Statistical nature resource analysis based on the modern spatial server technology

Kainulainen, S.

Forest and Park Service, Natural Resources Unit, Vernissakatu 4, FIN-01300 Vantaa, Finland, simo.kainulainen@metsa.fi

Abstract

Forest and Park Service is surveying during 1996–1998 biotopes and geomorphology in the northernmost Lapland. The complete data set will contain nearly one million spatial features, most of these are biotope regions covering about 2,5 million hectares of the land area administered by the Forest and Park Service. Along the nature survey the FFPS is designing and building a modern spatial information system and DBMS to manage surveying information of both spatial and non-spatial features of northern nature. Because standard RDBMS's does not support neither complex spatial data types or topological relations between objects, and a data volume of a normal spatial database is very large compared to traditional operational, modern spatial server technologies have been developed to avoid those problems: spatial middleware has a functionality to model and store complex spatial information as well to index the data spatially using e.g. R-Trees. Spatial SQL3-functions cover data retrieval within a spatial aspect, various geometric overlays and metric calculations which can be used directly from the client applications with no spatial interface.

Keywords: Forest resources, arctic environment, spatial server technology, ecosystem mapping, spatial statistics

1 Inventory systems pilot of Forest and Fark Service

1.1 Nature survey of the northernmost Lapland

Finnish Forest and Park Service (Metsähallitus, the FPS) is surveying biotopes and geomorphology during 1996–1998 in the northernmost Lapland (Fig. 1). The complete data set will contain nearly one million spatial features, most of these are biotope regions covering about 2,5 million hectares of the land area administered by Forest and Park Service, which is about 30 % of the total coverage administrated by the FPS. Along the nature survey the FPS is designing and building modern spatial information and database management systems (DBMS) to manage surveying information of both spatial and non-spatial features of northern nature. The geographical information system (GIS) development is carried as a pilot project for the 3rd generation integrated GIS in the FPS.

A biotope is a region or a group of regions having homogenous attribute values. Biotope regions are almost similar to forest stands, but much more information is collected about flora and site properties, especially on peat lands and mires. The size of a biotope polygon varies between 1 to 100 hectares. The aspect is hence to document natural resources from a view of nature protection instead of foresters.

1.2 GIS pilot project

Current 2nd generation's geographical information systems (forestry, land use) of the FPS are based on an Oracle RDBMS and a Swiss-

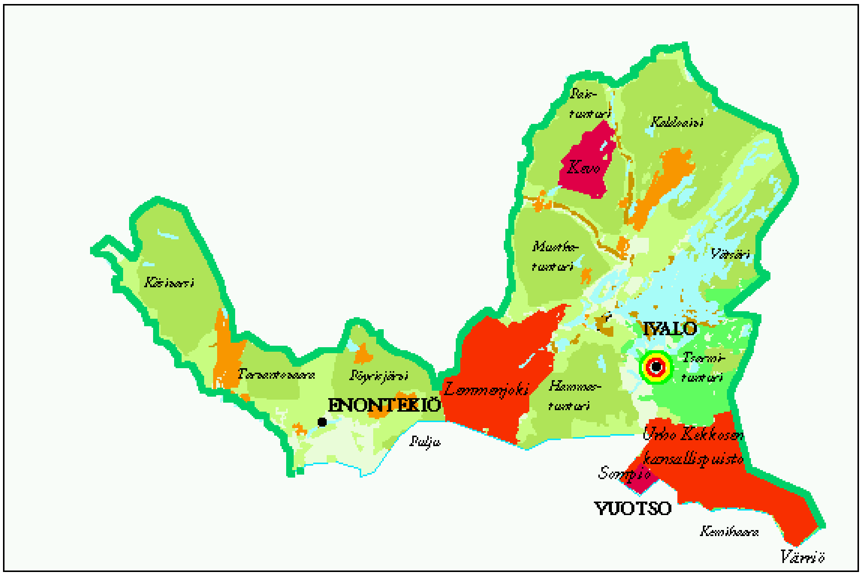

Figure 1. Nature survey area.

German product **Gradis-GIS**, which is an object oriented software for managing seamless spatial databases in a multi-user environment (unix). The system development ended in 1996.

During a product and technology survey in 1997 the FPS studied two main technologies: systems based on a heavy GIS-product (SmallWorld GIS, Xforest/GISBase) and client/server solutions based on GIS engines (MapInfo SpatialWare and ESRI's SDE). The FPS made a decision to go further with c/s-architecture.

The pilot system is based on the modern **MapInfo Spatial Ware** server technology and used clients are standard applications like **Microsoft Excel** (visualisation of simple thematic maps and tabular work sheets) and **MapInfo Professional** desk-top GIS-installations. Attribute values are handled and maintained in the form oriented **PowerBuilder**-application (a product of Sybase inc.), which is also a core for the statistical reporting.

Our goal is to produce very complex statistical reports and thematic maps driven by attribute values stored in the biotope database. It is possible to study freely defined regions of interest using the state-of-the-art spatial server technology and low cost desk-top mapping packages, such as MapInfo.

1.3 Work flow of refining field data to integrated information

The inventory survey is done by field surveys of pre-selected key stands (summer) and stereo interpretation (winter) on IR aerial images. The ratio between surveyed and interpreted biotope areas is about 15%. Biotope polygons are then drawn from aerial photos on the base map transparency (Fig. 2). A consulting

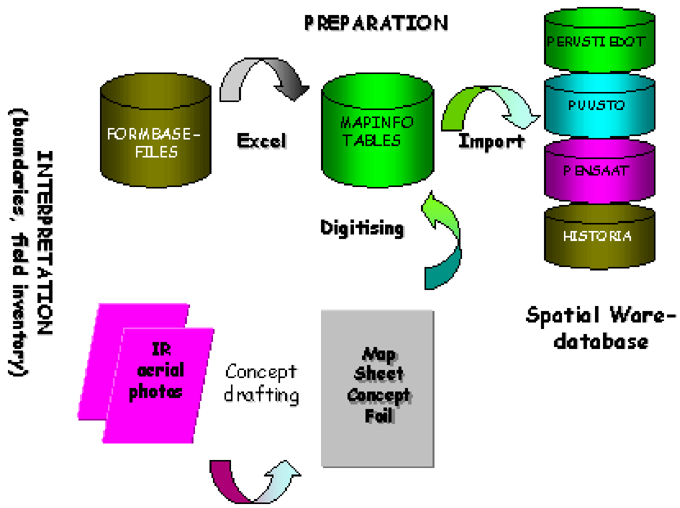

Figure 2. The work flow of data capture

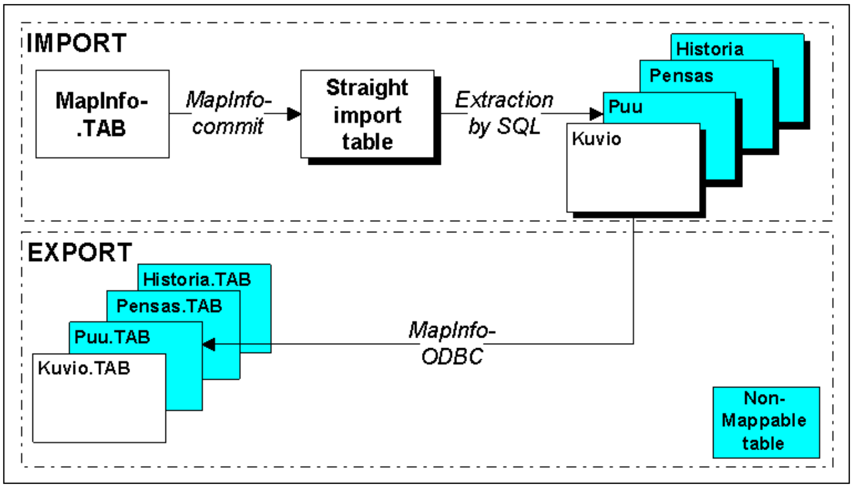

Figure 3. The Import/Export of mapping data

company is digitising map sheets (ca. 400 sheets of 10x10 km^2) into simple MapInfo databases.

The data capture environment is very low cost and it did not require much programming or design work. Attributes are entered by using a Xerox's card file software **FormBase** v1.0. FormBase databases and single map sheets can be coupled together in MapInfo for simple thematic map production or inspection of data quality.

Map databases of the first phase are not yet relational. The next task is to load tabular data into the relational environment managed by the Spatial Ware server.

The data model (Fig. 4) and a simplified representation about a data flow process (Fig. 3) are shown above. The first import table is just an image of a plain MapInfo-database table. The data is moved from MapInfo-tables by using a unix level import utility, which is part of Spatial Ware core software. After that the data is checked and loaded into four relational tables with an aid of sequence of SQL-selections. During the final import it is possible to check and repair topological inconsistencies on a polygon network.

1.4 Current status

A development team set requirements for the system about 1,5 years along field inventories and data capture. The system planning and design on a spatial server environment started in November 1997 with consulting database experts. The work was finished in January 1998, and the implementation phase took a time span of 3,5 moths, from February to May. The pilot installation is now made and it is in use until the end of August.

The captured data is in the MapInfo format and FormBase attribute tables can be read and linked directly to the map data. Linked ta-

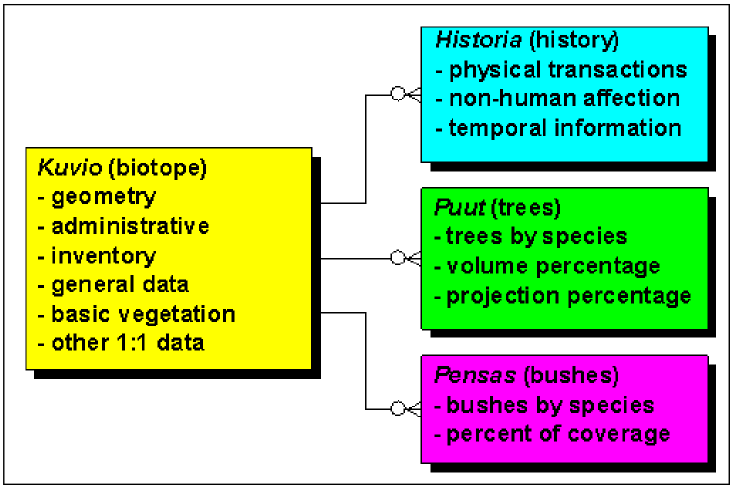

Figure 4.The relational data model of a biotope

bles are to be loaded into the centralised and seamless database. This work is in progress.

2 Introduction to the spatial server technology

2.1 Spatial Server technology

Standard RDBMSes do not support neither complex spatial data types; areas and linear features, nor topological relations between these objects. A data volume of a normal spatial database is very large compared to traditional operational databases, which reduces the performance of spatial queries and retrieval.

Modern spatial server technologies have been developed to avoid problems mentioned above: spatial middleware has a functionality to **model** and **store** complex spatial information as well to **index** the data spatially using e.g. R-Trees. Even the server application itself contains no graphics, it can perform analysis tasks by using SQL3-extensions. Spatial functions cover **data retrieval** within a spatial aspect, various geometric overlays and metric calculations which can be used directly from the client applications.

Spatial servers can be considered as standard interfacing layers between client applications and spatial databases. Servers have the full GIS-functionality except map visualisation. Separate client software is used as a map interface.

2.2 Server functionality

The server itself is one unix process per client. The server takes care about tasks described in the following chapters. Tools are delivered with a server there for system management, spatial queries and analysis (Fig. 5).

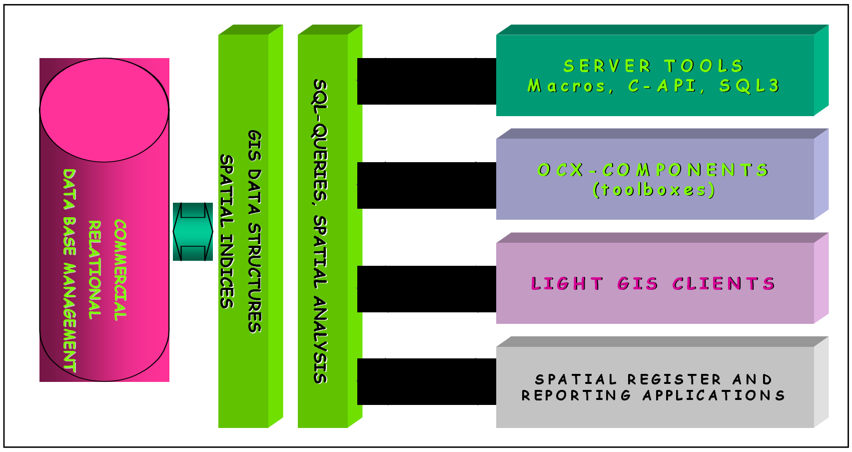

Figure 5. Server architecture

A unix level SQL3-interface (sql_spatial-program) comes with server software. This can send spatial SQL3-queries and encode geometric information. An ODBC-driver uses the same SQL-syntax than the line oriented SQL-interface.

Applications can be developed under unix with a Tcl/Tk-scripting language, which contains interfacing layers to graphics and SQL3. Part of supporting software is made by using Tcl/Tk. Other possibility on the unix level is to use C-API-library, which can be linked to applications.

2.2.1 Spatial data types

A primitive entity of the spatial data is a co-ordinate point. Higher geometric entities, such as linear and area features are constructed of point sequences. In a standard relational database the only spatial entity which can be modelled directly in the data model is a point (x, y, z). Complex geometry hierarchies must be decoded and stored into binary fields of relational tables by a server functionality. The decoded geometry data can not be used via standard database tools.

2.2.2 Spatial indexing

Even a small spatial database contains a very large volume of information compared to conventional information systems. A performance of retrieval of query results is better if a data field to be queried is indexed. Because the geometry is stored in one binary field, that can not be indexed as a normal relational 1-dimensional attribute.

The server system creates and maintains a spatial R-tree index of multidimensional geometry. With these indices the performance of spatial queries and encoding of geometry is hundreds of times better than without indexing.

2.2.3 Spatial Query Language (SQL3)

The standard SQL (Structured Query Language) queries can retrieve data rows from relational tables driven by attribute value restrictions and comparisons. The SQL3 language (SQL92 or SQL/MM Multimedia, a standard proposal of the International Standardizing Organisation, ISO and the International Electrotechical Committee, IEC) is an extension of SQL. It contains spatial functions and operators which use decoded geometric data like ordinary tabular information.

Spatial functions return either **metric values** or **spatial objects** as a result. Scalar metric values are results of calculations on geometry: *Area*(), *Length*(), *Perimeter*() and *Slope*() are Spatial Ware implementations. Functions which calculate new geometry as a result: *Adjacent*(), *Buffer*(), *Centroid*(), *Contain*(), *Geometry_union*(), *MER*(), *Overlap*() and *Skeleton*().

Special spatial comparison operators return a logical value. These are predicates between compared value formulas in the where-clause of SQL syntax. Instead of comparing just values of attributes (**a** is greater than **b**), spatial predicates use either geometric or topologic properties of spatial objects (i.e. a point is **located** inside of an specified area). Predicates supported by Spatial Ware: AT START OF, AT END OF, CONNECTED [TO], OVERLAPS, CONTAINS, [IS] CONTAINED [BY] and [IS] ADJACENT [TO].

In the Figure 6 are shown endangered species (bird and tree symbols) and biotope stands. A study area (re-

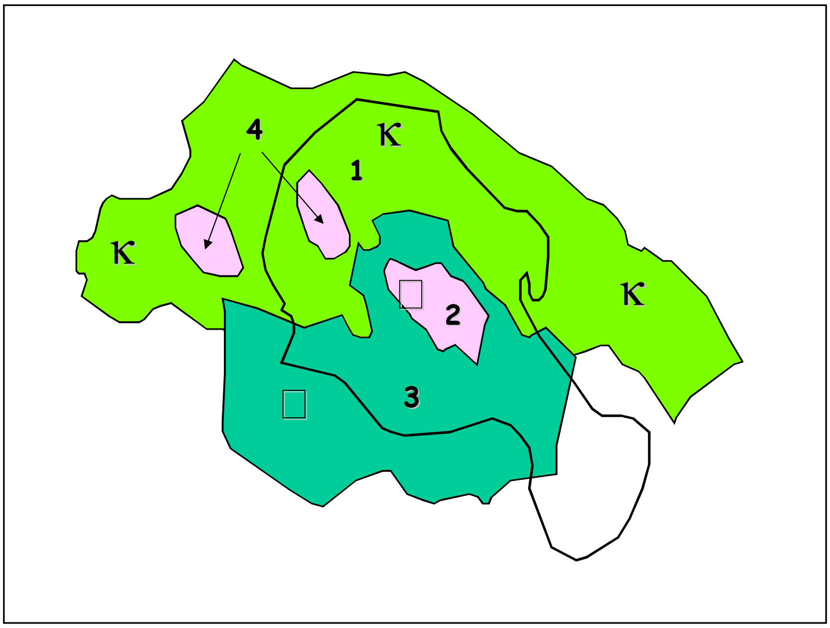

Figure 6. An query example of SQL3.

gion-of-interest) is marked as a black solid border. It is possible to calculate overlapping biotope areas and retrieve any point which is inside the study area (X) without a map interface, just using SQL3 directly or from a client application trough ODBC.

```
SELECT area(Overlap(biotope_area,
   study_area))
FROM biotope_table, study_table
WHERE biotope_area OVERLAPS
   study_area AND study_area = X;

SELECT object_data
FROM endangered_table
WHERE endangered IS CONTAINED BY
   study_area AND study_area = X;
```

2.2.4 Server Interface

One major task of the spatial server is to be a uniform interfacing layer to all client applications. The interfacing is described more detailed in the chapter 2.2. A basic idea is that the server does not have to know anything about client applications, but the connection (a messaging layer) between clients and the server database is standardised. The data integrity is maintained either by using constraints of the RDBMS or it is programmed in the interface layer on the server's side. That will always secure the integrity even the client software can not take the full advantage of all features of the data.

The server can perform analysis requests from the client, too. This makes possible to integrate spatial functionality into applications which don't have any map interface or spatial functionality.

3 Client applications environment

Client applications are scalable with a functionality and intelligence. The application itself does not have to have any spatial functionality, the spatial processing can be done on the server's side. Or a client can be a full scale GIS-system which only needs an interface to the pure spatial data. The most relevant clients are described in the following chapters.

3.1 Essential clients for integrated usage

Clients are communicating with the database through ODBC-compliant database drivers or linked programming libraries, like C application interfaces (C-API, see Fig. 7).

The FPS has defined four categories of client applications. These are

1. Server tools for snapshots or spatial queries.
2. Non-spatial ODBC-clients (register, reporting and data warehouse applications).
3. Customised spatial applications built with spatial toolkits.
4. Professional stand-alone software packages using the centralised data.

Server tool is either a command line SQL3-interface or a simple Tcl/Tk-script. Non-spatial applications (often form based) can use spatial data and functions through an ODBC-connection without a visible map. Essential spatial functions consist of area or other metric calculations and intersection calculations and selec-

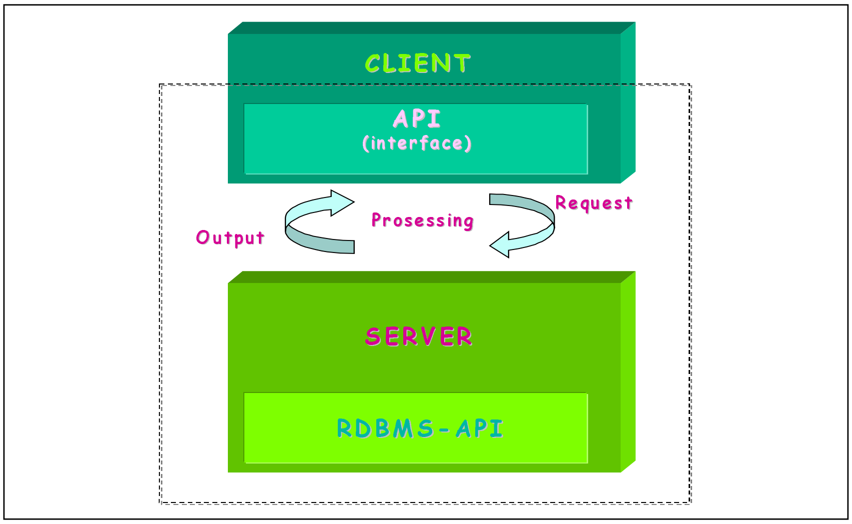

Figure 7. C/S concept (A simplified model)

tions with a spatial aspect. Spatial toolkits are program libraries containing an interfacing client API for the database. A part of the functionality can be linked other applications. MAP/X is a commercial product for the MapInfo Spatial Ware-environment.

The clients of the pilot are MapInfo Professional 4.5. as a map client and a PowerBuilder ODBC-client as a register and reporting application. These sub-systems work in conjunction (see Fig. 9). Objects can be equally selected from the map or form views and the selection is reflected on-line to an another application through a DDE- or OLE-link.

A MapInfo client is slightly customised version (with an ODBC-driver) of a standard MI Professional product. Even single objects can be uploaded from the database with the link. A conflict mechanism detects and reports concurrent manipulations of objects during a download process.

LUOTI Register Application is a client which is made with PowerBuilder. The attribute data is processed and the integrity is maintained by LUOTI application. It has a form based interface to the data. Register Application uses spatial functions of the server for selections and area calculations. The area value of a polygon or virtual polygons (polygon overlays) is not stored in the database but calculated on-the-fly whenever needed.

Various statistical reports about classified areas, timber volumes etc. can be generated with LUOTI. The basis for reporting is a *virtual polygon* which is an overlay between a statistical area (watershed, nature park, municipality, reindeer district…) and an overlapping biotope polygon.

The report is a non-formatted ASCII file which is opened and formatted automatically with VBA-macros of Microsoft Excel. The final formatting with statistical column and row information is finished and formatted in the Excel application. In the future the MS-Map for Excel will be used to visualise map data in this environment.

3.1 Metadata

Metadata is information about the data and it's logical structures. The spatial data should be uniformly described for client applications, especially in environments of multiple users and various client products.

Neither the Spatial Ware- server nor MapInfo- client software support real metadata management. During the implementation the Metadata Management Service has been implemented on the server's side.

The metadata engine (Fig. 8) is used to uniform the conceptual data model and different models used by clients.

The data model describes a conceptual architecture. The physical description is how the data is stored in the server database and client objects are for the implementation in the MapInfo client.

3.2 Feature classification

The feature class is the primary attribute for every object. That tells, what the object is in the real world. The data model will finally contain approximately 1000 primary feature classes. Every feature class belongs to a feature group where the attribute structure is homogenous.

The naming convention makes the data model hierarchic. I.e. an feature class "G11-01 (motorway)" is constructed the following way: the first letter G is *semantically* an application group *Base Map*. G1 stands for all traffic networks. G11 means the road network (physically a table). A code 01 is and attribute value of class and stands for a *motorways* feature class. With that notation is possible to select all roads (codes from

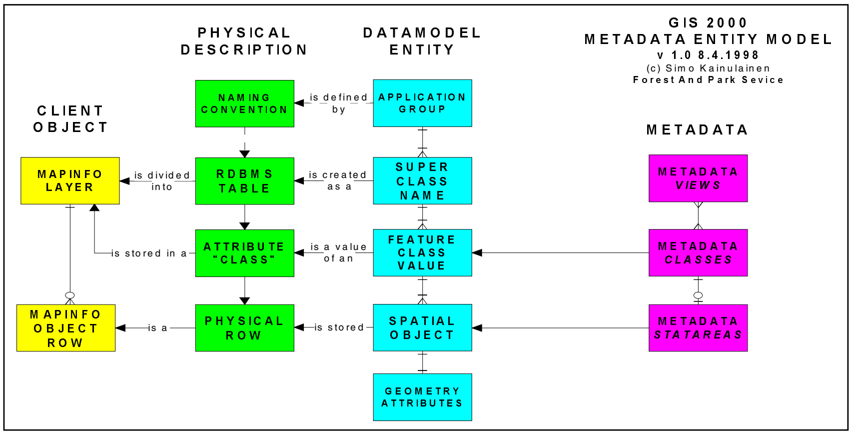

Figure 8. Different conceptual levels of the metadata engine

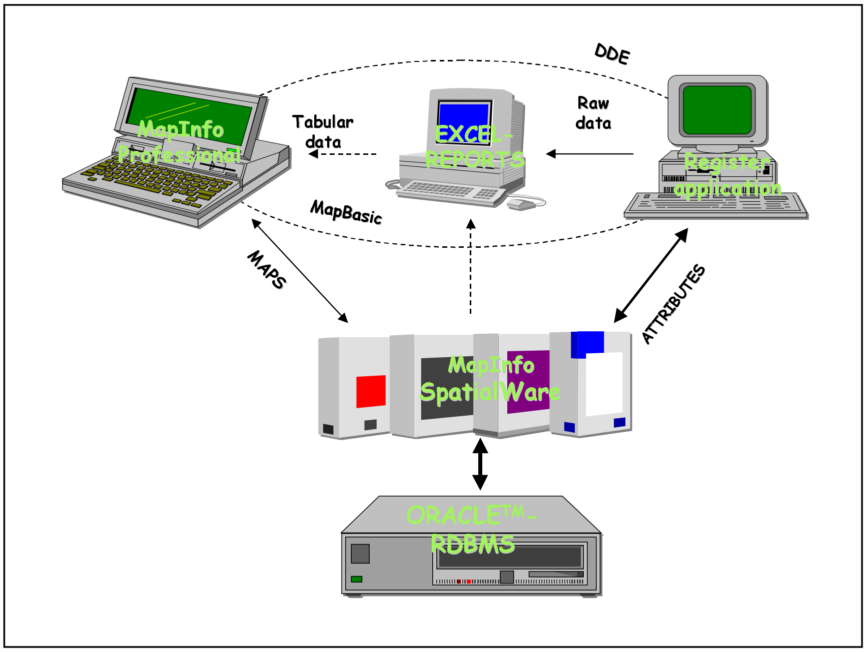

Figure 9. Data interchange and messaging between sub-systems

G11-01 to G11-06) or all traffic networks (codes starting with G1).

The classification and descriptions is stored in the *meta classes* table. For a usage it's convenient to group set of classes inside a *meta view* and select them all together into the application. Some of feature classes are statistical areas for reporting. The list and description of those are stored in the *meta statareas*.

3.3 System design

The system implementation is based on industrial standards which are supported by the Windows NT-operating system. All connections between client systems and the servers are realised by using the ODBC or DDE links. The data which goes though client applications confront all OLE/COM requirements.

When a user selects an object in the mapper client, the unique id is sent to a register application through a DDE link. If a new object is selected in the register application, a mapbasic command is sent to select same object on the mapper automatically. It is also possible to create and store user definable **working sets** of objects and name and store them in the database table, which is open to all applications.