

Management and Visualization of Large, Complex and Time-dependent 3D Objects in Distributed GIS

S. Shumilov A. Thomsen A. B. Cremers B. Koos
University of Bonn, Germany

{shumilov, thomsen, abc, koos}@cs.uni-bonn.de

Abstract

This paper presents solutions for architectures of distributed GIS employed for large scale geological modeling in contrast with more traditional GIS. Key technologies are proposed for dealing with complex geological spatio-temporal 3D models. These techniques are then illustrated on a prototype system developed to support interactive work on large models employed by existing geological 3D modeling tools. This prototype has already been successfully applied to the construction of large 3D and 4D models in a geological research project concerning the development of large 3D kinematic models of selected regions within the Lower Rhine Basin. We expect that the enhanced facility in handling large sets of complex data provided by these techniques will enable geoscientists to create more comprehensive and more powerful GIS.

Categories and Subject Descriptors

H.2 [Database Management]: Database Applications - *spatial databases and GIS*; H.3 [Information Storage and Retrieval]: Information Search and Retrieval - *query formulation, retrieval models, selection process*; H.3 [Information Storage and Retrieval]: Systems and Software - *distributed systems*.

General Terms

Algorithms, Management, Performance, Design, Experimentation.

Keywords

3D/4D geological modeling, data selection and retrieval, distributed spatial databases, open GIS, CORBA, Java, VTK, mesh decimation, progressive transmission, VRML, visualization, animation, temporal spatial data.

1. INTRODUCTION

The fast growing number of spatio-temporal applications in fields such as environmental monitoring, geology and mobile communication present new challenges for the development of geo-information systems. In contrast with classical 2D GIS dealing with the Earth Surface, no typical or standard way of

managing and analyzing subsurface 3D and 4D geometry data has yet evolved. On the one hand, the introduction of two additional dimensions causes an enormous increase in the volume of raw data resulting in the need for their compact storage and transmission. On the other hand, the analysis of geo-scientific data by means of different problem-specific tools requires the use of the interactive query processing facilities of the database ranging from complex set-based operations such as spatio-temporal joins to simple selections for on-line visualization [4].

Most modern 3D modeling tools do not have these features and employing them directly for the construction of complex large spatial models is not possible. Therefore, an approach for coupling heterogeneous 3D modeling tools with spatial databases is proposed. This paper summarizes our experience gained during the development of this distributed database-supported GIS used for the construction of complex geological models situated in the Lower Rhine Basin. These models, together with the necessary geological background, are described in the following section 2. The architecture of the prototype we have developed is presented in section 3. It describes all basic components and lists the key technologies used. A more detailed discussion of techniques that allowed us to build an efficient distributed system follows in sections 4, 5 and 6. The techniques for database navigation, visualization and progressive transmission are discussed. We conclude with a performance evaluation and a report of the results achieved.

2. GEOLOGICAL 3D/4D MODELS

Within the frame of a long-term collaborative research project SFB350* at Bonn university, a group of geologists, lead by A. Siehl, developed a number of 3D geometric and kinematic models of selected regions within the Lower Rhine Basin. These models, differing in extension, scale and detail, are: a general, kinematic, inclined-shear model of the Erft block [1], a subsidence and compaction model of the Rur and Erft Block [7], and a kinematic model of the Bergheim area [13].

To gain a deeper insight into the development of the Basin, the kinematic model of the small and intensely faulted region around the Bergheim lignite mine, which has an extent of about 2 km and a maximal depth of 500 m, was developed. The model comprises about 14 stratum layers, disrupted by more than 100 faults. Whereas earlier models of the Lower Rhine Basin, having

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GIS'02, November 8-9, 2002, McLean, Virginia, USA.
Copyright 2002 ACM 1-58113-591-2/02/0011...\$5.00.

* This research has been funded by the German Research Foundation (DFG) within the Collaborative Research Center (SFB 350) "Interactions between and Modeling of Continental Geo-Processes".

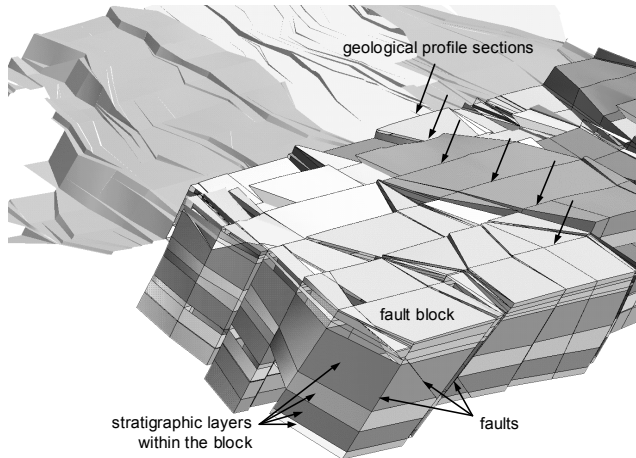


Figure 1: Structure of the Bergheim geometry model

lateral extensions of several tens of kilometers, were modeled using free-form surfaces, the Bergheim model was designed as a structure of several hundred volume blocks, to support relative displacements at fault surfaces and small deformations.

Figure 1 shows the structure of the Bergheim geometry model. The geometry at a given time step is represented using triangulated surfaces and complex fault-delimited volume blocks with a boundary representation. The blocks of the Bergheim model combine three types of triangulated surfaces - namely layer boundaries, faults and artificial boundaries – that follow the original geological profile sections. Its positions are marked by hatched lines at the top of the blocks. The figure also shows by arrows the positions of layer boundaries and places of possible displacements at faults.

The interactive design of the geometry was performed with GOCAD™ [14] 3D modeling program, while the GRAPE [6] library was used for visualization and animation. Time dependency is essentially realized by triangle meshes and vertex coordinates changing in time. Rouby's method of map plane restoration [9] was applied to individual horizons to determine reconstructed states within a time interval from about 18 million years ago to the present day to be in turn followed by a model of subsidence and compaction similar to [7]. During block movements, small inconsistencies at block borders are accepted and taken into account by Rouby's reconstruction method.

Whereas the Bergheim kinematic model consists of a large number of complex volume blocks separated by simple boundary surfaces, in many other cases geological 3D models comprise dozens to hundreds of triangulated surfaces, sometimes consisting of about a hundred thousand triangles or more. The size of the model is multiplied by the addition of the time dimension. Therefore, for subsequent processing in geoscientific applications, the selection of objects and parts of objects according to their geological type and feature must be supported. For example it must be possible to select boundary surfaces of volume blocks by geological type (e.g. "layer").

3. CORBA-BASED INTEGRATED ARCHITECTURE

During our long-term collaboration with the geologists, we identified the need for a flexible distributed infrastructure capable of integrating the existing heterogeneous and highly-specialized

geoscientific tools with spatial database systems. The Base Communication Infrastructure that we developed provides a set of object services and specific objects for the management of persistent interactive user sessions which make use of data servers encapsulating data from a variety of sources [4]. Using standard middleware technologies, the infrastructure is open and independent of any specific application domain. Based on this infrastructure, a prototype of a distributed 3D/4D GIS system was developed to support the construction of complex geological kinematic models (Figure 2). It consists of the components noted in the following sections.

3.1 Spatio-temporal object-oriented database

An object-oriented 3D/4D database developed on top of GeoToolKit [3] provides the management of persistent spatio-temporal objects. The GeoToolKit library is an object-oriented, spatio-temporal 3D database kernel realized on top of the ODBMS ObjectStore® [8]. *GeoToolKit* gives spatial database developers a set of geo-oriented building blocks incorporating efficient methods for ODBMS-based spatial data maintenance and clustering. Currently, *GeoToolKit* offers classes for the representation and manipulation of simple (point, segment, triangle, tetrahedron) and complex (curve, surface, solid) 3D spatial objects. An additional layer of spatio-temporal (4D) classes enables the modeling of spatial objects which change their location and shape in time [10].

3.2 GeoToolKit/CORBA Adapter

The database system is made available to the Common Object Request Broker Architecture (CORBA) environment with the help of a special adapter – GeoToolKit/CORBA Adapter (GTA), which was developed on top of an eXtensible Database Adapter (XDA) - our prototype of a CORBA/ObjectStore object adapter development framework [11]. It provides the necessary primitives and basic mechanisms to define, manipulate, and share persistent data in CORBA. GTA represents an extension of XDA by GeoToolKit-specific classes that are necessary for remote interaction with persistent GeoToolKit objects from the CORBA environment. Covered by the GTA adapter, GeoToolKit can be regarded as a distributed 3D/4D geo-database made available for remote CORBA-compatible clients.

To speed up both the transmission and visualization of 3D objects on the client side, support for progressive spatial data transmission

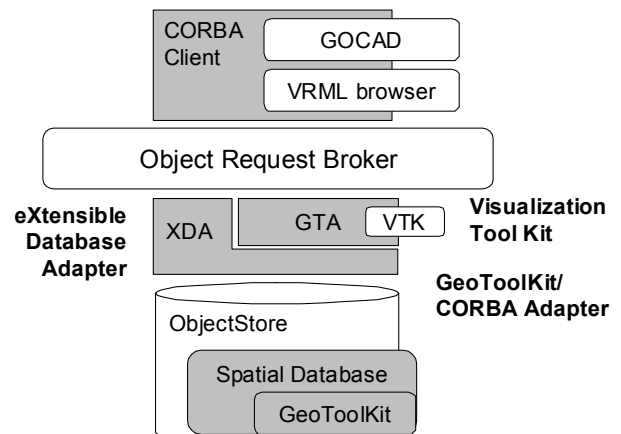


Figure 2: CORBA-based integrated architecture of the prototype

has been implemented in GTA. It reduces the size of the original spatial data by using a special format for their progressive transmission through the CORBA. The algorithm for progressive mesh decimation on the server side was implemented with the help of a freely available computer graphics library *Visualization Tool Kit (VTK)* [12].

3.3 Portable client interface

The integration of geo-scientific applications with the CORBA-based infrastructure was achieved by developing a problem-specific, portable, client interface. The platform-independence of the interface is supported by the Java execution environment. The client communicates with remote GeoToolkit-based databases by CORBA's IIOP protocol and provides querying, previewing and conversion of retrieved objects into application-specific formats (Figure 3). Retrieved objects are kept in a cache for reuse in subsequent queries or for forwarding to geo-scientific applications for further treatment. Currently the client supports GOCAD and VRML output formats [15]. The data retrieved can then be visualized using a VRML browser before they are stored or forwarded to a geoscientific application for further processing. By means of a special extension (Java applet), it is possible to observe and control the movement of selected objects in time. Considering the size of complex spatio-temporal geoscientific models, the client can take full advantage of progressive data transmission when previewing large models at a lower level of detail while avoiding redundancy when higher resolution is required. A more detailed discussion of database navigation, visualization and progressive transmission techniques follows in the next sections.

4. DATABASE NAVIGATION AND QUERIES

A typical session with the client interface might proceed as follows. During work with a 3D/4D modeling application, the user wishes to retrieve geometry objects from a remote database for insertion into his local model. After starting the client interface and establishing a connection with the database, a GUI window with several sections appears (Figure 4). Its listbox windows display possible attribute values, (section A from right to left) which can be used for object selection. Objects and their parts (attributes) are classified according to three basic *geological types*: "Layers" (stratigraphic surfaces), "Faults" (tectonic discontinuities), and "Profiles" (geological sections). The list "Blocks" displays a fourth type, the volume blocks of the model. These *geological types* are typical for many geological

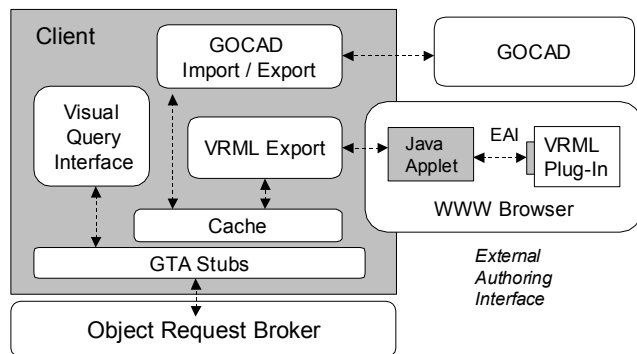


Figure 3: Architecture of the client and extension of VRML-browser for visualization of time-dependent spatial data.

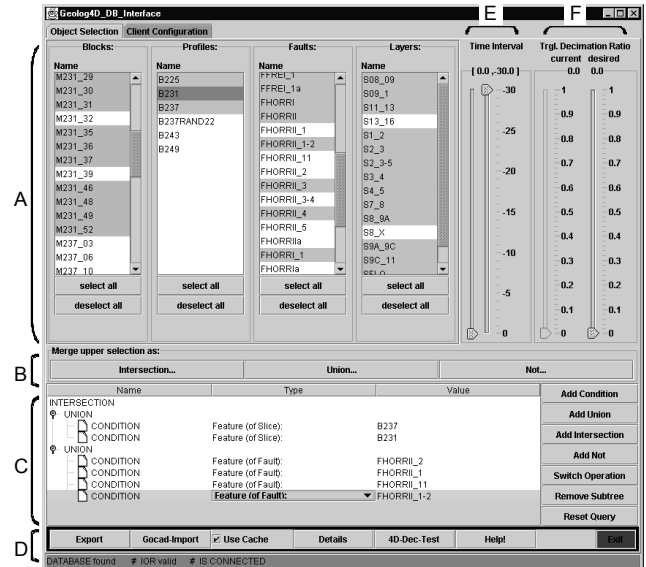


Figure 4: A screenshot of the client's graphical user interface.

applications, and do not limit the usability of the interface to the single Bergheim model. The interface helps the client to navigate through the database following the attributes of selected objects. For example, in Figure 4, A attribute "Profile" with value *B231* has been selected. This selects all objects having the "profile" boundary surface *B231* as an attribute. The selected profile value is marked in dark gray. In addition, we observe that a number of "Fault" and "Layer" values are marked in light gray, signaling the other attribute values taken by the selected objects.

The retrieval of selected objects from the database is performed with help of queries (C), which are activated with a button "Export" (D). Objects selected for retrieval in one of the listboxes (Figure 4, A) can be inserted in queries using one of the buttons "Intersection", "Union" or "Not" (B). The user may modify the selection condition by choosing additional objects in the same or other listboxes. Additionally, activating button "Details" (D) it is possible to investigate the internal structure of retrieved objects or select only their parts for retrieval.

4.1 Complex queries

To define more complex queries, the user may use a query constructor and manipulate the query tree displayed in the lower part of the GUI (C). The query is composed of boolean operations (union, intersection or not) combining simple type-value pair conditions. Figure 4 displays a query consisting of an intersection of two union operations: the first operation selects objects with the attribute "Profile" values *B237*, *B231* and the second selects objects having at least one of the displayed "Fault" values. The user can add new operations to the query by activating corresponding buttons at the right side of the window (C). For example, activating button "Add Condition", the user can define a new condition and combine it with the selected boolean operation. Other buttons permit changing the type of an existing operation and removing either a subtree of the query or the whole query.

In this particular prototype, we did not implement the purely spatial, e.g. bounding box, queries provided by GeoTooKit. In the present application, attribute-based queries provide sufficient problem-specific criteria and navigation facilities. However, spatial queries may be added to the client in the future.

4.2 Time-dependent queries

When retrieving time-dependent geometry objects, the user may specify a time instance t or a time interval $[t_1, t_2]$ by means of two slide rulers at the right side of the query window (Figure 4, E). If a single time instance t is specified, a 3D “snapshot” of the object at time t will be returned. If instead a time interval $[t_1, t_2]$ is specified, a new 4D object resulting from truncation of the original 4D object is returned with $[t_1, t_2]$ as the new interval of validity. If necessary, the object’s states at the boundaries will be interpolated. Obviously t and t_1, t_2 must be inside the original interval of validity. By means of a VRML browser extended by a Java applet, it is possible to observe and control the movement of selected objects in time (Figure 5).

4.3 Queries at multiple resolutions

The volume block objects of the Bergheim model consist of a limited number of surfaces with few triangles. This is not typical for geoscientific applications, where often several dozens of large triangulated surfaces with up to some 100000 triangles must be managed. As an example, a test set of 3D stratigraphic surfaces of the Lower Rhine Basin was investigated (cf. section 6).

To reduce the considerable amount of time spent on data transmission and visualization in such models, methods for progressive transmission were studied [5] and adapted for geoscientific applications. When working with large scale models, it is reasonable first to retrieve queried objects for an overview of a large area at reduced resolution, and later to add detail only to those objects that are finally requested. Therefore, a second group of slide rulers (Figure 4, F) allows the user to specify a reduction factor ranging from 0.0 (no reduction) to 1.0 (a maximal resolution determined by constraints imposed on the decimation). Using this technique, it is possible to significantly reduce the time for retrieval and rendering of large models.

5. PLATFORM-INDEPENDENT VISUALIZATION OF 3D/4D OBJECTS

A preview operation will be often used in practical work before loading the selected data for further processing in specialized geoscientific applications. Basic geological 3D entities turned out to be well mapped on to the VRML representation [2]. Therefore, when a graphical representation is required, the retrieved objects are converted to VRML and visualized using a common VRML-capable WWW Browser. However, VRML offers much more advanced facilities, such as the animation of 4D objects (Figure 5). The animation is defined by a set of keyframes representing 3D geometries for critical moments in an animation cycle. *CoordinateInterpolator* nodes are used to compute all the intermediate positions between keyframes that are necessary for a smooth animation. Whereas well known VRML browsers support animation, they do not provide sufficient control of animations as they evolve. For this purpose, we have studied two different mechanisms: time sensors and an additional Java applet.

TimeSensor nodes provide a simple and natural way to control an animation in VRML browser. Floating point values between 0 and 1, generated by the sensor, are routed to coordinate interpolators, which use these values as keys to calculate the interpolated values. The duration of animation in this case is equal to the value of *cycleInterval* of the corresponding time sensor. An animation of a group of visualized geological objects is started by using the *TouchSensor* node defined in the group and routing the events from the touch sensor to the time sensor. The touch sensor detects

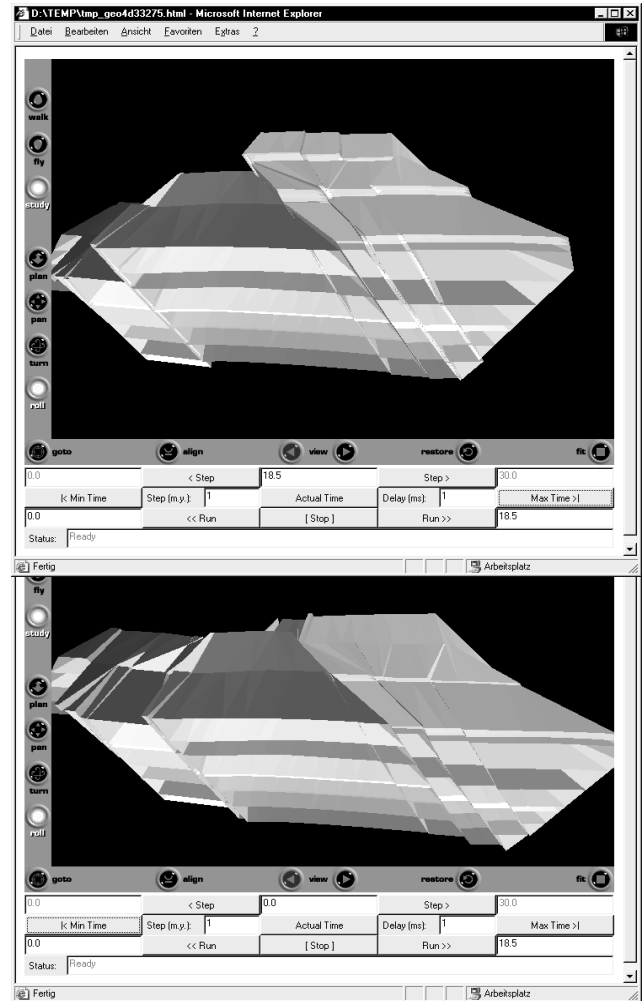


Figure 5: 4D geometry objects at different time steps - above: 18.5 million years ago, below: today - in a VRML browser (in our prototype Cortona™).

mouse clicks anywhere in the group and generates an event that starts the time sensor. However, since touch sensors can only start and stop the animation, this functionality is still insufficient. To control the consistency of complex kinematic modes it is necessary to start and stop the animation at user-defined times. The mechanism of time sensors is unusable in this case because there is no way to limit the interval of key floating point values generated within the cycle.

Therefore, another method is developed to control the visualization process in the second version of our prototype. The original *TimeSensor*'s functionality was emulated by means of a Java applet generating key floating point values and a simple VRML Script forwarding these values to *CoordinateInterpolator* nodes. The applet communicates with the VRML-browser through the *External Authoring Interface* (EAI) and provides a simple user interface that allows the user to control the animation. Interacting with the applet the user can explore the model at particular time steps, start and stop an animation at arbitrarily times, change its direction and time interval boundaries. Figure 5 shows the interface in the window of VRML browser with several tectonic blocks from the Bergheim kinematic model at two time steps.

Employing the automatic generation of VRML objects with the powerful combination of VRML scripts and Java, it was possible to achieve exact control over the multiple synchronized animations, e.g. several blocks moving with different directions and speeds. Actually, using this method, it is also possible to create more sophisticated interpolators than VRML97's built-in linear interpolators, and use Bezier, B-spline, and cardinal curve methods to obtain smooth interpolation with fewer keyframes.

6. MESH GENERALIZATION AND PROGRESSIVE TRANSMISSION

Real 3D data sets - especially time-dependent 3D data sets - are usually quite large. In such cases, the preview operation is not useful because of intolerably slow performance on this task. Our GeoWeb study [2] showed that this was rather due to the slow performance of the network and of VRML-based visualization, than to the software and algorithms applied. Some experiments with GeoWeb made clear that the problem can only be solved by reducing the complexity of the visualized data. In most cases where the preview operation is used, a coarse representation of selected 3D/4D object is sufficient for their unambiguous identification. The *Level Of Detail (LOD)* node, that is typically used in VRML for switching between different levels of details at specified distances, is useless, because it requires that geometries for all resolutions are already present in VRML input. Of course, it is not possible to find a general definition of how many details can be skipped. It depends on the particular kind of data, its semantic and its interpretation. The approximation of a required LOD could only be performed by the development of algorithms specific for every application domain. Therefore, in the general case, it seems more realistic not to try to find the optimal LOD, but to provide a possibility for the user to select the required LOD.

This can be achieved by means of a method for progressive data transmission developed within this project (Figure 6). In the prototype, the functionality for progressive mesh decimation was implemented in the *GeoToolkit/CORBA Adapter (GTA)*. Implementation of a basic mesh decimation operation based on edge collapsing was taken from a freely available computer graphics library - the *Visualization Tool Kit (VTK)* [12]. A bi-directional progressive representation of the 3D meshes transmitted allows a remote CORBA client to gradually improve the resolution, according to the requirements of a particular model and application. Through a sequence of edge collapse operations the geometry to be transmitted is converted into a progressive generalized mesh form consisting of a minimal mesh at a coarse resolution followed by a sequence of refinement operations -

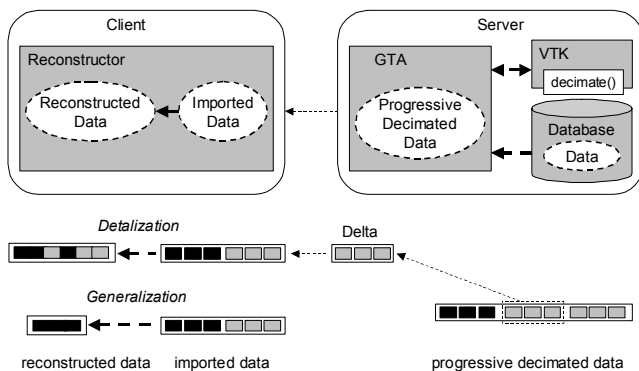


Figure 6: Architecture for progressive transmission.

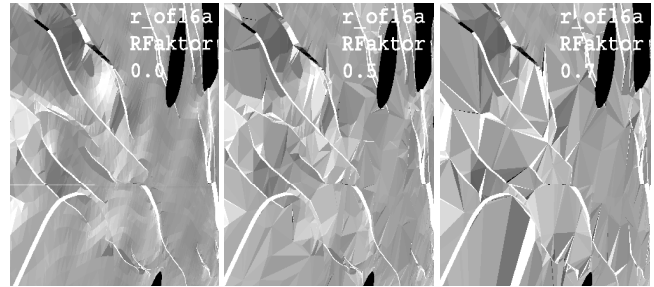


Figure 7: Part of a layer surface with different reduction factors: 0.0, 0.5, 0.7.

deltas (Figure 6, progressive decimated data). The specified maximum approximation error defines the maximal level of decimation that can be achieved. Upon receiving a client's request, first only the minimal mesh is transmitted to the client and used for visualization in VRML. If the user decides that this resolution is not enough, he/she can gradually refine the mesh by requesting the necessary deltas from the server.

6.1 Performance evaluation

Two different datasets were used to test the usability of progressive data transmission. The first test set consisted of real 3D stratigraphic surfaces from the Lower Rhine Basin provided by RWE Rheinbraun AG. The surfaces were modeled at Rheinbraun from different sorts of real geological data: geological section profiles, wells, etc., and converted for subsequent editing with the GOCAD 3D modeler. The crossing faults were filled by additional triangles to obtain continuous surfaces. Every surface contains about 80000 triangles and has an average complexity of geometry that makes them a good test suite for decimation algorithms. Generally, most of the surfaces are flat enough to survive a strong decimation with only minimal visible changes in geometry. Sometimes, however, they contain fault structures with relatively complex geometry. Figure 7 shows such a region in one of the surfaces with different levels of detail. Quite strong deformations in the geometry of faults are visible in the right picture with a strong resolution reduction factor of 0.7. The central picture has a lower level of decimation of 0.5 and looks similar to the original on the left. Tests have shown that this is the optimal level of decimation for our test suite in order to preserve enough details for most of the surfaces.

The second test dataset was a sequence of synthetic 3D surfaces generated with the help of GeoToolkit's 3D surface generator. It is a set of automatically created triangulated 3D surfaces with different realistic relief types ranging from flat to mountains. The size and the geometry of the surfaces are gradually varied from 1000 to 100000 triangles, making them a perfect test suite for testing the speed of the network data transfer. Figure 8 shows the results of our study. Graph A shows the time required to read an object from the remote database. Graph C shows the time required for second retrieval of this item. After the first time request, objects are kept in a cache, which explains the speed-up in accessing these objects a second time. Graph D shows the time required by the overhead for operations with progressively represented data on the client side. Implemented in Java, the array-based representation has an almost linear graphic with max. time about 300ms for 100000 triangles. Finally, graph B shows the sum of the three times required to read a 10% geometry in the first read, a corresponding delta-geometry in the second read and the overhead in geometry conversion.

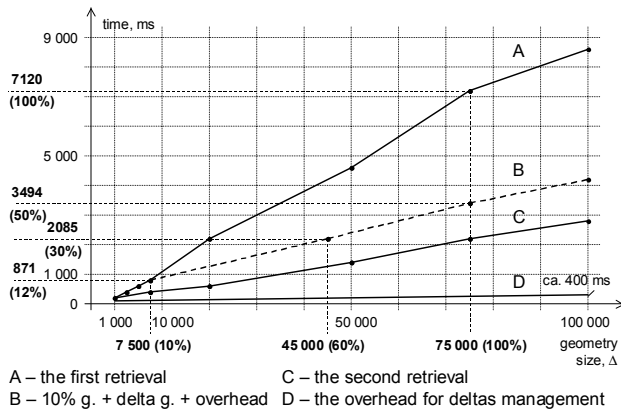


Figure 8: Speed of data transfer in relation to object size.

From these graphs, it is obvious that simplification of the geometry used for pre-visualization can significantly reduce the access time. For example, it is possible to save 88% of the time required for the loading of a surface consisting of 75000 triangles by loading its simplified 10% representation consisting of only 7500 triangles (A). Of course, when decimated to this level, the geometry is very coarse, but this is quite sufficient in most cases where only a rough surface identification is required. In cases where more detail is required, the 10% representation can be improved by loading a corresponding delta. This requires significantly less time than the first loading because of the pre-caching. For example, a second loading of a 50% delta geometry for our surface will require about 1100ms (C). That, together with the time required for its integration in the 10% geometry (~100ms, D), will take only 1200ms. Summarizing processing time for all operations including the time for loading of the first 10% geometry we will receive 2085ms (B). This is still only 30% of the time necessary for the loading of a complete geometry. These results were obtained on a segment of 100mbit Ethernet between a 400Mhz PC with 128mb memory (client) and a Sun SPARC Ultra2 station 2x160Mhz with 1Gb memory (server).

7. CONCLUSION

The paper presents solutions for architectures of GIS employed for the construction of large complex spatio-temporal models. Key techniques proposed for dealing with such models have been illustrated using the prototype of a distributed GIS we have developed. The prototype has been successfully applied for the construction of large 3D and 4D models in a geological research project on the development of the Lower Rhine Basin.

In contrast with traditional monolithic GIS the presented architecture has an open distributed infrastructure, which integrates the existing heterogeneous geoscientific tools into one distributed system, where unique analytical and modeling facilities of these highly-specialized tools are coupled with efficient data storage, navigation and retrieval. This synergy of technologies makes up an essential basis for all GIS tackling with large complex geoscientific models. Originally developed for such systems this technology can be also applied to more traditional applications, e.g. working over connections with low bandwidth.

We expect that the proposed techniques will assist the geoscientists in creating more comprehensive and better validated spatial models, due to the eased handling of large sets of complex spatial data.

8. ACKNOWLEDGMENTS

The concepts realized in the present work have been developed in close co-operation with the Geological institute at the University of Bonn. Our thanks go to Agemar Siehl's research group who provided the necessary application background and numerous useful remarks. Our further thanks go to RWE Rheinbraun AG, Cologne and its chief mine surveyor, Sven Asmus, for providing a series of large real world geometry objects for testing purposes. Finally Tom Arbuckle for proposed valuable feedback and corrections for drafts of this paper.

9. REFERENCES

- [1] R.Alms, O.Balovnev, M.Breunig, A.B.Cremers, T.Jentzsch, A.Siehl. Space-time Modeling of the Lower Rhine Basin supported by an object-oriented database. In *Physics and Chemistry of the Earth*, Vol.23, No.3, p.251-260, 1998.
- [2] O.Balovnev, A.Bergmann, M.Breunig, S.Shumilov. Remote Access to Active Spatial Data Repositories. In proceedings of the *First International Workshop on Telegeoprocessing*, Lyon-Villeurbanne, France, 1999.
- [3] O.Balovnev, M.Breunig, and A.B.Cremers. From GeoStore to GeoToolkit: The second step. In *Lecture Notes in Computer Science*, Vol.1262, p.223-237, Springer, 1997.
- [4] A.Bergmann, M.Breunig, A.B.Cremers, S.Shumilov. A Component Based, Extensible Software Platform Supporting Interoperability of GIS Applications. In proceedings of the *Umweltinformatik 2000 - Computer Science for Environmental protection*, Metropolis-Verlag, 2000.
- [5] H.Hoppe. Progressive meshes. In proceedings of the *SIGGRAPH '96*, p.99-108, Addison Wesley, 1996.
- [6] SFB 256. GRAPE Graphics Programming Environment. Version 5.3, IAM Bonn, 1997.
- [7] T.Jentzsch, A.Siehl. Kinematic subsidence modeling of the Lower Rhine Basin. *Geol. Mijnbouw*, 2002.
- [8] Object Design Inc. *C++ API User Guide*. 2001.
- [9] D.Rouby, Th.Souriot, J.P.Brun, P.R.Cobbold. Displacements, Strains, and Rotations within the Afar Depression (Djibouti) from Restoration in Map View. In *Tectonics*, 15(5), 1996.
- [10] S.Shumilov, J.Siebeck. Database support for temporal 3D data: Extending the GeoToolkit. In proceedings of the *7th EC-GI & GIS Workshop*, Potsdam, Germany, 2001.
- [11] S.Shumilov, A.B.Cremers. eXtensible Database Adapter - a framework for CORBA/ODBMS integration. In proceedings of the *2nd International Workshop on Computer Science and Information Technologies*, Ufa, Russia, 2000.
- [12] W.Schroeder, K.Martin, W.Lorensen. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*. Prentice Hall, 1997.
- [13] A.Thomsen and A.Siehl. Towards a balanced 3D kinematic model of a faulted domain - the Bergheim open pit mine, Lower Rhine Basin. *Geol. Mijnbouw*, 2002.
- [14] Tsurf Inc. GOCAD. 2002. <http://www.t-surf.com>.
- [15] The VRML Consortium Inc. International Standard ISO/IEC 14772-1:1997: The Virtual Reality Modeling Language (VRML), 1997