# Usage of Spatial Data Stores for Geo-Services[56]

Martin Breunig, Wolfgang Bär and Andreas Thomsen
University of Vechta
P.O. Box 1553, 49364 Vechta, Germany
{mbreunig, wbaer, athomsen}@fzg.uni-vechta.de

## SUMMARY

*Distributed mobile geo-services require versatility, interoperability, portability of services and data. These are achieved through the use of efficient data stores and data processing services. However, restricted bandwidth is a bottleneck for the transmission of large data sets typical in 2D and 3D geo-scientific applications. Although the extensible markup language (XML) is a flexible data exchange format in widespread use, the object-oriented approach is widely accepted as the standard approach to 3D geometry modeling. In this paper, we discuss the usage of different object-oriented data stores for distributed 2D and 3D geo-scientific applications that use XML for data exchange. We compare the performance of a native XML data store with two object-oriented database management systems (OODBMS) featuring a page-server and object-server architecture, respectively. A 2D route planning system realized with an XML- and an object-oriented data store illustrates the advantage of a genuine OODBMS over an XML data store, if the retrieved data has to be transformed afterwards into an object representation for further processing. A 3D geological model based on simulations serves as a test bed for the comparison of page-server and object-server OODBMS, illustrating the superiority of page-servers for large complex 3D geometry objects composed of more than 100,000 elements. We finally present a test study on the construction of a profile section as an example of a geo-service. This illustrates that the main load is caused by the database and cutting operations, whereas the time for XML conversions of the resulting objects can be neglected.*

**KEYWORDS:** *spatial data usability, geo-data management, XML, object-oriented data store, mobile geo-service.*

## INTRODUCTION

In future, geo-services could provide ubiquitous access to geo-data from a technical stand point. Therefore the efficient exchange of geo-data will be a central and critical task of information processing (Giguère, 2001; Breunig & Bär, 2003). In our view a geo-service is not only responsible for the retrieval of geo-data, but also for the storage, update and – depending on the type of geo-service – the processing of the data before serving it.

The requirements on versatility, interoperability, portability, and performance of mobile and distributed geo-services are considerable, as are the requirements on 3D and 4D geo-databases in a distributed and mobile environment. Such geo-databases must efficiently manage a considerable number of large and complex application-specific objects such as 2D, 3D and spatio-temporal models. The necessity to achieve a clear, reliable and not overly complex implementation has to be weighted against the requirements on flexibility and performance managing complex geometric and topological objects in

[56] Research project "Advancement of Geoservices" (Weiterentwicklung von Geodiensten) funded by the German Ministry of Education and Research (BMBF) by grant no. 03F0373B et al. The responsibility for the contents of this publication is by the authors.

the database server. Furthermore, the requirements on interoperability and speed of data transmission in an environment of low bandwidth have to be respected.

The object-oriented approach to geometry modeling has been well established for several years. In the domain of database management, however, object-oriented DBMS have not succeeded in supplanting the so-called object-relational approach, i.e. the extension of relational databases by object-oriented features. In this paper, we shall concentrate on genuine OODBMS, because they support a more straightforward mapping of geometry models onto persistent storage, rather than the object-relational approach.

As a means for flexible and extensible data exchange in a heterogeneous environment, XML is in widespread use. Due to its extensibility, XML can be used to express any object-oriented data structure, though at the cost of considerable redundancy. However, this issue can be resolved by the use of general-purpose compression techniques. Moreover, the style sheet language XSL and the XSL transformation language XSLT provide tools for building data format interfaces between different XML representations, whereas XQUERY and XPATH are languages designed for the retrieval of parts of XML-coded data repositories.

We present two examples, a case study and a geo-scientific application scenario and compare them using XML-based and different object-oriented data stores. In the next section, we briefly describe a mobile route planning service as an example of a 2D geo-service. The storage and retrieval of data in an XML- and an object-oriented data store are discussed and evaluated. We then introduce geological application scenarios as examples of processing complex 3D geometries. Tests with large 3D geometry objects illustrate the performances of page-server and object-server based architectures in OODBMS.

## Case study "route planning-service"

Mobile route planning services need database support for the search of paths in a graph and for spatial region queries as well as for the support of updates in routes. A mobile bicycle route planning service described in detail in (Breunig & Bär, 2003) has been coupled with an XML data store and with an object-oriented data store, respectively.

The data set of the bicycle routes consists of 40,908 nodes connected through 51,800 bidirectional edges weighted by Euclidean distance. There are no one-way streets. The selected routing area in Lower Saxony, Germany extends approximately 60 km E-W, and 80 km N-S.

Like all services discussed here, the route planning service was implemented in Java. It comprises a server data store providing initial route computation and a mobile data store for supporting offline data management at the mobile client. In this text, we focus on the server side and leave aside the experimental prototype of a simple spatial object store for the mobile client. Retrieval from the data store is followed by a transformation step before processing the data by graph algorithms. In the following we present the extension of a native XML and an object-oriented data store for the management of graph data.

### *Native XML data store*

Extending the Java API of a native XML-based data store, we have implemented a spatial R-tree based index (Guttman, 1984) with a simple support of transactions being independent of the data store. The extension also comprises interfaces and classes for accessing and managing XML route data. To enhance retrievals by location, all XML documents containing a defined element "BoundingBox" have been inserted into the spatial index. Furthermore, the XML query language of the data store has been extended by spatial query operations like *intersects, contains* and *nearestOf*.

Table 1 shows the performance of the routing operation on the XML-based server data store expressed by the ratio of total time for route computation by number of nodes on the computed route.

| # Nodes | # Edges | # Nodes on route | Routing time/Nodes on route (ms per node) |
|---|---|---|---|
| 14442 | 18553 | 334 | 382.4 |
| 7140 | 9172 | 226 | 274.5 |
| 835 | 1088 | 66 | 456.4 |

*Table 1:* Performance of the routing algorithm on top of the XML-based data store

Visibly the times for the XML-based data store for the routing application are not yet acceptable, the principal reason being the necessary transformation of the data structure within the XML-based data store into the object representation required by the routing algorithm. Nevertheless, this extension of the data store API for the management of XML route data has drastically reduced the necessary time for spatial access to the XML documents, especially for small result sets.

*Object-oriented data store*

Our spatial access extension of an object-oriented data store for the management of route data is based on an R*-tree (Beckmann et al. 1990) with algorithms for nearest neighbour search. The R*-tree is modeled by application objects of the object-oriented data store. The object-oriented data store is responsible for transaction processing and concurrent access to the objects modeling the R*-tree.

| # Nodes | # Edges | # Nodes on route | Routing time/Nodes on route (ms per node) |
|---|---|---|---|
| 14442 | 18553 | 334 | 137.7 |
| 7140 | 9172 | 226 | 132.7 |
| 835 | 1088 | 66 | 393.9 |

*Table 2:* Performance of the routing algorithm on top of the object-oriented data store

Table 2 shows the query results for the object-oriented data store. The routing times are by factor 2 to 3 faster than those for the XML-based data store, except for small routing queries with only a few sub-graphs involved.

*Remarks*

These results were obtained for single user queries with a sub-graph loading approach (see Breunig & Bär, 2003) that leads to blocking of the graph processing until the next requested sub-graph is loaded into memory. Using this architecture in a multi-user scenario with sub-graph caching strategies will improve the performance significantly.

In both tests the necessary time for transformation of the results to XML after processing is not included. The results show the advantage of an object-oriented over an XML based data store, if the geo-services involve further processing of the retrieved data before serving the client application. On the contrary, if the main focus is on serving the retrieved data directly as XML, or if the data is to be transformed only in different XML formats through XSLT transformation, a native XML data store may be the right choice.

## Geological 3d application scenario

Whereas classical GIS-applications generally represent the surface of the earth by 2D or 2.5D geometry models, geological applications concern the subsurface, and therefore use genuine 2.5D or 3D geometry models. There is, however, another feature that distinguishes geological modeling from classical GIS: the subsurface in its entirety is not accessible to observation. Therefore the position, form, and structure of geological bodies must be inferred from spatially limited observations e.g. by statistical estimation, or by the interpretation of seismic data. A lot of geological background knowledge is necessary to achieve a useful model of the subsurface, which will always be subject to a considerable uncertainty.

Therefore, the maps and sections a geologist uses, are always only so many projections of and cuts through a background model that has undergone a number of interpretation and estimation steps. This is true even for traditional geological maps. The introduction of informatics, however, has permitted to communicate such knowledge not only in 2D maps and sections derived implicitly from geometry models, but to make the process of geometrical modeling of the subsurface explicit and communicable. Present-day 3D-modelling tools like GOCAD®, SURPAC®, LANDMARK®, GEOQUEST® allow several geologists to co-operate during the establishment of a subsurface model, making the process of modeling reproducible.

*Requirements*

Database services for subsurface geology applications must provide access to entire 3D-models, as well as to 2D representations (projections and sections) derived thereof. Whereas in the case of mostly undisturbed sedimentary bodies, triangulated 2.5D surfaces representing strata boundaries may be sufficient, the general case comprising also important faults and folds, or non-stratiform bodies (e.g. salt-domes) requires true 3D-models.

In a distributed and mobile environment, a 3D-geometry database server for geological applications should enable the geologists in the field, as well as in the laboratory, to refer to a shared common model of the subsurface during the process of data caption, processing, interpretation and assessment. The cycle of steps involved in updating a geological model can be long, however, and the result may never be free of subjective appreciation. Therefore, rather than supporting direct editing by transaction management, it is advisable to use strategies of version management to control the evolution of the shared model.

A comprehensive subsurface model may consist of hundreds of geological bodies, each represented by complex objects, e.g. triangulated surfaces, composed of up to more than a hundred thousand elements (e.g. triangles). Considering a portable client instrument, e.g. a robust PDA combined with a GPS client, both the transmission and the graphical representation of such a complex model are not realistic, because of insufficient available transmission bandwidth and performance of the graphical display. On the other hand, the geoscientist in the field probably needs only a selected part of the information, specified by e.g. a 3D-region, a stratigraphic interval, a set of thematic attributes, and some other geometric and thematic criteria. Even such a reduced information may be too large for use in the field, motivating the use of techniques of data reduction and progressive transmission (Shumilov et al., 2002). Graphical representation of a 3D-model can be reduced to a sequence of 2D-sections and projections that are displayed using the limited graphical capabilities of a mobile client. By sliding through successive sections, even a 2D display can provide insight into the form and structure of a complex 3D body.

At the heart of a distributed portable environment for geological applications, there are three components: an efficient 3D-geometry database providing shared access, storage, and retrieval, a comprehensive set of problem-specific operations and transformations, and an interactive 3D-modeling system. Our research project concentrates on the 3D-database and basic geometrical/topological operations. 3D geometry models are represented by simplicial complexes or boundary representations of

complex volume bodies, while an internal R*-tree supports spatial access. We use GOCAD® (Mallet, 1992) as an external interactive modeler.

*3Dto2D service for geological models*

As a concrete geological application we discuss a 3Dto2D geo-service. This service has to provide all the necessary functionality of information reduction from complex 3D models to 2D models in order to make the model usable and displayable on constrained client devices (PDAs). Such a service on a mobile device will allow the field geologist to compare the actual observed situation with information provided by the subsurface model, and to take decisions on sampling accordingly.

The 3Dto2D service consists of the following sub-services joined providing the derivation of 2D profiles from a 3D model:

- *RetrieveService* – supports queries for the complex geological objects.
- *PlaneCut* – cuts a planar profile through the 3D model for a given plane.

6. *BufferCut* – filters objects based on a distance parameter to given buffer object.
7. *PlaneProjection* –projects objects onto the plane profile.

- *AffineTransform* – transforms the resulting 3D object into a 2D xy plane.

The construction of piecewise planar profile sections by repetition of these operations with different parameter values is straightforward. The computed result from the 3D model of this service will be a 2D map in the xy plane representing an arbitrary plane profile section across the model with additional information projected onto the profile. Figure 1 shows the principle steps of the 3Dto2D service.

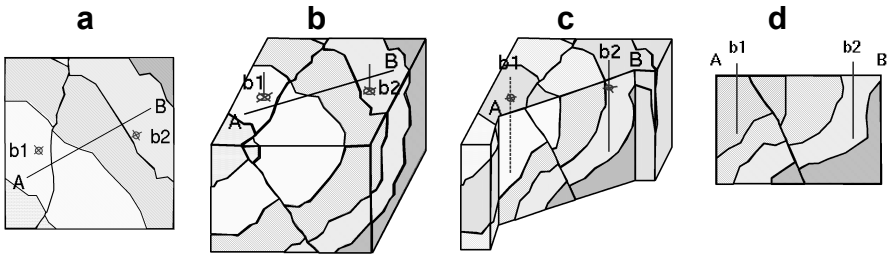


*Figure 1:* Example of a 3Dto2D service: Planar profile section between endpoints A and B, with boreholes b1 and b2. – (a) location in map plane, (b) block view of 3D model, (c) view of profile section with part of model removed, (d) resulting 2D profile section with projected borehole profiles.

Application data used in the current database research project stem from different geological research projects that comprise large surfaces and a number of different 3D-bodies separated by fault surfaces, both as simplicial complexes and boundary representation. For the first performance tests presented here, however, we used simulated geometry objects from the CS-department at Bonn University (Breunig et al., 2001). The benefit is the possibility to produce data of arbitrary complexity with up to millions of geometry elements in one complex object. On the contrary, for testing the 3Dto2D geo-service, we used real world data from an earlier research project (see below).

*Performance Tests for Object Internal Geometric Queries*

The performance tests for the object-oriented data stores are carried out respectively with a page- and an object-server based system architecture of an OODBMS. We examined different spatial queries for the retrieval of the internal geometric elements of a complex geological object. Such queries are used in

almost all geological algorithms based on the optimization through internal spatial indexing of the complex object elements.

For each architecture, a test database was built consisting of two triangulated surfaces comprising 100,000 and 200,000 triangles. Thereafter different spatial queries on the internal geometric elements of the complex objects were executed. Table 3 and 4 show the times needed for the insertion of the surfaces into the test databases as well as the retrieval times for different spatial queries.

The spatial join query between the two intersecting surfaces resulted in a set of 1,979 intersecting triangles. The window queries, performed with two different query sizes, resulted in 1,144 (5%) and 61,255 elements (25%).

| **Page-server based data store** | | **Number of triangles in the surface** | |
|---|---|---|---|
| | **Unit** | **100000** | **200000** |
| Insertion | Minutes | 0.6 | 1.2 |
| Spatial Join | Seconds | 44.3 | |
| Window Query 5% of space | Seconds | | 1.9 |
| Window Query 25% of space | Seconds | | 33.5 |
| Nearest Neighbour | Seconds | | 1.2 |

*Table 3:* Results for the page-server based system architecture of an OODBMS.

| **Object-server based data store** | | **Number of triangles in the surface** | |
|---|---|---|---|
| | **Unit** | **100000** | **200000** |
| Insertion | Minutes | 9.0 | 42.2 |
| Spatial Join | Seconds | 233.3 | |
| Window Query 5% of space | Seconds | | 8.1 |
| Window Query 25% of space | Seconds | | 234.6 |
| Nearest Neighbour | Seconds | | 2.5 |

*Table 4:* Results for the object-server based system architecture of an OODBMS.

Clearly the page-server architecture outperforms the object-server architecture for this special application, due to the large number of internal elements of the complex geological objects. A triangulated surface with 200,000 triangles comprises, including internal R*-tree objects and internal topological relations, approximately 1,100,000 individual objects. Under such a load the object-based server turns into a bottleneck through its objects-based network transport and its object-level locking. The page-server architecture benefits from the fact that it groups about 150 to 200 small objects onto one page which also is the level of locking and of transport over network.

*First results of a prototypical implementation of the 3Dto2D Service*

We prototypically implemented the single sub-services – RetrieveService, PlaneCut, PlaneProjection and AffineTransformation – together providing the 3Dto2D Service. Additionally, we implemented an Output-Service to our internal XML-Format and created an XSLT style sheet for transforming the resulted XML data to a GML format.

The datasets used stem from a former DFG research project (Siehl et al.,2002, Balovnev et al., 2004, Breunig et al., 2001, Breunig et al. 1999). For the purpose of testing the 3D geometry DBMS, 25

boreholes and two geometry data sets were selected: GESAMT (fig. 2), a set of 10 stratum boundary surfaces and a digital terrain model from the "Schaumburg-Lippesche Kreidemulde" in Lower Saxony, Germany, and PORTA (fig. 4), a set of 8 stratum boundary surfaces and a number of fault surfaces, of a smaller area within GESAMT. Whereas the former gives an overall picture of the geological situation, based on the "Geotectonic Atlas of North-West Germany" (Kockel 1996), the latter model represents a smaller area in more detail with the aim of constructing a true volume model in boundary representation. The models were prepared by R. Seidemann and E. Kroll under the direction of A. Siehl at Bonn Institute of Geology.

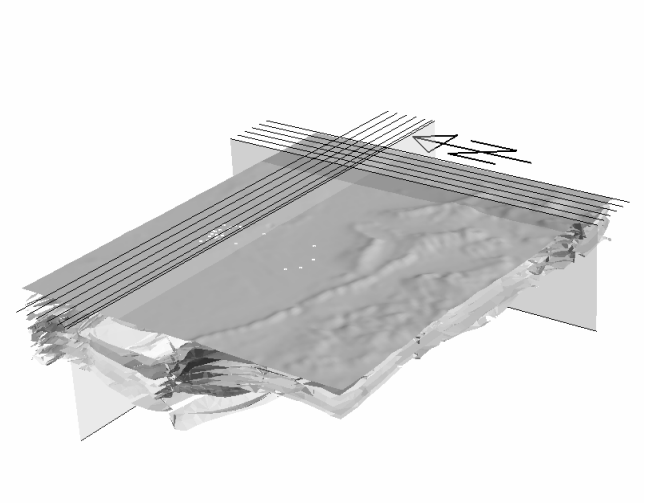

*Figure 2:* The dataset GESAMT consisting of 10 triangulated strata boundary surfaces and a digital terrain model. Locations of profile sections are indicated by straight lines. Vertical exaggeration is 3 times.

## FIRST RESULTS

The results (fig. 3, 5) are derived from the computation of different profile sequences across the GESAMT and PORTA datasets described above. The sequences of sections are computed by the repeated application of the 3Dto2D service with different parameters defined by the individual cutting planes.

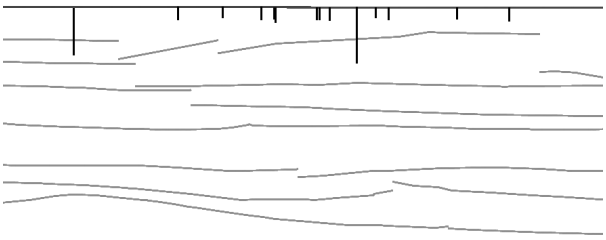

*Figure 3:* Part of a profile section through the dataset GESAMT with digital terrain model, strata sections and projected borehole paths.

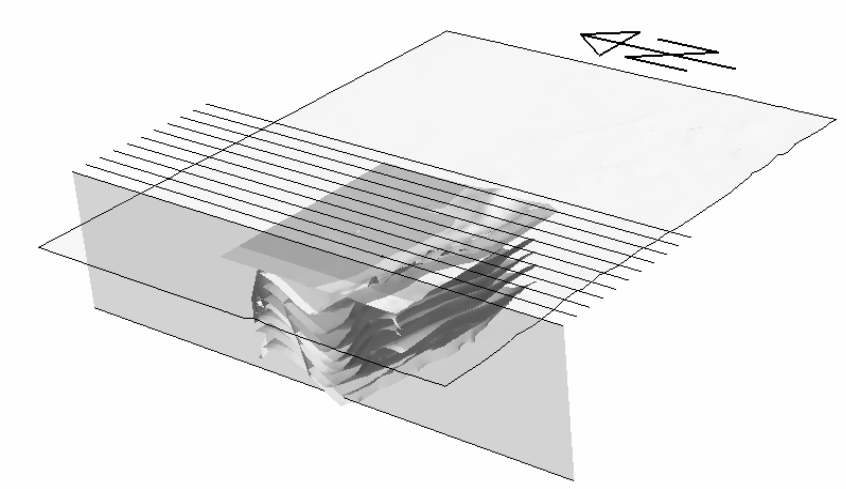

*Figure 4:* Dataset PORTA within the model GESAMT with locations of profile sections. Vertical exaggeration is 3.

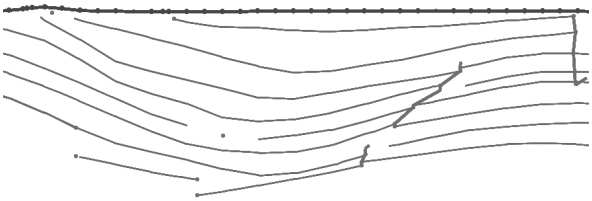

*Figure 5:* Part of profile section through data set PORTA with digital terrain model, strata boundaries and faults.

Table 5 shows the overall time taken to process one profile in the two tests profile sequence 1 and 2. The test dataset GESAMT consists of 10 stratum boundary surfaces with together 89,006 triangles. The time needed for the sub-service PlaneCut is about 15 seconds per profile and the time for the following output as XML and the transformation to GML is dependent on the amount of elements which intersect the cutting plane. The sub-services for projecting (PlaneProjection) further information from the nearby neighbourhood onto the plane, like boreholes, and the AffineTransformation into the xy plane is not shown here as the time needed for this sub-services is negligible in contrast to the time needed for the PlaneCut, XMLOutput and XSLT sub-services (cf. Table 6 for comparison).

| | **PlaneCut (avg. in ms)** | **Elements in Result (avg.)** | **XML Output (avg. in ms)** | **XSLT (avg. in ms)** |
|---|---|---|---|---|
| Profile sequence 1 | 15565 | 1441 | 186 | 3812 |
| Profile sequence 2 | 15363 | 1180 | 149 | 2085 |

*Table 5:* Average times needed for computation of different sub-services.

In Table 6 the time for processing the sub-services PlaneCut, PlaneProjection and AffineTransformation for the profile sequence 1 and 2 is given per intersecting element. As expected with an OODBMS with cache architecture, the first run always takes more time than the following runs where parts of the needed information are already available in the client cache. This impact of the cache can be seen in table 6 as the times for the first profile and the average time of the following profile computations differ by a factor of 2.

| | Service | First profile (in ms) | Following profiles (avg. In ms) |
|---|---|---|---|
| Profile sequence 1 | PlaneCut | 17.63 | 8.45 |
| | PlaneProjection | 0.41 | 0.32 |
| | AffineTransformation | 0.04 | 0.03 |
| Profile sequence 2 | PlaneCut | 20.78 | 10.08 |

*Table 6:* Processing times per element of the different sub-services for the profile sequences 1 and 2 (see text for additional explanation).

As the PlaneProjection and AffineTransformation times are negligible to the complete performance of the 3Dto2D service, for the profile sequence 2 only the PlaneCut measurements are shown.

Further tests with the datasets GESAMT and PORTA showed that the time per intersecting element for the PlaneCut sub-service is always about 8 to 14 ms depending on the complexity of the cut (number of special cases to process).

## REMARKS

Although the tests show that - beside the plane cut processing – the XSL transformation of the result is the main time consuming operation, this approach allows a great flexibility in supporting different XML and GML related graphic formats provided as outputs, because they are used quite extensively in 2D (GML, SVG, GML derivates).

The given implementation of the service is capable of processing arbitrary planes in 3D space and is not restricted to axes oriented planes such as used in the presented tests. We intentionally did not tune our implementation for that special case so that the presented results can also be expected for cases with arbitrary planes.

## CONCLUSION AND OUTLOOK

Data storage and retrieval of XML data and objects have been examined via a 2D mobile bicycle route planning system and a 3D geological scenario, respectively. We utilized 2D data to compare retrieval of graph algorithms with both native XML and object-oriented data stores. The OODBMS outperformed the XML-based data store for spatial queries such as spatial join and window queries.

When examining at 3D with a simulated data set, the advantage of the page-server for very large complex objects in comparison to the object-based architecture was illustrated. The 3Dto2D geo-service created supports spatial box queries, and operations such as the cutting of a plane with complex geometries, the projection of boreholes to a plane and affine transformation. Within this service the plane cut operation has the longest processing requirements. The XML conversion of the result objects plays a minor role.

In future work, we will extend our research on geological services based on object-oriented data stores towards further types of services. Interesting areas to look at are the distribution of server and client operations, the coupling of the database services with augmented reality (AR) services, and the management of spatio-temporal data for mobile geological services.

## ACKNOWLEDGEMENTS

We thank Prof. Agemar Siehl for kindly leaving us the example data from the project "Geological mapping with GIS based on 3D models" for our ongoing research. Figures 2 and 4 were prepared using the GOCAD software from Earth Decision, Nancy, France.

## BIBLIOGRAPHY

Balovnev O., Bode T., Breunig M., Cremers A.B., Müller W., Pogodaev G., Shumilov S., Siebeck J., Siehl A., Thomsen A., The story of the GeoToolKit – An Object - Oriented Geodatabase Kernel System, GeoInformatica, Vol.8 No. 1, 5-47, 2004.

Beckmann N., Kriegel H.-P., Schneider R., Seeger B., The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. Proc. ACM SIGMOD, Atlantic City(NY), 322-331, 1990.

Breunig M., Bär W., Usability of Geodata for Mobile Route Planning Systems. Proc. 6th Int. AGILE Conference on Geographic Information Science, Lyon, 451-462, 2003.

Breunig M., Cremers A.B., Götze H.-J. Schmidt S., Seidemann R., Shumilov S., Siehl A., First Steps Towards an Interoperable 3D GIS – An Example from Southern Lower Saxony, Germany". Physics and Chemistry of the Earth, Part A, Vol. 24, No. 3 179-190, 1999.

Breunig M., Cremers A.B., Müller W., Siebeck J., New Methods for Topological Clustering and Spatial Access in Object-Oriented 3D databases. Proc. 9th ACM GIS, Atlanta(GA), 10p., 2001.

Dijkstra E.W., A Note on two Problems in Connection with Graphs. Numerische Mathematik (1), 269-271, 1959.

Giguère E., Mobile Data Management: Challenges of Wireless and Offline Data Access. Proc. 17th. Intern. Conf. Data Engineering, Heidelberg, IEEE Computer Society, Los Alamitos(CA), 227-228, 2001.

Guttman A., R-Trees: A Dynamic Index Structure for Spatial Searching. Proc. ACM SIGMOD, Boston(MA), 47-57, 1984.

Kockel F. (Editor): Geotektonischer Atlas von NW-Deutschland, BGR, 4 p., 16 maps, 8 profile fig., 1 CD-ROM, Hannover, 1996.

Mallet J.-L., GOCAD: A Computer Aided Design Program for Geological Applications. In A. K. Turner (ed.), Three-Dimensional Modeling with Geoscientific Information Systems, NATO ASI 354, Kluwer Academic Publishers, Dordrecht, 123-142, 1992.

Shumilov S., Thomsen A., Cremers A.B., Koos B.: Management and Visualization of large, complex and time-dependent 3D Objects in Distributed GIS. Proc. 10th ACM GIS, McLean(VA), 2002.

Siehl A., Cremers A.B., Götze H.-J., Breunig M., Kroll E., Pogodaev G., Schmidt S., Seidemann R., Shumilov S., Thomsen A.: Final report „Geologische Kartierung mit GIS auf der Grundlage von 3D-Modellen" mit thematischem Bezug zum Bündelantrag „Interoperable geowissenschaftliche Informationssysteme" (Abschlußbericht Deutsche Forschungsgemeinschaft DFG Az Si 73/15-2, 1.8.1998-31.12.2001), 49p., 34fig., Bonn, 2002.