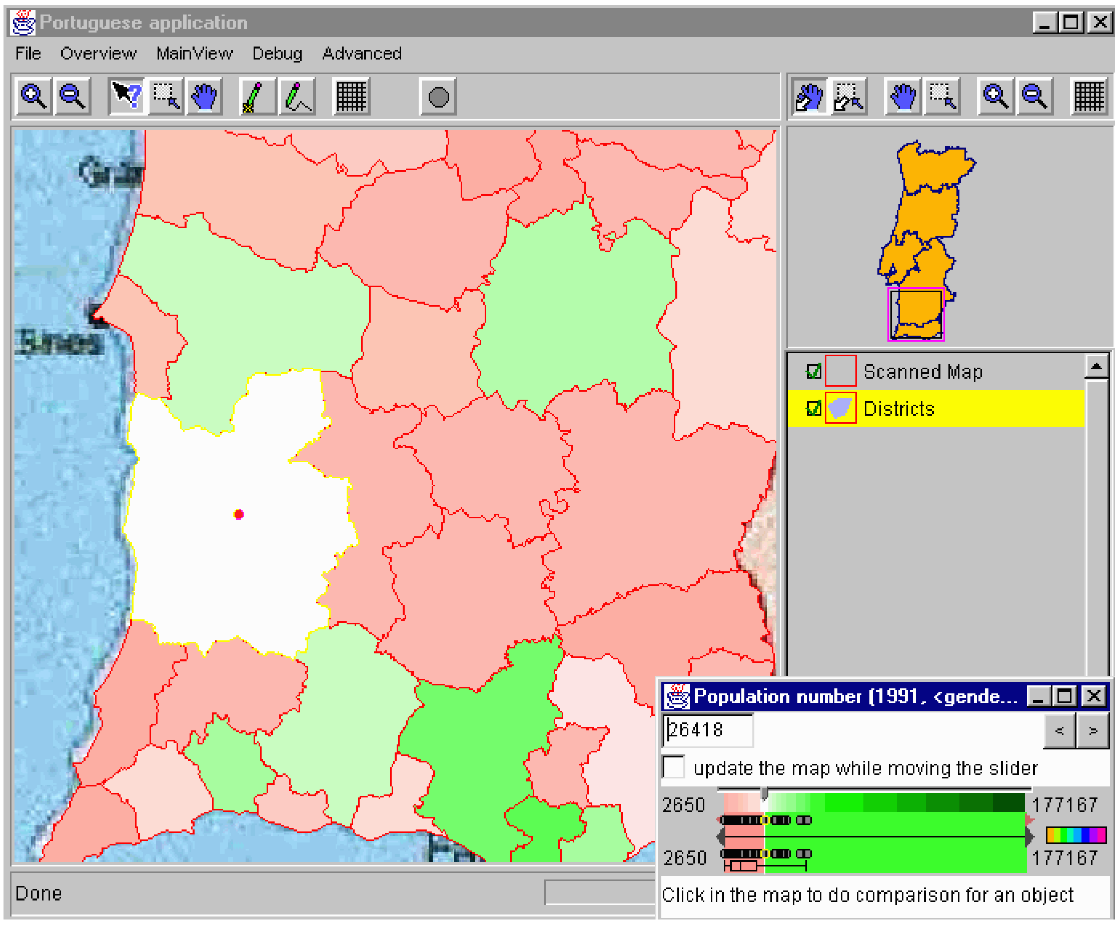

Portuguese application
File
Overview
MainView
Debug
Advanced
Scanned Map
Districts
Population number (1991, <gende...
26418
update the map while moving the slider
2650
177167
2650
177167
Click in the map to do comparison for an object
Done

Figure 6. *A screenshot of the user interface of the integrated system*

4 Conclusion

The proposed architecture is being implemented mostly by decomposing Descartes into parts to be incorporated into Lava/Magma. Generally, this decomposition is reusable. It will be relatively easy to link the “new Descartes” with other GIS software having open object-oriented architecture and allowing extensions.

The integration allows us to couple unique features of Descartes (knowledge-based map design and interactive displays) with highly efficient Internet mapping capabilities of Lava/Magma.

References

1. Andrienko, G. and Andrienko N. “Interactive Maps for Visual Data Exploration”, *International Journal Geographical Information Science*, Vol.13, no.4, pp.355-374, 1999
2. van den Berg, C., Tuinman, F., Vijbrief, T., Meijer, C., van Oosterom, P., and Uitermark, H. “Multi-server Internet GIS: Standardization and Practical Experiences”, In Goodchild, M., Egenhofer, M., Fegeas, R., and Kottman, C. (eds.) *Interoperating Geographic Information Systems*. Boston: Kluwer Academic Publishers, pp.365-377, 1999