

Remote Access to Active Spatial Data Repositories

Oleg Balovnev, Andreas Bergmann, Martin Breunig,
Armin B. Cremers, Serge Shumilov

Department of Computer Science III
University of Bonn, Germany

e-mail: oleg@cs.uni-bonn.de

Abstract.

1 Introduction

Due to the vast variety of geo-scientific tasks it is hardly imaginable that all of them could be solved within a single GIS. There will be always a need for special-purpose applications which could solve tasks not good enough covered by established GISs. It is natural for such application to benefit from general geo-scientific data already accumulated in standard GISs. However, these data are not always available without significant losses inevitable during data conversions. Even in a larger degree it is true for data processing services provided by GISs.

At the University of Bonn we are involved in several joint projects with diverse geo-scientific groups¹. These projects are rather different, however, all of them have a common subgoal concerning a non-redundant, efficient and robust maintenance of spatial data within databases. A uniform representation and access to objects permits to open data, which were available only for a particular geo-scientific group, for related geo-scientific domains.

The central role in our approach belongs to active spatial data repositories built on top of *GeoToolKit* [4,5] - an object-oriented class library primarily intended for the efficient maintenance of spatial data within object-oriented databases. A common basic object model which is mandatory inherited both by all data repositories and by all programmes provides a necessary background for the interoperation between extremely heterogeneous geo-

scientific applications. Uniformly stored objects serve as mediators between different data representations. The role of such spa-

tial data repositories is not restricted only to the passive data storage. The object-oriented paradigm makes available for external applications various geo-services (e.g. efficient spatial retrieval, preliminary data filtering, etc.) which can be applied to data stored in repositories.

2 Heterogeneity of Geo-Scientific Data

Modeling geological structures and history always starts with a careful geometric analysis of the present assembly of geological surfaces, bodies and property distributions known as geological maps. Till now geological maps are two-dimensional. Only a well-trained geologist is capable to restore in mind underlying 3D structures from a set of 2D maps. The progress in computer sciences makes actual the maintenance of digital 3D geological maps. An actual geological map is also a key to reveal the nature and interaction of earlier processes. A process of computer-aided development of a consistent geometric model is known in geology as *interactive 3D/4D modeling* [1]. The generated model is tested by backward restoration and balancing. The final model is used for the specification of boundary conditions for 3D transport models, or for the production and consistent updates of geological maps.

The starting point for the interactive geological 3D/4D-modeling is the digitalization of geological *sections* gained from open-cast workings. A section is a geological abstraction obtained as a result of an intersection of a vertical plane with geological strata and faults. Geological strata are sheet-like structures in earth with different mineral composition, texture and/or grain size. A fracture in earth materials along which the opposite sides have been displaced is known as a fault. A fault is modeled as a simple surface. A stratum needs a more complex representation. Usually it is modeled as a list of layered bodies which are usually specified by their bounding surfaces. Alternatively a stratum can be defined as a set of spatial solids. The choice of the representation depends on requirements posed by a particular geological application. For example, a solid representation occurs to be more preferable for geological modeling because it permits an efficient derivation of stratigraphic values for arbitrary location in a space. Within a section strata and faults are represented as point sets grouped in stratigraphic and fault lines. Every point in a section contains 3D-coordinates complemented with geologically-specific data like stratigraphy. The second step in the interactive 3D-modeling is the generation of *triangulated surfaces* spread between sections [9]. The final step is a transition from stratigraphic bounding surfaces to *volumes*.

¹ This work is funded by the German Research Foundation (DFG) within the collaborative research center SFB350 at the University of Bonn and the joint project "Interoperable geo-scientific information systems".

Another large source of raw data used in geological modeling is digital elevation model (DEM) and drilling wells. A typical well consists of a mandatory header incorporating its location and diverse accounting data. Optionally a well contains diverse measurements: electrical logs, samples, stratigraphic interpretations, etc. In general data relevant to a single well have very sophisticated hierarchical structures which need non-standard data types for their representation. At first spatial aspects of well data managing were ignored because available well processing algorithms treated a well isolated from all others wells. However, later it became clear, that a 3D spatial was a good complementation to already available cross-sections and surface representations which could increase the quality of geological modeling.

3 GeoToolKit

To facilitate the development of 3D/4D geo-applications we have developed a component-based software called *GeoToolKit* [4]. The idea is to provide for the application developer a set of geo-oriented software building blocks involving DBMS-based spatial data maintenance, special support for efficient spatial retrieval, communication, visualization and graphical interfaces, which can be assembled in a ready-to-use application even by geo-scientists not experienced in programming. A necessary basis for the integration of diverse components is provided by the object-oriented programming technique. *GeoToolKit* is not a closed GIS-in-a-box package - it is rather a library of C++ classes that allows the incorporation of spatial functionality within applications.

GeoToolKit addresses primarily the efficient storage and retrieval of 3D-spatial objects within a database, providing a class hierarchy for the representation of various geometric entities. Currently *GeoToolKit* includes classes for the representation of simple (point, segment, triangle, tetrahedron) and complex (curve, surface, solid) 3D spatial objects. Complex shapes are approximated and decomposed into a set of adjacent simple objects of the same dimension. Such representation turned out to be especially beneficial for the maintenance of non-regular shapes which are typical for the majority of geo-applications [6]. Therefore polylines, triangle networks and tetrahedron networks serve as default representation for curves, surfaces and solids, respectively. However, the user can realize his own representations. Following the object-oriented modeling technique, real world entities such as drilling wells, geological sections or strata, can be modeled as specialization of spatial classes. The *GeoToolKit* class hierarchy is complete: any 3D-spatial object can be modeled either directly by one of the built-in spatial classes or as a composition of these classes within a group. Geometric operations are closed: operation results can be used in further operations. Spatial objects are maintained in special container classes - *spaces* - which are capable of efficient retrieval of elements according to their location in space. To provide an efficient retrieval a space utilizes specialized spatial indexes.

GeoToolKit was used for the development of several geo-scientific applications. Applications developed with *GeoToolKit* simply inherit geometric functionality from *GeoToolKit*, extending it with the task-specific semantics. *GeoStore* is an information system for the management of geologically defined geometries [3, 4] used in 3D interactive modeling. Now *GeoStore* evolves into a general-purpose spatial data browser capable to deal with arbitrary *GeoToolKit*-based spatial data repositories. Among other applications it is worth to mention *WellStore* [3] used for managing well data and *GeoDeform* [2] used for managing spatio-temporal geological data.

The design methodology we elaborated while developing these applications is to reuse as much functionality from the *GeoTool-*

Kit geometric kernel as possible. To achieve this we tried to find a spatial object most suitable to represent a given geological entity. Then we specified a new class describing the geological entity as a direct successor of the chosen spatial object. Due to this fact any geological entity could be always treated as a spatial object. Often there was even no need to redefine geometric functions. Class definitions contained only application specific data members and functions - the classes became more observable and understandable. Less code needed to be written also turned out to be a very encouraging consequence because the future developers are likely to be rather geo-scientists than programmers. Some geological entities contained additional data not known in the "pure geometric" world (e.g. topological neighborhood). Using the methods overriding mechanism we used these topological guidelines to make geometric operations more efficient. However, all overridden methods still work as before (though not as efficient as newly-defined ones). This technique has a very important consequence: the geometric representation of any object is automatically available for all other applications built on top of *GeoToolKit*. It can provide a basis for the development of general-purpose spatial data browsers.

4 Accessing Spatial Data Repositories

We distinguish three levels of access to data repositories: direct access, via an object request broker and via the internet. In the first case data are accessed via the DBMS client-server bus. Applications using the direct access have to import class definitions from the *GeoToolKit* class library. Implicitly (via *GeoToolKit*) they utilize *ObjectStore* classes. The direct data access provides the highest efficiency rate and the most complete functionality. It can be recommended for tasks which need a permanent access the database content, for example, during geometric computations.

However, there are several severe restrictions. All applications which intend to use such data access are obliged to utilize the *GeoToolKit* class library. The problems can arise from the incompatibility of C++ compilers on different platforms and from the tight coupling with the *ObjectStore* ODBMS. Currently *GeoToolKit* is supported only for Sun and SGI workstations. All application accessing the data repository are required to be *ObjectStore* clients that can considerably restrict the number of potential users. Database-level communication facilities are usually not flexible enough for the advanced navigation in the network. Slow networks can also be a bottleneck. For example, the fact that the whole processing is performed at the client site can lead under some conditions to the unjustifiable overload of the network.

This motivated us to develop complementary methods which would enable a more flexible and independent access to our spatial data repositories. What we needed was a kind of standard distributed computing platform. Taking into account the object-oriented nature of data in the majority of geo-scientific applications the most suitable solution is *Object Management Architecture* [10] which promises to become a world-wide standard. Basing on *Common Object Request Broker Architecture* (CORBA) we can expect that any other CORBA-compliant application can get an open access to *GeoToolKit*-based data stores.

According to CORBA an application only needs to hold a reference to a target object, the *Object Request Broker* (ORB) is responsible for automating other common activities, which usually include locating the target object, activating it if necessary, delivering a request to it, and returning any response back to the caller. A client treats a remote object as an ordinary object utilizing the procedure invocation mechanism conventional for this programming language. Parameters passed as part of the request are automatically and transparently marshaled by the ORB, which ensures

correct interoperability between application and objects residing on different platforms.

CORBA object interfaces are described using an Interface Definition Language (IDL). Since the IDL specifications are automatically translated potential inconsistencies between client and server counterparts are significantly reduced, providing a higher degree of type safety than for the bit-stream oriented sockets. We completely re-produced the *GeoToolkit* class hierarchy in CORBA's IDL, trying to keep the interfaces as close as possible to the original C++ ones. To deal with persistency we use a method based on the substitution of the *Basic Object Adapter* by a specialized *Spatial Object Database Adapter* (SODA). A remote application being an *Orbix* client can access via the IDL interface all objects maintained in spatial data repositories. The *GeoToolkit* functionality is available practically in the full volume. All operations which need to process large amounts of data are completely performed at the server site. Since *SODA* runs at the site of the *ObjectStore* server the data exchange rate between them is maximal. Data transmissions between remote clients and SODA are reduced to the minimum. Data are transmitted only when it is really needed, e.g., during initial load of data in the repository.

CORBA-based access was used for the iterative refinement of 3D-model by using in rotation geological (*GOCAD*) and geophysical (*IGMAS*) tools located in Bonn and Berlin, respectively [3]. Geophysical modeling applies gravimetric and magnetic evaluations of the potential fields to extrapolate the geological information gained at the earth surface into the depth [8]. To reduce the variability of initial parameters geophysical algorithms need a kind of rough cast which can be provided by the interactive geological modeling. Spatial data repositories serve as mediators in data exchange between these tools.

This method provides a more flexible remote access to our spatial repositories. It is more platform independent and more suitable for low-speed networks. However, it is also restricted because to access our spatial data repositories an application has to be a CORBA-client and it should be preliminary prepared: pre-compiled with *GeoToolkit*'s IDL specifications.

5 Spatial Data Repositories in the Web

The spread of the internet made actual the access to spatial data repositories via the Web. Naturally such access is relative restricted in comparison to the overall functionality provided by *GeoToolkit*. At the same time Web-Browsers can provide rather advanced facilities concerning presentation of 3D objects using VRML [12]. VRML (Virtual Reality Modeling Language) is a network transparent protocol for communicating 3D graphics over the World Wide Web. The ability to represent 3D-worlds can be very beneficial for Web-oriented geo-scientific applications.

To provide the access to spatial object repositories via Web-browsers we used HTML-forms and the *Common Gate Interface* (CGI). A CGI script activated from a web-page does not directly communicate with the data repository. Instead of this it establishes a connection to a permanently running program – *GeoServer* – which supervises all *GeoToolit*-compliant databases managed by the given *ObjectStore* server. *GeoServer* runs at the same site as *ObjectStore* server does, therefore, has the most efficient database-level access to the repositories. Every applications accessing a *GeoToolkit*-compliant database automatically communicates with *GeoServer* using a standard socket-mechanism integrated in the class library. Primarily *GeoServer* carries out diverse accounting tasks. It maintains its own database where it stores accounting data which permits to achieve a higher level of global referential integrity. For example, it automatically register all

inter-database links appearing when a user creates his own data configuration on top of existing data repositories.

In order not to multiply the number of permanently running programs we decided to augment *GeoServer* with a special component (*GeoWeb*) responsible for the communication with the Web. The CGI scripts activated from Web-pages contact *GeoWeb* using the socket-stream mechanism. Thus we have got a two-layer client-server architecture. *GeoWeb* is a server related to asynchronously activated CGI-scripts (clients). At the same time *GeoWeb* is itself a client related to the *ObjectStore* server. One of *GeoWebs*' main tasks is to provide for multiple clients a safe concurrent access to data repositories.

A typical form for selecting objects of interested from spatial data repositories is presented on Fig. 1. *GeoWeb* analyses input it gets in the CGI-format and performs a corresponding query. Query results are returned back to the CGI-script either in the HTML-format for text data or in the VRML-format when a graphical representation is required. If necessary *GeoWeb* generates new forms for a more precise selection of objects. Another important task which performs *GeoWeb* is a sessions control. *GeoWeb* stores the workspace (identified by the session) between consequent activation of CGI-scripts. Session control permits to avoid permanent re-initialization of the examined database.

GeoToolkit's class library also was extended: every spatial class has got a member function producing output in the VRML format [12]. Basic geological entities turned out to be good mapped onto VRML representation (Fig.2)². For example, a geological cross-sections are represented as a combination of an *IndexedFaceSet* node which corresponds to a vertical cutting plane, and a list of *IndexesLineSet* nodes, each representing a margins between neighbor geological layers. Geological surfaces (faults and stratigraphic boundaries) are modeled as *IndexedFaceSet* nodes. A digital elevation model is mapped onto the *ElevationGrid* node.

Some geological specializations of geometric types could benefit from representations different from those they by default inherited from their spatial parents. For example, a geological well is represented and visualized in *GeoStore* as a line segment. VRML can offer much more advanced visualization facilities than *GeoToolkit*'s embedded 3D-viewer can afford. By redefining a method responsible for the VRML export we can generate a more informative 3D well representation, e.g., by using the *Extrusion* node.

² For the visualization we use the *CosmoPlayer* plug-in by SGI which supports a much more advanced VRML2 standard and is rather fast and robust.

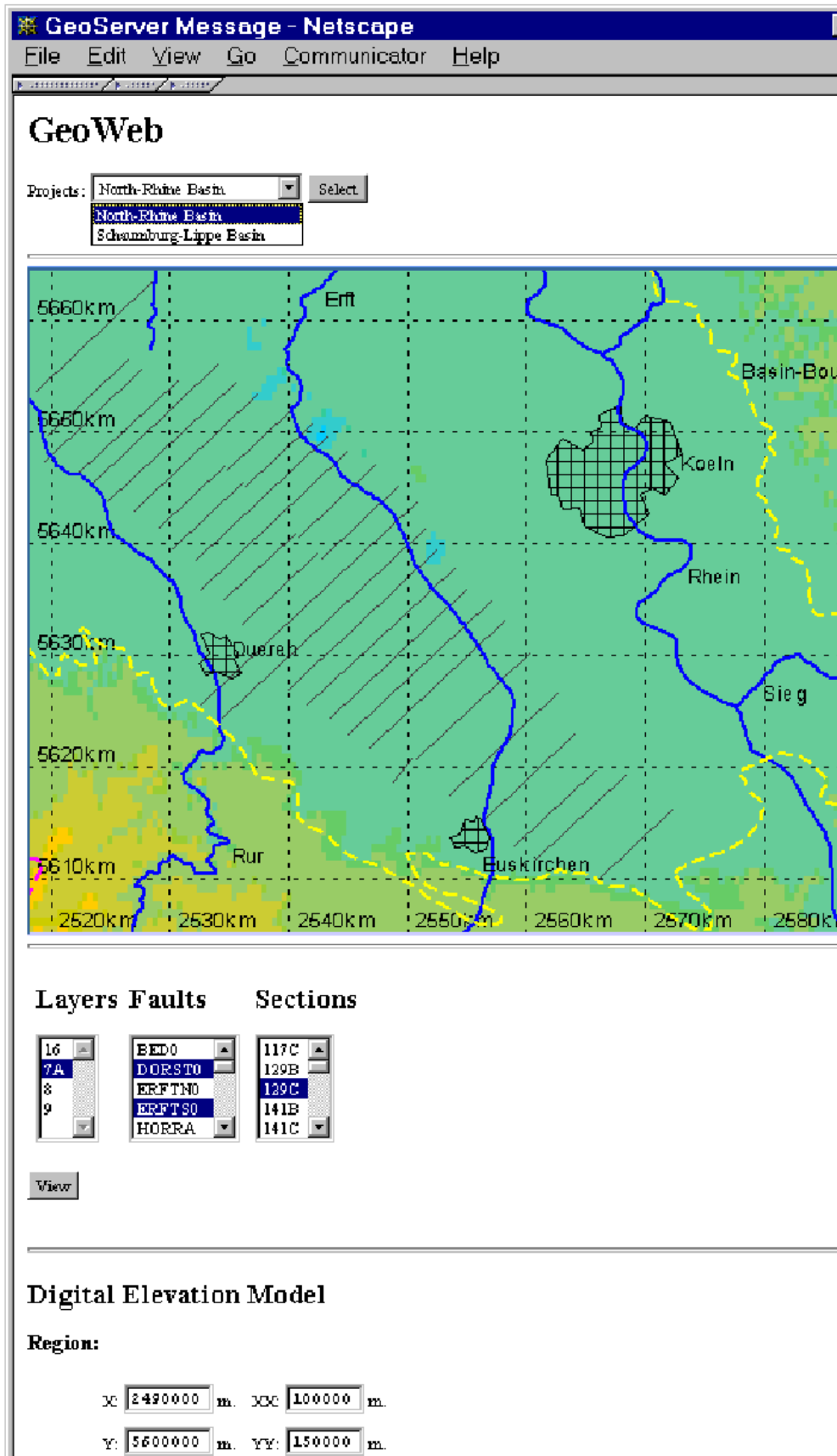


Fig 1. Forms for querying GeoToolKit-based spatial repositories in the Web.

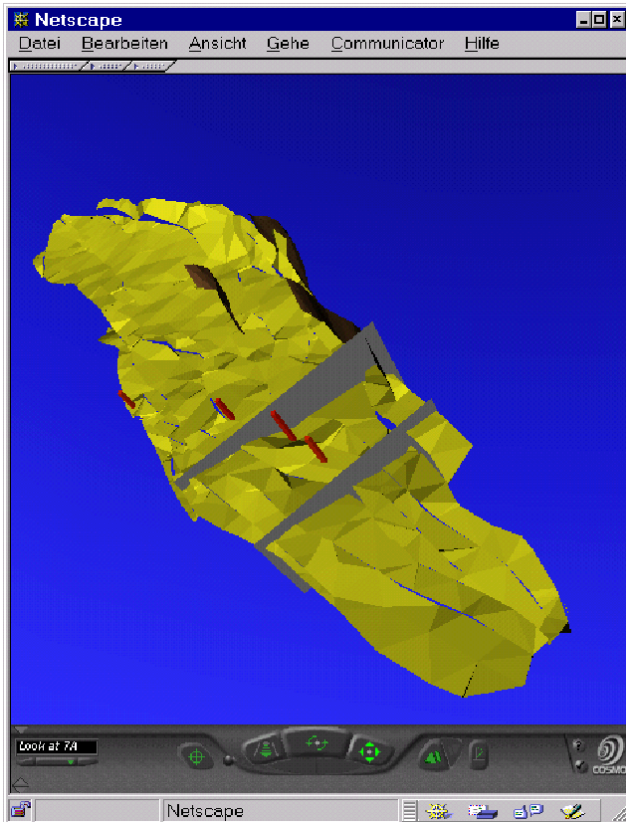


Fig. 2. Geological entities in VRML: stratigraphic surface, faults surfaces, geological cross-sections and wells

Currently the querying of data repositories is performed in rather traditional non-spatial form provided by HTML-forms (Fig. 1). Naturally, spatial navigation capabilities are more preferable in geo-scientific applications. In order to provide spatial navigation capabilities similar to those implemented in the *GeoStore* browser we develop an extensible set of graphical 2D/3D interface components using Java and a CORBA compliant communication infrastructure. This work is carried out in the context of a research project dealing with integrated access to distributed and heterogeneous geo-scientific data [7]. This approach has several advantages. Because nowadays Java technology is fully integrated with Web-Browsers (at least by means of the Java Plugin) the user can be provided with extensive capabilities for intuitive spatial browsing within the usual browser environment. The exploration of diverse geo-scientific data types can be supported in an adequate manner by different applets. An additional advantage is the dynamic configuration capabilities of the interface components. In contrast to the former approach the user is now enabled to mask unnecessary information and adapt the presentation according to special requirements.

At first, our work focused on the development of a 2D map component (Fig. 3). All spatial entities are represented by 2D geometric objects like polygons or markers in case of point information. The graphical objects are organized in layers. The visibility of the different layers can be controlled by the user. Furthermore, every geometric object is connected to a legend object describing the respective visual properties. The user can manipulate these properties to create his personal view of the map. Different levels of detail are also supported. The zooming facilities of the map component and the possibility to control layer visibility by the zoom factor together allow for logical

zooms. The map component is geo-referenced and supports spatial queries by selection predicates like points and rectangular areas. By drawing a rectangle on the map the user instructs the *GeoServer* to locate any geo-scientific object being completely or partly covered by the specified area. The results are visualized in the map. The user may further explore a geo-scientific object by clicking on the respective marker. The requested information as well as in case the interface component needed to visualize and navigate the data is fetched from the *GeoServer* and presented to the user.

Currently we develop active interface components for the visualization and exploration of drilling data as well as other 3D geo-scientific objects and maps based on VRML.

Acknowledgements

This work was not possible without the excellent cooperation with the geo-scientific groups of Richard Dikau and Agemar Siehl (University of Bonn) and H.-J. Götze (Free University of Berlin). Special thanks to Wolfgang Mueller, Martin Hammel, Markus Assmann and Uwe Radetzki for their contribution to the implementation.

References

- [1] R. Alms, K. Klesper, A. Siehl. Three-Dimensional Modeling of Geological Features with Examples from the Cenozoic Lower Rhine Basin. In: Foester, A, Merriam, D (Eds.) *Geological Modelling and Mapping*, 113-133, Plenum Press, New York, 1996.
- [2] R. Alms, O. Balovnev, M. Breunig, A.B. Cremers, T. Jentzsch, A. Siehl. Space-Time Modelling of the Lower Rhine Basin Supported by an Object-Oriented Database. In: *Physics and Chemistry of the Earth*, in print. XXII General Assembly of Geophysical Society, Vienna, Austria, 21-25 April 1997, In: *Physics and Chemistry of the Earth*.
- [3] O. Balovnev, M. Breunig, A. Bergmann, A.B. Cremers, S. Shumilov "A CORBA-Based Approach to Data and Systems Integration for 3D Geoscientific Applications." *Proceedings of International Conference on Spatial Data Handling - SDH'98*, Vancouver, Canada, July 98.
- [4] O. Balovnev, M. Breunig, A.B. Cremers. *From GeoStore to GeoToolkit: The second step*. M. Scholl, A. Voisard (Eds.): *Advances in Spatial Databases*, *Proceedings of the 5th International Symposium on Spatial Databases (SSD'97)*, Berlin, Germany, July 1997. *Lecture Notes in Computer Science*, Vol. 1262, Springer, 1997.
- [5] O. Balovnev, M. Breunig, A.B. Cremers. *GeoToolkit: Opening the Access to Object-Oriented Geodata Stores*. M. Goodchild, M. Egenhofer, R. Fegeas, C. Kottman (Eds.): *Interoperating Geo-Information Systems*, Kluwer, 1998 (in print).

- [6] O. Balovnev, M. Breunig, A.B. Cremers, M. Pant. "Building geo-scientific applications on top of *GeoToolkit*: a case study of data integration". In: *Proceedings of the conference on scientific and statistical databases*, 9pp., Capri, IEEE Computer Society, 1998, pp.260-268.
- [7] A. Bergmann, H. Gärtner, M. Breunig, A.B. Cremers, R. Dikau. *Design and First Steps in the Development of OPALIS*. D. Fritsch, M. Englich, M. Sester (Eds.): *Proceedings of the ISPRS Commission IV Symposium on "GIS - Between Visions and Applications"*, Stuttgart, Germany, September 1998. IAPRS, Vol. 32, Part 4, 1998.
- [8] H.-J. Götze, B. Lahmeyer. *Application of three-dimensional interactive modeling in gravity and magnetics*. In: *Geophysics*, Vol. 53, No. 8, pp. 1096-1108, 1988.
- [9] J.L. Mallet. *GOCAD: a computer aided design program for geological applications*. In: A.K. Turner, *Three-Dimensional Modeling with Geoscientific Information Systems*, NATO ASI Series C, Vol. 354, Kluwer Academic Publishers, pp.123-141, 1992.
- [10] Object Management Group. *CORBA 2.0/IIOP Specification*. OMG formal document 97-09-01. <http://www.omg.org/corba/c2indx.htm>
- [11] A. Siehl. *Construction of Geological Maps based on Digital Spatial Models*. *Geol. Jb.*, A104, Hannover, p.253-261, 1988.
- [12] An Overview of the Virtual Reality Modelling Language.

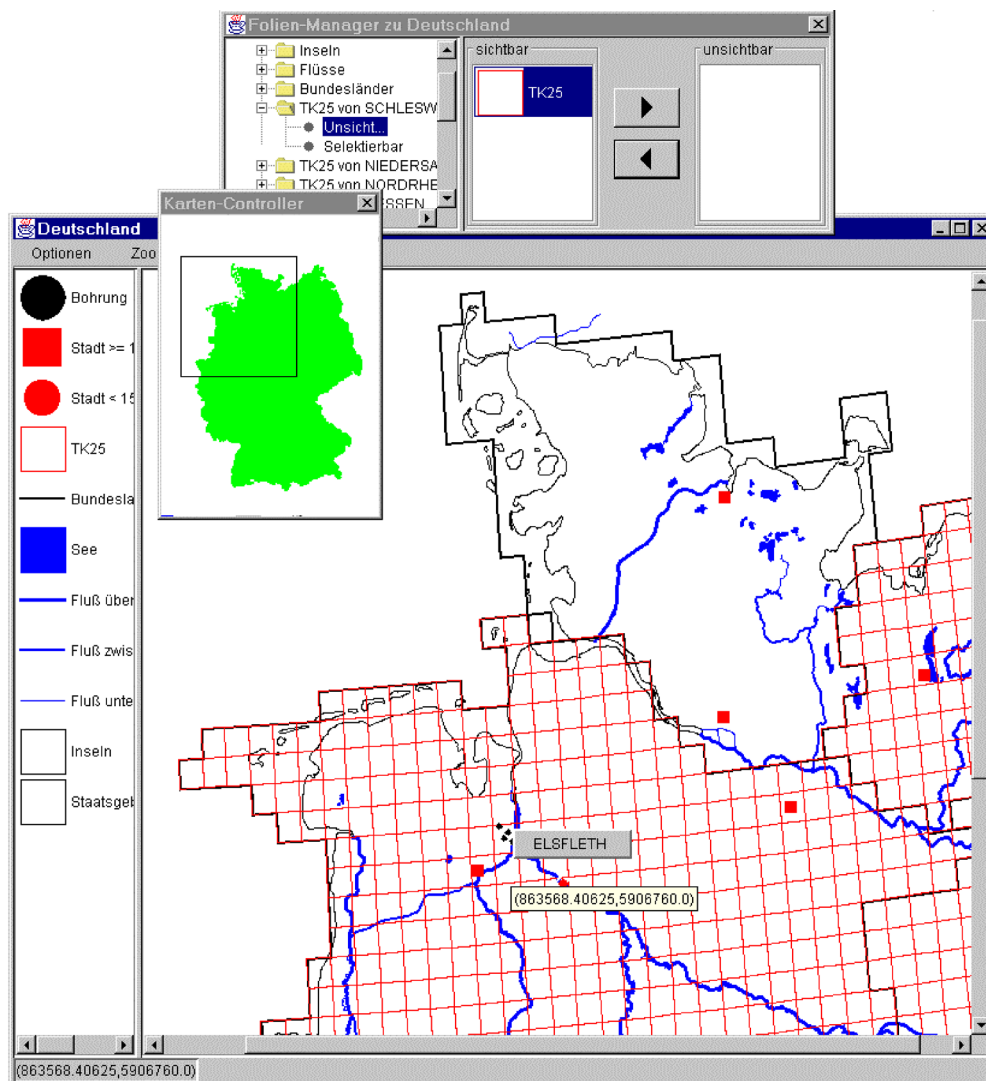


Fig. 3. The map interface component.