ISPRS Journal of Photogrammetry and Remote Sensing 117 (2016) 187–205

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

The story of DB4GeO – A service-based geo-database architecture to support multi-dimensional data analysis and visualization

Martin Breunig [a], Paul V. Kuper [a,*], Edgar Butwilowski [a], Andreas Thomsen [a], Markus Jahn [a], André Dittrich [a], Mulhim Al-Doori [b], Darya Golovko [c], Mathias Menninghaus [a]

[a] Karlsruhe Institute of Technology, Geodetic Institute, D-76131 Karlsruhe, Germany
[b] School of Engineering, American University in Dubai, Media City, Dubai, P.O.B 28282, United Arab Emirates
[c] GFZ German Research Centre for Geosciences, Section 1.4 – Remote Sensing, Telegrafenberg, D-14473 Potsdam, Germany

ARTICLE INFO

Article history:
Received 3 June 2015
Received in revised form 2 December 2015
Accepted 23 December 2015
Available online 19 January 2016

Keywords:
Spatio-temporal data modelling
Geodatabase
Geoservices
Geovisualization
n-d topology

ABSTRACT

Multi-dimensional data analysis and visualization need efficient data handling to archive original data, to reproduce results on large data sets, and to retrieve space and time partitions just in time. This article tells the story of more than twenty years research resulting in the development of DB4GeO, a web service-based geo-database architecture for geo-objects to support the data handling of 3D/4D geo-applications. Starting from the roots and lessons learned, the concepts and implementation of DB4GeO are described in detail. Furthermore, experiences and extensions to DB4GeO are presented. Finally, conclusions and an outlook on further research also considering 3D/4D geo-applications for DB4GeO in the context of Dubai 2020 are given.

© 2016 Published by Elsevier B.V. on behalf of International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS).

1. Introduction and related work

Due to the rapidly increasing amount of collected data for all kind of environmental monitoring tasks, research for the efficient management of very large multi-dimensional data sets is likely to become a key issue for the International Society for Photogrammetry and Remote Sensing (ISPRS) in the next decade. In this paper we stress on the handling of 3D geospatial and spatio-temporal data. The strong need for handling spatial and temporal data sets in Geographical Information Systems (GIS) Worboys, 1994 was already noticed by Worboys, one of the pioneers in GIS data handling. The consistency checking of spatio-temporal data organized in graphs has been examined by Del Mondo et al. (2013). In the last years, also the strong need for spatio-temporal data model standardization was recognized (Bohlen et al., 1998; Schmidt and Jensen, 2003). This branch of research has been supported by initiatives of geo-data model standardization and interoperability such as the Open GeoSpatial Consortium (http://www.opengeospatial.org, 2015) and the installation of geo-data infrastructures all over the world. However, best practice examples covering efficient standardized access to spatio-temporal data are still rare. Pouliot and her group (Pouliot et al., 2008, 2010), have made a significant contribution by conceiving a Geological Web Feature Service storing GOCAD® 3D modelling data (Mallet, 1992, 2002) in a standardized Web Feature Service. The group of van Oosterom and Stoter (2010); Döner et al. (2011) is studying the problem of modelling and managing multi-dimensional data including time and scale ("5D approach"). "4D" interpreted as space plus scale has been examined by the group of Stoter and Biljecki (Ohori et al., 2015). The modelling of spatio-temporal data in geological applications from a mathematical point of view has been addressed by the group of Schaeben (Le et al., 2013). In a cooperation between Schaeben's group and Laurent's GOCAD group, Frank and Apel developed an Extensible Markup Language (XML)-based 3d geoscience information system framework (Frank et al., 2003; Apel, 2004). The handling and visualization of 3D data in time for object-oriented database management systems in geoscientific applications (Siehl et al., 1992) was examined by Shumilov et al. (2002). Polthier and Rumpf (1994) made a significant contribution on how to model and handle the geometry and topology of spatio-temporal objects continuously in visualization tools. The group of Schaeben picked up the implementation of DB4GeÓs spatio-temporal data model (Breunig et al., 2010; Bär, 2007) by defining an own mathematical framework of a

* Corresponding author.
E-mail addresses: martin.breunig@kit.edu (M. Breunig), paul.kuper@kit.edu (P.V. Kuper), edgar.butwilowski@kit.edu (E. Butwilowski), andreas.thomsen@kit.edu (A. Thomsen), markus.jahn@kit.edu (M. Jahn), andre.dittrich@kit.edu (A. Dittrich), maldoori@aud.edu (M. Al-Doori), dgolovko@gfz-potsdam.de (D. Golovko), mathias.menninghaus@kit.edu (M. Menninghaus).

http://dx.doi.org/10.1016/j.isprsjprs.2015.12.006
0924-2716/© 2016 Published by Elsevier B.V. on behalf of International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS).

spatio-temporal data model for the geosciences (Le et al., 2013). The problem of efficient access to quickly changing geo-data for expected trajectories is tackled by the group of Seeger (Schmiegelt et al., 2014) in the context of data stream management systems (DSMS) Babcock et al., 2002.

This article is structured as follows: In Section 2 we introduce the reader into the historical roots of *DB4GeO*, our service-based geo-database architecture: Delving into more than 20 years history, we start with the development of *GeoStore* and *GeoToolKit*. In Section 3 we discuss the lessons from the roots that directly influenced the development of *DB3D*, the predecessor system of DB4GeO. Section 4 presents concepts and implementation issues of DB4GeO. Its system architecture is discussed as well as its geometric, topological, and spatio-temporal data model. Section 5 presents experiences with and extensions of DB4GeO such as DB4GeÓs the support of GML data. Subsequently different approaches for the visualization of geo-data such as a DB4Geós WebGL viewer and the insights of a mobile Augmented Reality viewer are presented. Finally, conclusions and an outlook on ongoing research with geo-applications in the context of Dubai 2020 are given.

2. The roots of DB4GeO

2.1. Development strategy

The development strategy right from the beginning was to learn from highly specialized geological applications and to transfer the database requirements to other 3D and 4D geospatial applications. Because it would not have been useful to develop many specialized geo-databases such as GeoStore (Bode et al., 1994), we developed GeoToolKit (Balovnev et al., 2004) that unified the needed kernel functionality in its geometric data model and spatial access methods for all geo-applications that can process geometries based on simplicial complexes. Because visualization tools also use simplicial complexes in their explicit geometric models, the exchange of GeoToolKiťs geospatial data with the geometries in visualization tools is straight forward. From a software development point of view, this was the time of object-oriented programming and modelling. Some years later service-oriented software architectures came up and OGĆs GeoWeb-Services were developed (http://www.opengeospatial.org, 2015). Furthermore, the Java programming language allowed easy web-based programming. In this phase DB4GeO (Bode et al., 1994; Bär, 2007) was developed picking up these new concepts and developments. Also new mobile GIS played a role for DB4GeÓs client development (Bode et al., 1994). Finally, it became evident that a 3D/4D geo-database architecture should not only provide a geometric model, but also a separated topological and a spatio-temporal model.

At the beginning of our research work in 3D and 4D (three spatial dimensions plus time) geospatial databases, we closely cooperated with a geological research group (Siehl et al., 1992; Alms et al., 1998; Breunig et al., 1999) who examined the basin reconstruction of the Lower Rhine Basin. That is why the requirements for client and server database components first came from sub-surface applications and interactive geological modelling. DB4GeO (Breunig et al., 2010, 2013) has its direct roots in GeoStore (Bode et al., 1994) and GeoToolKit (Alms et al., 1998), both developed to support the mentioned sub-surface application in the Collaborative Research Centre 350 at Bonn University, Germany (Sonderforschungsbereich 350).

2.2. GeoStore (1992–1996)

GeoStore (Bode et al., 1994) is an object-oriented information system that has been developed to support the interactive geological modelling (Siehl et al., 1992) of the Lower Rhine Basin and to handle geologically defined geometries such as sections, strata, and faults. Motivated by the different steps of interactive geological modelling in the Lower Rhine Basin

- digitizing profile sections,
- constructing horizons and faults between the profile sections as surfaces in 3D space,
- constructing "real" 3D model by closing the surfaces of each solid.

GeoStore is able to retrieve information from each object of its object model (see Fig. 1). Furthermore, the "switching" between the map interface and the profile section interface is enabled. Fig. 1 presents the user interface showing browser data, 2D map and 3D model of the Lower Rhine Basin containing profile sections, faults, and stratum data.

Fig. 2 shows the object model of GeoStore (figure from Balovnev et al. (2004)). Here the thematic classes and the geometric classes are not strictly separated. This separation between the geometry kernel classes and the application classes has been realised later during the development of GeoToolKit.

The main classes of the thematic component `Stratum`, `Section`, and `Fault` as well as their mapping on a profile section as `StratInSection` and `FaultLine`, respectively, are supporting the first and the second step of the interactive geological modelling.

GeoStore supports three kinds of spatial queries:

1. The browsing of strata, faults and sections: thematic attributes and geometries.
2. Queries on sections: thematic attributes and line geometries of strata and faults located on vertical sections.
3. Spatial queries on triangulated surfaces of strata and faults.

GeoStore also supports spatial integrity checking, e.g. during the insertion of geologically defined geometries to check if surfaces are correctly triangulated without holes or to check discontinuities in their boundaries. Originally being implemented on Oracle®, GeoStore was later re-implemented on top of the object-oriented database management system ObjectStore® with a C^{++} application programming interface (API). Soon it became clear that other groups strived for getting their "own GeoStore" for their specialized applications. This was the starting point for the development of GeoToolKit that was supposed to become a "tool box" to handle geometric 3D and spatio-temporal data (3D plus time).

2.3. GeoToolKit (1996–2005)

In contrast to GeoStore, GeoToolKit (Balovnev et al., 2004) does not provide a thematic geological model, but a detailed C^{++} library of 3D geometric data types to model simplicial complex data types (Egenhofer et al., 1989) such as triangle- and tetrahedron-meshes. This geometric "toolkit" allows the handling of geometrically composed objects as part of geoscientific data models. Fig. 3 shows the object model of GeoToolKit.

GeoToolKiťs object model is based on simplicial complexes composed by 0D to 3D simplices (for an introduction see (Egenhofer et al., 1989). Also analytical objects (line, plane) have been realised (see Fig. 3). In the `Group` class, instances of spatial types can be aggregated. All spatial classes provide geometric operations, functions, and predicates. Physically, the instances of spatial classes are managed in a collection, called `Space`. This container class supports the spatial retrieval of its objects. The retrieval may be accelerated e.g. by using the spatial R*-Tree access method (Beckmann et al., 1990) for the objects of a collection. The vertices

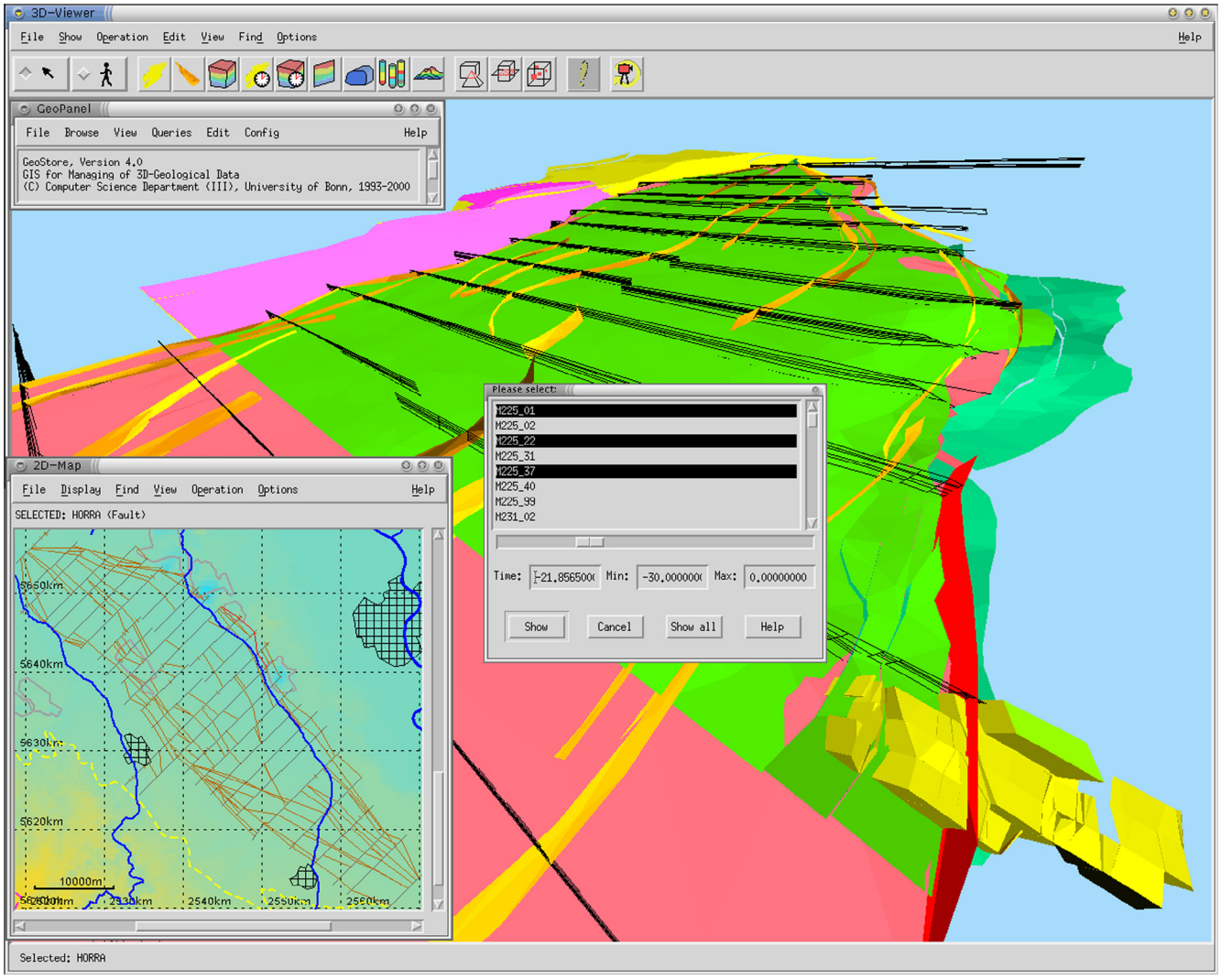

Fig. 1. Browser, 2D map and 3D model viewer of GeoStore.

of the complexes are handled by a point manager that is inserting, deleting, retrieving, and clustering point data. The d-simplices ($0 \leqslant d \leqslant 3$) are managed by a topology manager that updates and clusters d-simplices. Each d-simplex knows the references to its $(d+1)$ vertices and references to its $(d+1)$ neighbours. Besides the R*-tree, GeoToolKit provided an Octree (Tamminena and Samet) index allowing the definition of user-defined spatial access methods. Furthermore, GeoToolKit provided advanced geometric 3D operations such as

- Intersection computation between two triangulated surfaces.
- Projection of line geometries on a vertical section.
- R*-tree supported distance algorithm between two triangulated surfaces being stored in an R*-tree, respectively. This distance function starts at the roots of the R*-trees following the child nodes recursively until the leaf nodes are hit.
- Distance between two convex solids.

Fig. 4 illustrates the algorithm for the distance function following neighboured triangles between two convex solids represented by their triangulated convex hulls.

GeoToolKit has been implemented on top of the object-oriented database management system ObjectStore providing a C++ API. Later, extensions of GeoToolKit based on spatio-temporal simplex classes have been developed (Siebeck, 2003).

3. Lessons learned from the roots

Both GeoStore and GeoToolKit have been designed and realised to support the data management of spatio-temporal geological models of physical processes – mainly deformation – processes in the subsurface for projects in the Collaborative Research Centre 350 at Bonn University, Germany (Sonderforschungsbereich 350). GeoStore and GeoToolKit were employed as 3D/4D-DBMS in combination with various geoscientific modelling software tools: with GOCAD (Breunig et al., 2000) for 3D modelling, with a 4D kinematic modelling software (Alms et al., 1998) based on the GRAPE library (Graphical Programming Environment for Mathematical Problems, cf.(Polthier and Rumpf, 1994), developed by Alms at the Institute of Geology, Bonn, and as an interoperable GIS (Breunig et al., 2000) using CORBA as middleware with the gravimetric modeler IGMAS (Interactive Gravity and Magnetic Application System) and GOCAD. A Java-based interactive client was developed that supported visualization of movements and zooming in and out using progressive meshing via network (Sonderforschungsbereich 350).

This was the starting point to re-use the concepts developed originally in GeoStore and GeoToolKit to handle sub-surface data, to "refresh" the implementation paradigms by new developments such as Open Geospatial Consortium (OGC) GeoWeb Services, and to extend their functionality. The result of this newly designed

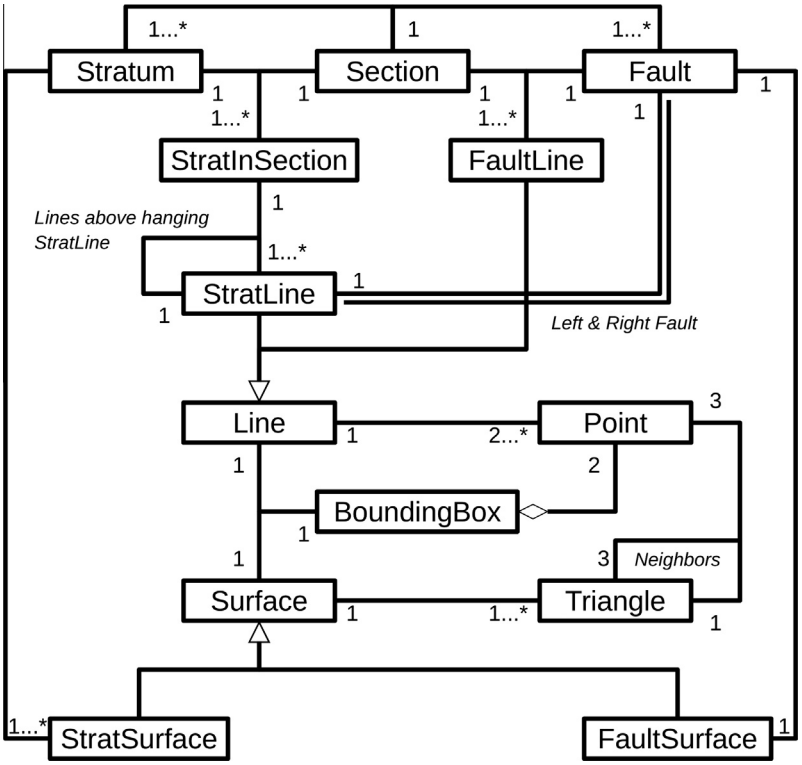

Fig. 2. Object model of GeoStore.

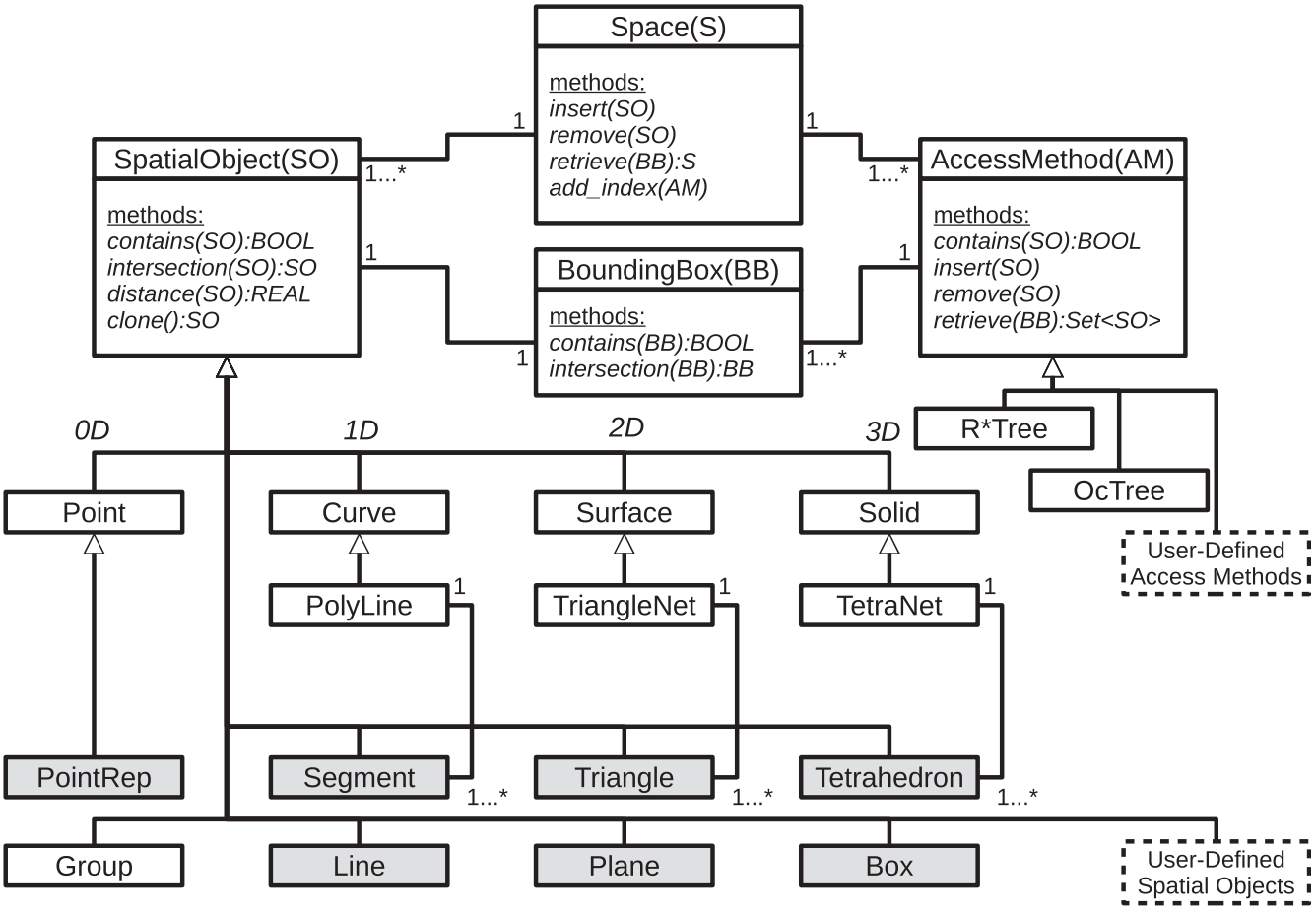

Fig. 3. Object model of GeoToolKit (figure from Balovnev et al. (2004)).

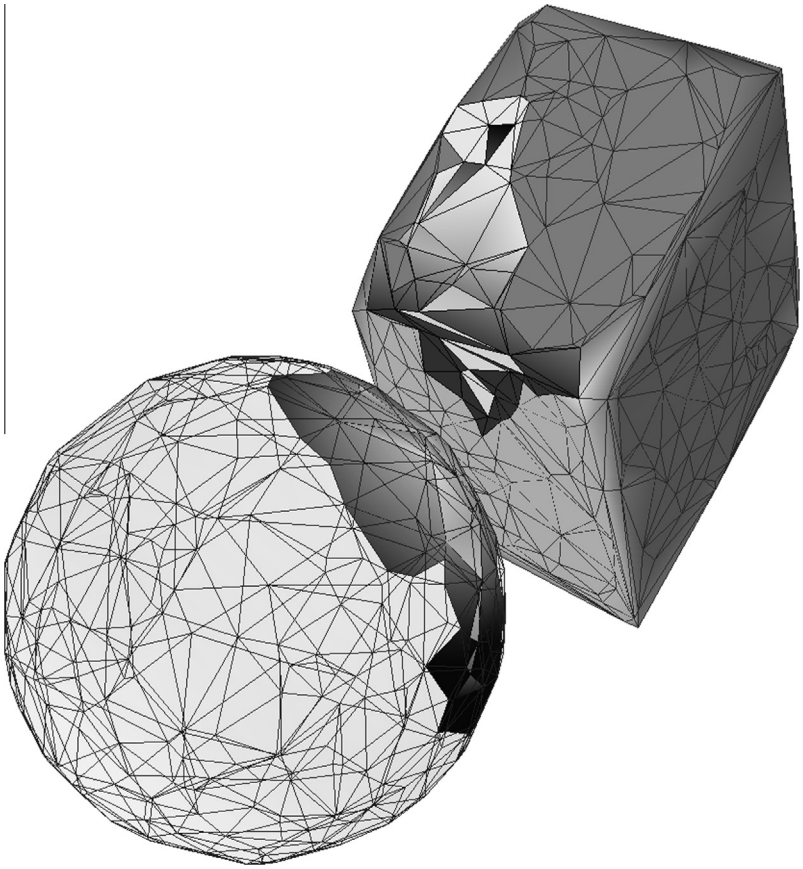

Fig. 4. Sample run of the distance algorithm between two convex solids. Visited triangles have been marked in dark grey (figure from Balovnev et al. (2004)).

system has been called *DB3D* (Database for 3D objects) geo-database architecture. The DB3D implementation (Bär, 2007) is based on a *strict web-service based system architecture* and has been implemented on top of the ObjectStore object-oriented database management system providing a Java API.

Concerning the object models, on the one hand it turned out that the geometric model of GeoToolKit was well suited as a kernel of a geo-database architecture. On the other hand, strictly separated from the geometric model, a topological model had to be added to navigate in topological data structures. Finally, mechanisms to handle spatio-temporal objects had to be added to the kernel functionality.

During the next years the need for unified 3D and spatio-temporal data modelling and data handling especially in the context of Geotechnologies (http://www.geotechnologien.de/index.php/en/index.html, 2015) and Early Warning Systems (http://www.geotechnologien.de/index.php/en/fruehwarnsysteme.html, 2015) became obvious. Discussions with other research groups later resulted in the development of a mathematical model for the handling of geo-scientifically defined spatio-temporal objects (Le et al., 2013). Furthermore, new requirements in city and infrastructure modelling demanded that data management, applications for the visualization, and analysis of 3D sub- and above-surface models should be seamlessly integrated in a unified way.

Java RMI (Remote Method Invocation) (http://docs.oracle.com/javase/8/docs/api/java/rmi/server/package-frame.html, 2015) has been used as communication platform of DB3D. As published in Bär (2007) also a mobile database client has been realised with DB3D that interacts with versioning of 3D geometric objects on the server side. Similar to the GeoToolKit approach, the geometric and topological data model of DB3D is based on simplicial complexes, i.e. on points/nodes, lines/edges, triangle nets/meshes and tetrahedron nets/meshes, respectively. The data model has been extended to simplicial complexes varying with the parameter "time" meaning that every object has its own history of versions.

Finally, DB3D has become the kernel of our new geo-database architecture, called *DB4GeO*. DB4GeO is fully implemented in the Java programming language and is based upon the open source object-oriented database management system db4o (Pouliot et al., 2010) using R*-Tree based spatial access structures.

The lessons learned can be summarized as follows:

- Starting with a concrete application such as geological basin construction helps a lot to design a geo-database architecture, especially to define the requirements and to specify the geometric classes to be handled in the geo-database.
- Providing real 3D functionality in a geo-database architecture means that geometric and topological operations have to be implemented that consider the hull and the interior of the objects. Also composed 3D operations should be provided by complex services, i.e. service chains.
- Providing 4D (3 space dimensions plus time) functionality not only means to handle versions of an object in a geo-database architecture, but to offer a spatio-temporal model that allows to insert and delete new versions as well as to interpolate the geometry of an object between two time steps of the model.
- During decades of geo-software system development it is indispensable that new requirements will come that will update your original data model approach. Thus a consequent separation of geometry, topology, time, and semantics makes life easier for data model evolution.

- Some improvements can be achieved by just waiting ten years for new software paradigms such as new programming languages or OGCs geo-services. However, if doing so, a complete re-design and re-implementation of the software becomes inevitable.

4. DB4GeO – Concepts and implementation (since 2006)

DB4GeO was a beneficiary of the lessons learned listed above. Thus it has a service-based system architecture, contains a geometric, topological and spatio-temporal data model, and provides web-based 3D and 4D geo-services.

4.1. System architecture

Fig. 5 shows the general system architecture of DB4GeO. The foundation of DB4GeO is formed by an object-oriented database management system (OODBMS), e.g. db4o (database for objects) (Paterson et al., 2006). On that fundament, a geometric model was developed to handle simplex-based 3D objects. Such objects can be handled by spatial access structures. Both, the topological model (Section 4.3) and the spatial–temporal model (Section 4.5) are based on the geometric model. GIS clients may access the database via its web services.

4.2. Geometric model: concept and implementation

The underlying 3D data model of DB4GeO is based on simplicial complexes, i.e. points/nodes, lines/edges, triangle nets/meshes and tetrahedron nets/meshes to model geo-objects managed in Physical Geography and other geo-scientific applications which usually have irregular shapes. As well known, simplicial complexes are a special form of cell decomposition, which are often used for the modelling of geo-objects in geoscientific applications. A special feature of the cell decomposition is the consideration of topology

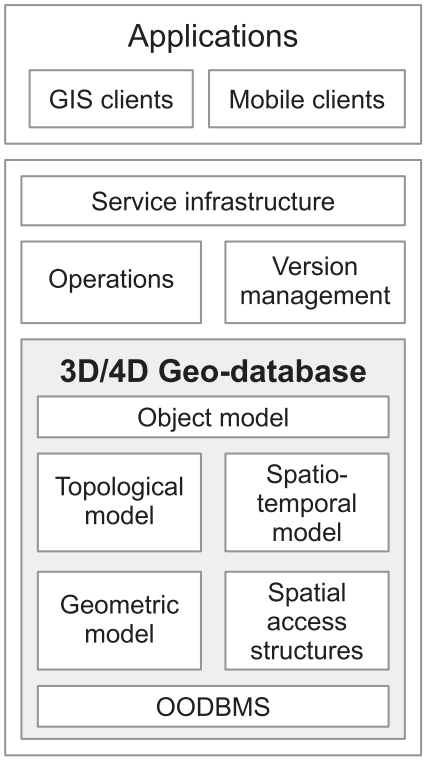

Fig. 5. System architecture of DB4GeO.

(Thomsen et al., 2008). The decomposition is done with cells of the same type. For instance, a surface decomposition can be achieved with triangles (2-simplex) and volumetric decomposition with tetrahedrons (3-simplex). Simplexes in DB4GeO are topologically classified in the usual way, i.e. towards their dimension: 0-simplex = node, 1-simplex = edge, 2-simplex = triangle, 3-simplex = tetrahedron.

Internally these simplexes are managed in four classes: points (class `Point3D`), segments (`Segment3D`), triangles (`Triangle3D`) and tetrahedral (`Tetahedron3D`). Instances of these classes store coordinates of the simplex vertices. Additionally, the `<Simplex>3D` classes can be extended with pointers to their neighbours. Such extended elements are represented by the classes `<Simplex>3DElt`. An aggregation of contiguous non-overlapping `<Simplex>3DElts` of the same type forms a simplicial complex represented by instances of the classes `<Simplex>Net3DComp`.

The spatial part of a 3D object can consist of several disjoint simplicial complexes. In order to account for such cases, the spatial part of a 3D object in DB4GeO is represented by the classes `<Simplex>Net3D`, which can have multiple components (see Fig. 6 for a complete overview).

In order to provide efficient access to elements of a 3D object, an R*-tree and an Octree, respectively (see Section 4.4), is used for each net component. It is built up during the construction of the net component. Each element is enclosed by a minimum bounding box represented by the DB4GeO class `MBB3D`.

Moreover, the core geometry model of DB4GeO includes classes representing planes (class `Plane3D`), lines (`Line3D`) and wireframes (`Wireframe3D`). The latter are used for the intermediate storage of non-simplex results of spatial operations, especially the intersection operation. A wireframe in DB4GeO is simply a collection of nodes with links to their neighbour nodes. With appropriate filtering and meshing, wireframes can be transformed into sets of ordinary simplexes.

Additionally, DB4GeO manages an epsilon value to handle impreciseness caused by the rounding of the object coordinates. If the distance between two points is smaller than the epsilon value, they are considered as identical.

The `Thematic3D` interface provides methods to simplify working with thematic attributes of complex geometry types. It is possible to define a vector of attributes (records) for the individual elements of the complex geometry and the complex geometry itself to store the metadata of the `Object3D` itself. Each record belongs to a `ThematicGroup` that provides the type definitions of the individual record elements.

Furthermore, special methods are provided by each *d*-dimensional component which suit the need of its dimension. In many operations, the specific operations of the *d*-dimensional elements of the components are used. The basic operations (Bär, 2007) are:

- Testing topological relations based on the bounding box.
- Testing topological relations with precise geometry intersections.
- Projection calculations on planes.
- Testing strict equality and geometric equivalence.
- Testing for validity, regularity and well-formedness.
- Calculation of the boundary simplices.
- Calculation of specific parameters (length, area, volume, center, angle, etc.).

At the network level referring to each dimension of the database kernel, the following basic operations are provided:

- Testing topological relationships between networks and simplices of different dimensions.

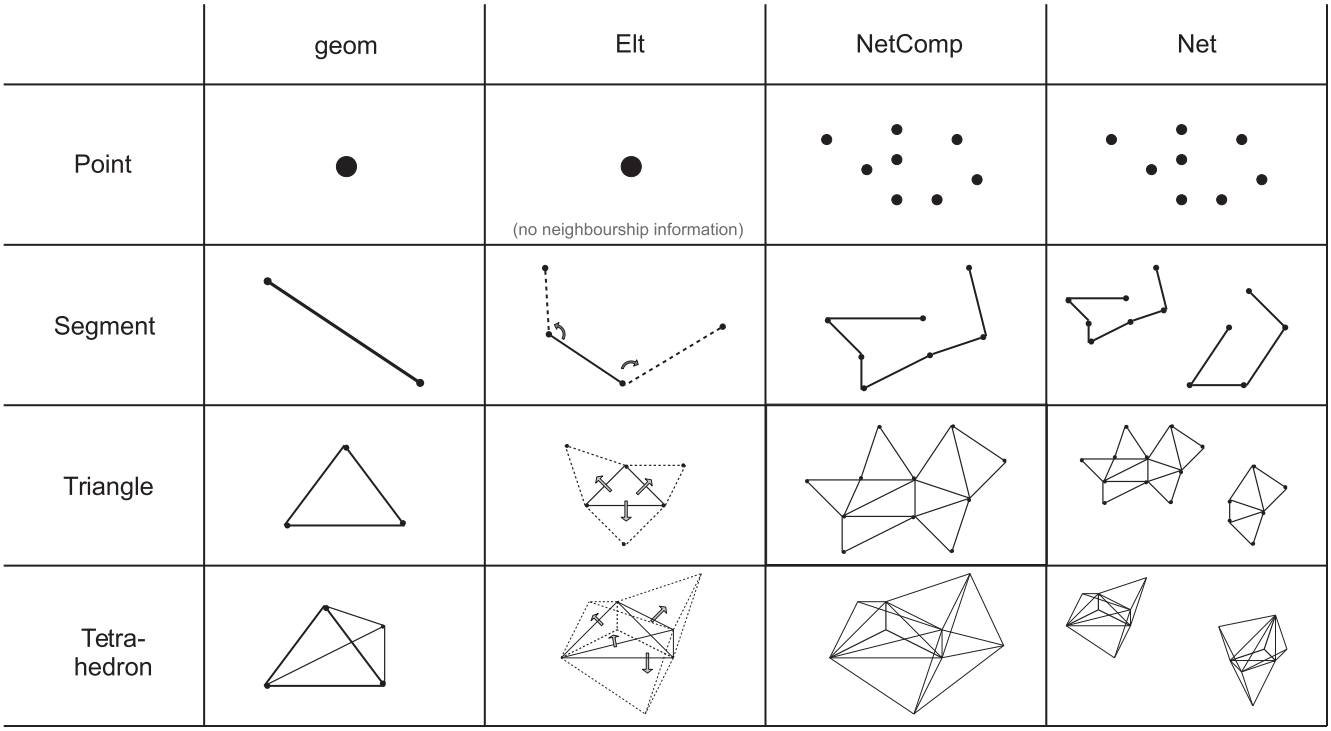

Fig. 6. Geometric data model of DB4GeO.

- Calculating the boundary elements of each dimension.
- Calculating specific parameters (total length/area/volume, number of boundary elements, number of simplexes/edge-simplexes per dimension).

An element is the "atom" of the geometric objects in DB4GeO. It provides methods for the identification in a component, links to its spatial neighbours and is derived from a given simplex type. The management of database objects is taken over by an implementation of the interface `Space3D`. A `Space3D` defines certain information, such as the name of the `Space3D` or its owner. In addition, certain properties of the objects can be defined in a `Space3D`. These properties include the constraints all objects have to fulfil, the reference system and the accuracy constant. Furthermore, index structures according to thematic and geometric properties exist to handle efficient selections of objects. The management of `Space3D` instances is performed by the implementation of the interface `Project`. Projects have a name and a unique ID. Projects are managed by the `DB` class. The `DB` class provides methods to connect to the database and handles the transactions. Following figure summarizes the class hierarchy of the 3D model of DB4GeO from top level `Space3D` (see Fig. 7).

4.3. Topological model: concept and implementation

Simplicial complexes of dimensions 0 . . . 3 are at the heart of geometric modelling in DB4GeO. They provide a consistent geometric as well as topological framework (Thomsen et al., 2008) for elementary objects and operations. For a numerous practical applications, however, other representations seem to be more adequate than 2D or 3D simplex meshes i.e. triangle resp. tetrahedron nets. In City modelling, for instance, a boundary representation using plane rectangular surfaces, or generally regular surfaces, for walls and roofs is adequate for most types of buildings. Moreover, right angles, horizontal and vertical orientation play a most important role. Also, the distinction of levels of detail is required in many indoor and outdoor applications. On the other hand, for geological 3D models consisting of irregularly shaped solids and surfaces that are well represented by simplicial complexes, these are somehow too fine grained for the description of distinct parts or zones or for the assembly of several bodies into larger entities.

Therefore, during the extension of DB4GeO for new requirements the need for a more general spatial model beyond simplicial complexes became obvious. It was decided, however, to build a cellular complex representation on top of the original structure, in order not to lose the advantages of the original simplex model. In this extended spatial model, each cell either consists of a group of simplex cells, or its boundary representation consists of cells of lower dimensions which in turn consist of simplicial meshes. Geometrical constraints on cells such as planarity, orthogonality etc. must be modelled explicitly and verified after each modification of a cell.

In consequence, the geometry of these cellular complexes is completely determined by the underlying simplexes, and only the topology of their aggregation into larger cells must be modelled. This is achieved using a combination of the multi-dimensional Generalized Map (*d*-GMap) Lienhardt, 1991 with the roughly equivalent CellTuple Structure (Brisson, 1993). The modelling of hierarchies of cells follows B. Lévy (2000) Hierarchical GMaps (H-GMaps). These models rely on the theory of algebraic or more precisely combinatorial topology. The topology of a cell structure is represented by an *abstract simplicial complex* composed of "darts" (Lienhardt, 1991) resp. "cell tuples" (Brisson, 1993) referencing *d*-tuples of cells of dimension 0, 1, . . ., *d*. e.g. (node, edge, facet, solid). These are linked by involution transformations, which can be understood as exchange operations on exactly one component ("switch" operation). Involutions of different dimension can be combined into "orbits" characterizing subsets of the abstract simplicial complex. The main advantage of this topological model is its homogeneity over different dimensions and the simplicity of the basic navigation algorithms, with a performance comparable to comparison to classical BRep models (Lévy, 2000). Two disadvantages are an increase in memory consumption, and a problem at the interface with the structural basis: a *d*-simplex is so to speak

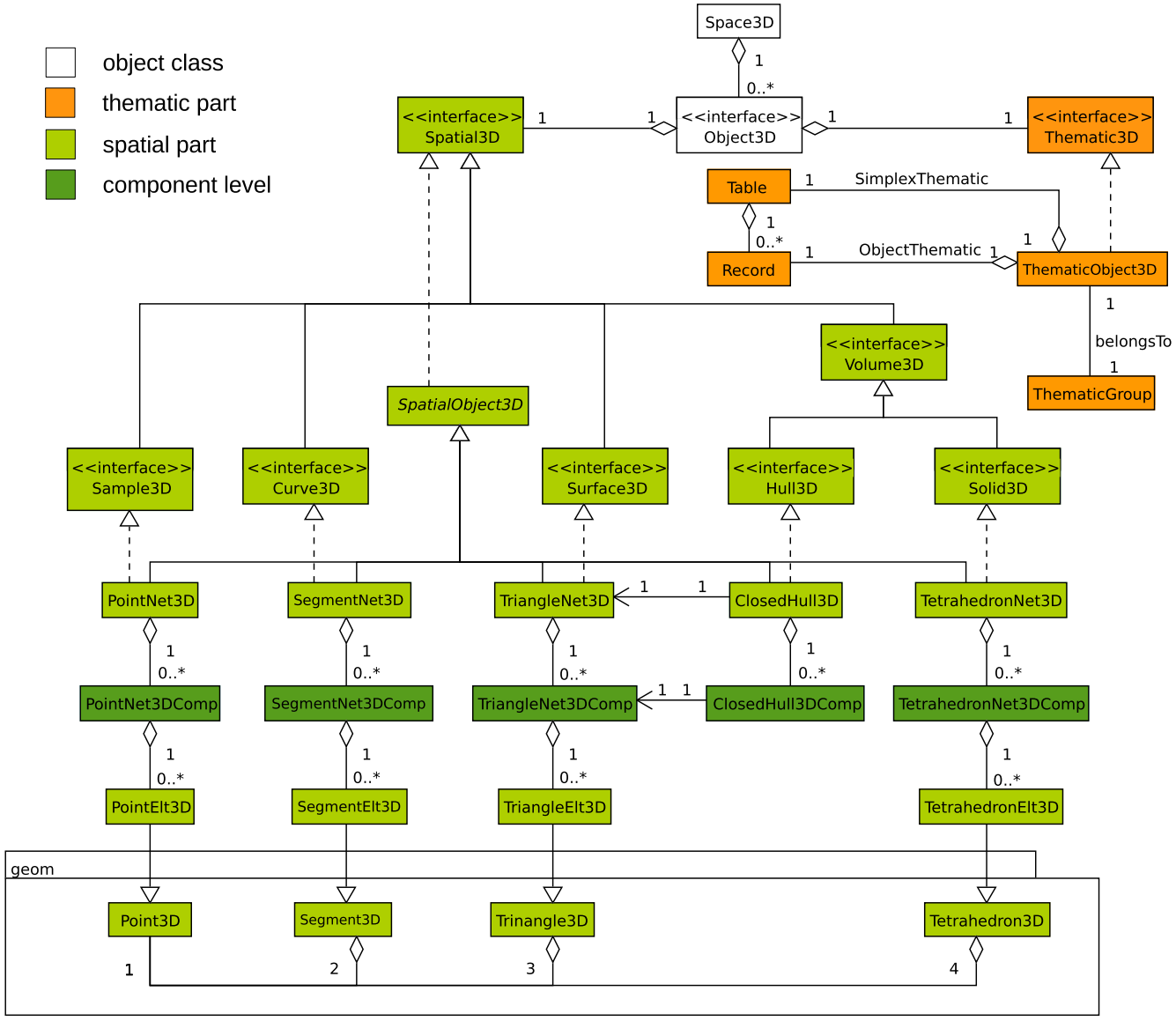

Fig. 7. Class hierarchy of DB4GeO's 3D Model.

the worst performance case of a cell for the GMap representation, as this in turn would imply to represent the d-simplex as a complex of $(d+1)!$ abstract simplexes (darts or cell-tuples). For $d=3$ it would mean to describe a tetrahedron by 24 darts. Therefore the topological GMap representation is restricted to larger compositions of several simplexes. If required, GMap representations at the lowest -simplex level can easily be generated automatically.

Lienhardt's d-GMaps is defined as follows:

A d-dimensional Generalized Map, or d-GMap is a tuple $(D, \alpha_0, \ldots \alpha_d)$ such that

1. D is a set of "darts".
2. The α_i are involutions $D \rightarrow D$, i.e. $\alpha_{\mathrm{I}}(\alpha_{\mathrm{I}}(d)) = d$.
3. For all pairs of dimensions I, j with $|i-j| \geqslant 2$, $\alpha_{\mathrm{I}} \circ \alpha_{\mathrm{j}}$ is an involution, i.e. $\alpha_{\mathrm{I}}(\alpha_{\mathrm{j}}(\alpha_{\mathrm{I}}(\alpha_{\mathrm{j}}(d)))) = d$.
4. An Orbit $\mathrm{Orb}^{d}_{i\,1\ldots ik}(d)$ is defined as the set $\{d' \in D |$ there is a finite family of indices $i_1 \ldots i_n$, such that $d' = \alpha_{i1} \ldots (\alpha_{\mathrm{in}}(d)) \ldots)\}$, i. e. d' can be reached from d by a corresponding sequence of involutions.
5. An Orbit $\mathrm{Orb}^{d}_{i\,1\,\ldots\,\mathrm{ik}\,\ldots\,id}(d)$ characterizes a k-dimensional cell.

Hierarchies of nested cell complexes are modelled topologically following the Hierarchical Generalized Map (H-GMap) model. At each level, down to the simplex level, a cell consists of a group of contiguous cells of the next lower level, and its topology is modelled accordingly.

4.3.1. Implementation

As introduced before, in DB4GeO a unique topology representation for different dimensions and level of details was implemented. In the first step we started to define four types of cells (classes realizing the `Cell` interface, see Fig. 8).

The realised cell types are `Node`, `Edge`, `Face`, and `Solid`, i.e. all d-cells of dimensions d, $\{d \in \mathbb{N} | 0 \leqslant d \leqslant 3\}$. Though, G-Maps are generally capable of modelling an arbitrary amount of dimensions and though there exist algorithms for G-Maps that operate dimension independently, for practical reasons we restrict the model to max. 3 dimensions, since this is actually sufficient for our target applications.

The `Cell` interface defines that a cell has to provide methods that allow retrieving all neighbouring (i.e. adjacent or incident)

cells in a result list. For example, a face object provides lists of all its incident solids, of its adjacent faces, of its incident edges, and of all its nodes. An additional `getBoundary` method always returns the cells that are incident to the given d-cell and that are of dimension $d-1$ (the boundary cells), with the exception of `Node` cell type, which has no boundary cells and thus returns `null` here. Additionally, each concrete cell can also return its cell-dimension (d) and its geometric embedding, which is of DB4GeO simple geo-object type (node is embedded as point, edge is embedded as segment, face is embedded as triangle, and solid is embedded as tetrahedron).

The methods that retrieve the neighbouring cells of a given cell, internally utilize cell-tuple structure/G-Maps in the algorithms. In order to use these concepts in the algorithms, we first designed a `CellTuple` class as a *composition* of all available cell types (see Fig. 9).

A valid `CellTuple` always references exactly (it cannot exist without) one cell of each dimension (i.e. one `Node`, one `Edge`, one `Face`, and one `Solid`). Additionally, a cell-tuple references all its adjacent cell-tuples that can be reached through one involution step (see `alpha0` through `alpha3` references in Fig. 9). On the basis of this topology kernel, it is possible to navigate on top of the topology with high flexibility. Having a reference to an object of type `CellTuple`, means having the possibility to switch to adjacent/incident cells in only two algorithmic steps. For example, having a cell-tuple of a certain node and searching for the adjacent node, means only to follow the `alpha0` reference and then to access the 0-dimensional cell (`Node` of `CellTuple`).

In order to have two-way directions between cells and cell-tuples, we also defined an `anyCellTuple` back reference from `Cell` class to `CellTuple` class (see Fig. 10).

The `anyCellTuple` back reference always allows getting back from the cells to the cell-tuple structure. This is important for most topological queries, since a cell is usually the starting point for a topological request. The `anyCellTuple` back reference returns an arbitrary cell-tuple, giving no statement on its topological position. The returned cell-tuple can then again be used to navigate between neighbouring cells.

The presented topological types are used as the kernel for complex topological querying and editing operations. For example, the involution operations are concatenated to form the more complex topological orbit operations. Due to the presented topological class model, orbits can be defined in an elegant orbit iterator framework that fits into the native Java `Iterator` framework. Orbit iterators are then used in the cell methods that return lists of neighbouring cells.

Motivated by the experiences with GMaps and the need of a better thematic model, the object model and the geometry-library of DB4GeO has been completely revised. The main goal was to establish a meta-structure which allows an easy integration of the GMaps paradigm. In addition, to optimize the thematic model of DB4GeO the focus was on providing an Application Programming Interface (API) that simplifies the implementation of multi-dimensional algorithms. For this purpose a cell system was introduced. Each cell consists of three parts the spatial-, temporal- and thematic-part. This approach is similar to the general approach of the simple- or general-features of the OGC or the ISO 19107. The 3D and 4D geometry model has been merged. All geometry types, such as networks, components, sequences or elements of any dimension are able to be referenced by the spatial part of a cell (see Fig. 11).

In addition, all temporal classes have been moved to their own packages. It is up to the specific cell type how to link the three parts. This enables a large set of different object types. Furthermore, the cells are organized in a tree structure. Each cell maintains multiple children cells. The super cell determines which cell type the child cell is going to be. Which combinations of spatial- and temporal-parts are possible and which standard thematic attributes the cells provide, is set through a `DBMSDefinition` class. A standard implementation of the Database Management System (DBMS) class sets up the first cell hierarchy. E.g. the root node is always an instance of the DBMS class. It manages projects and those in turn spaces. The spaces contain the actual database objects. All of these classes are specializations of the Cell-class (see Fig. 12).

It is enabled that database objects are able to reference not only networks on their spatial part but also components, sequences or elements such as its superclass. In this way DB4GeO is able to adapt dynamically to the application just by creating sub-objects as needed. Special methods will be provided to adjust spatial, temporal, thematic and referential redundancies. The integration of the GMaps module is to be implemented by a specialization of the DB4Object class to determine the structure of the object-tree. An example is shown in Fig. 13.

4.4. Spatial index support

An important aspect of software efficiency in geo-databases is the ability to rapidly deliver results to queries, especially under high request load, i.e. high number of simultaneous requests and for large amounts of data. The user of the DB4GeO API can choose to manage geometric data with an implementation of the R*-tree

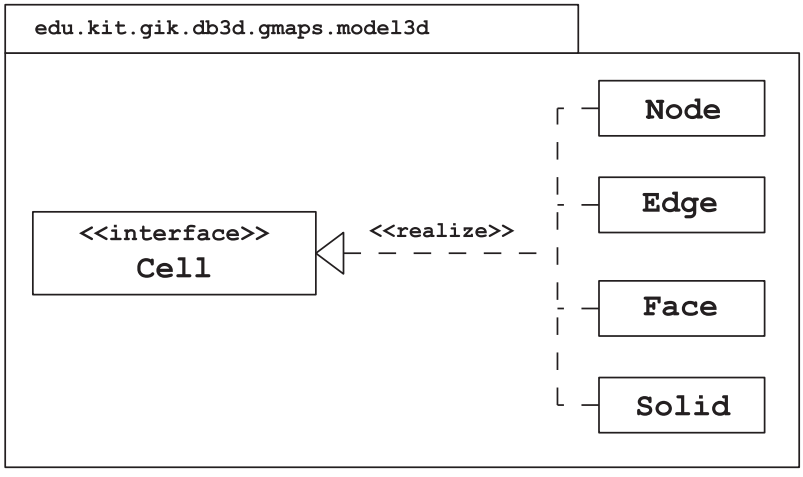

Fig. 8. Cells node, edge, face, and solid realising the cell interface.

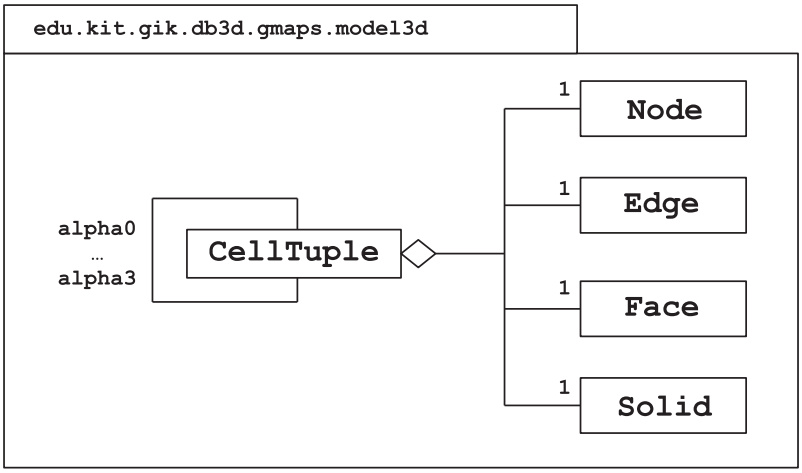

Fig. 9. CellTuple object is modelled as composition of cell objects.

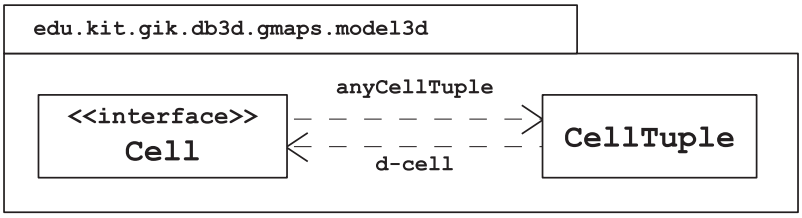

Fig. 10. All cells have a back reference to a cell-tuple.

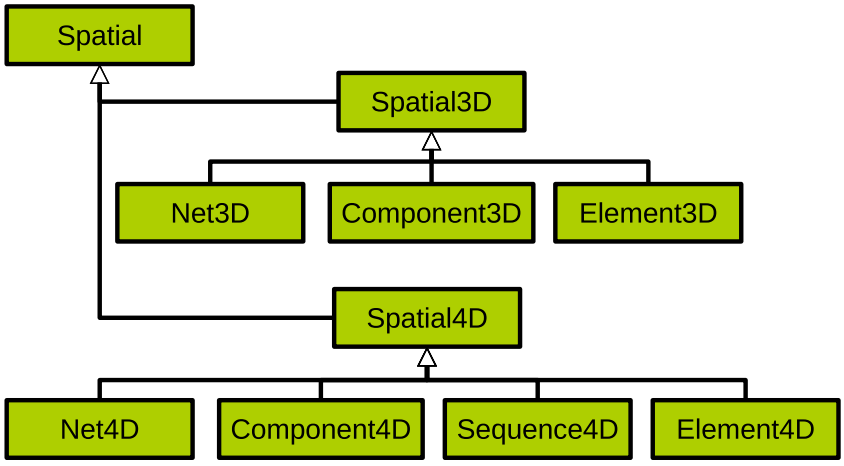

Fig. 11. Spatial class hierarchy.

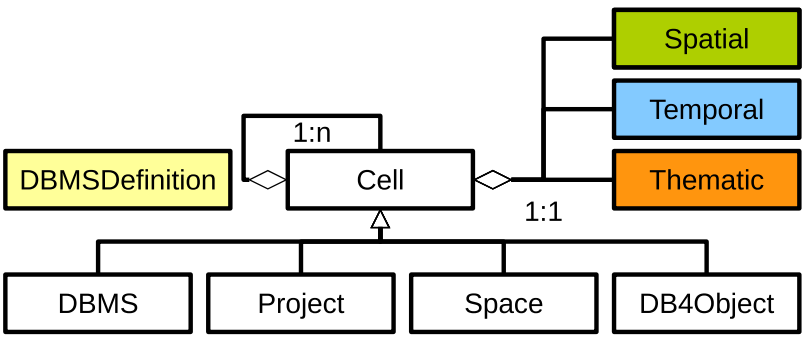

Fig. 12. Cell model of DB4GeO.

(Beckmann et al., 1990), with an Octree (Samet, 1990), and with an own customized spatial index implementation, respectively. DB4GeO exploits the object-oriented approach to gain flexibility in the usage of spatial access methods. The starting point of the DB4GeO spatial index implementation is the `SAM` i(Spatial Access Manager) interface (see Fig. 14).

DB4GeO itself already realises the `SAM` interface through the classes `RStar` and `Octree` (cf. Fig. 14). All classes that realise the `SAM` interface have to provide at least the methods `insert()`, `remove()`, `intersects()`, `getMBB()`, and `getCount()` to be a valid spatial access method. These class methods are used in many other parts of the DB4GeO API to access and manage a `SAM`.

4.5. Spatio-temporal model: concept and implementation

In extension to DB4GeO's 3D model, 4D elements such as `Point4DElement` are integrated. Each 4D element corresponds to two 3D geometries of `Point3D`, `Segment3D`, `Triangle3D`, and `Tetrahedron3D` describing two time steps of the 3D geometry. Furthermore, time intervals are integrated so that e.g. the first time point of the `SpaceTimeElement` corresponds to the first `Point3D` of the `Point4DElement`. All those elements are collected in spatial and temporal sequences such as `Point4DSequence` to model the movement of the object and `TimeSequence` to model the different time steps. Therefore the morphing of real world objects are enabled (see Fig. 15).

The before mentioned model has been optimized to avoid redundancies, i.e. to reduce the amount of stored data per time step. In the new model an efficient, flexible and performant management of continuously changing spatio-temporal objects was developed. For this purpose, three main concepts have been designed and implemented in DB4GeO. An exemplary use of these concepts is shown in Fig. 16.

(1) *Delta-Storage:* The concept of delta storage is implemented for an efficient storage of spatio-temporal objects. Extending a time-dependent geo-object (`Object4D`) by a new time step, this concept ensures that only the data is stored which differs from the preceding time step. Therefore DB4GeO checks for common parts of these time steps and references those, if possible.

(2) *Point-Tube concept:* The vertices of a 4-dimensional object are managed separately from its mesh-topology. Due to this concept, the information of the meshing can be reused across multiple time-steps. This approach saves both storage space and computing power of geometric operations such as interpolation between time steps.

(3) *Concept of pre- and post-objects:* The model of so called pre- and post-objects introduced by Polthier and Rumpf (1994) for continuously as well as discontinuously changing objects has been adapted and implemented in DB4GeO. Therefore it is possible to manage spatio-temporal objects with a changing mesh-topology by the use of pre- and post-objects. A meshing-interval is completed when a geo-object changes its internal meshing and therefore its topology at a particular time step. This interval is completed with a so-called pre-object which adapts the geometry of the new time step and it starts a new meshing-interval with a post-object using the mesh of the inserted time step. Whereas inside of a meshing-interval all movements must be continuous, discontinuous changes can occur at the contact of meshing intervals.

The user of DB4GeO does not have to call these concepts individually, but the database automatically detects which of the concepts can be applied to an existing 4D object during the insertion of new time steps. The class `Object4D` offers the function `addTimestep()` to initialize or extend a spatio-temporal object.

For a useful and user-friendly behaviour of the advanced spatio-temporal model we developed different services, e.g.to provide access to a specific time-step or even an interpolated intermediate step we offer the *4Dto3DService*. Therefore the class `ServicesFor4DObjects` provides the function `getInstanceAt(Object4D, Date)` to create and return an (if necessary) tempo-

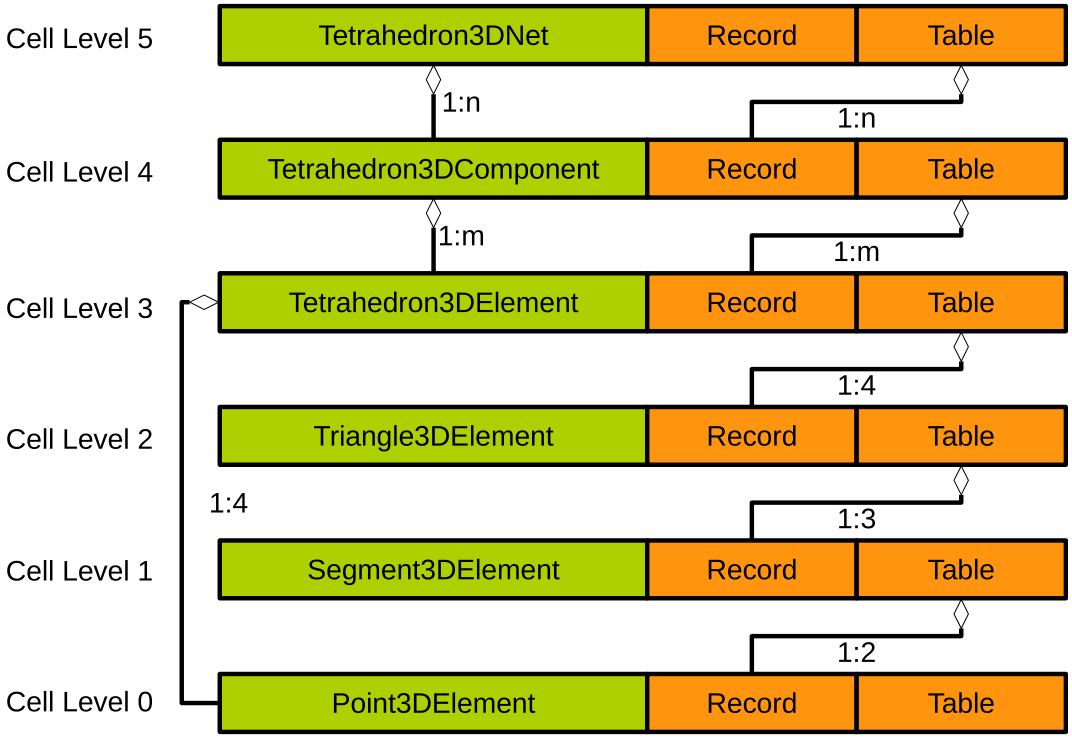

Fig. 13. Structure for topological data handling.

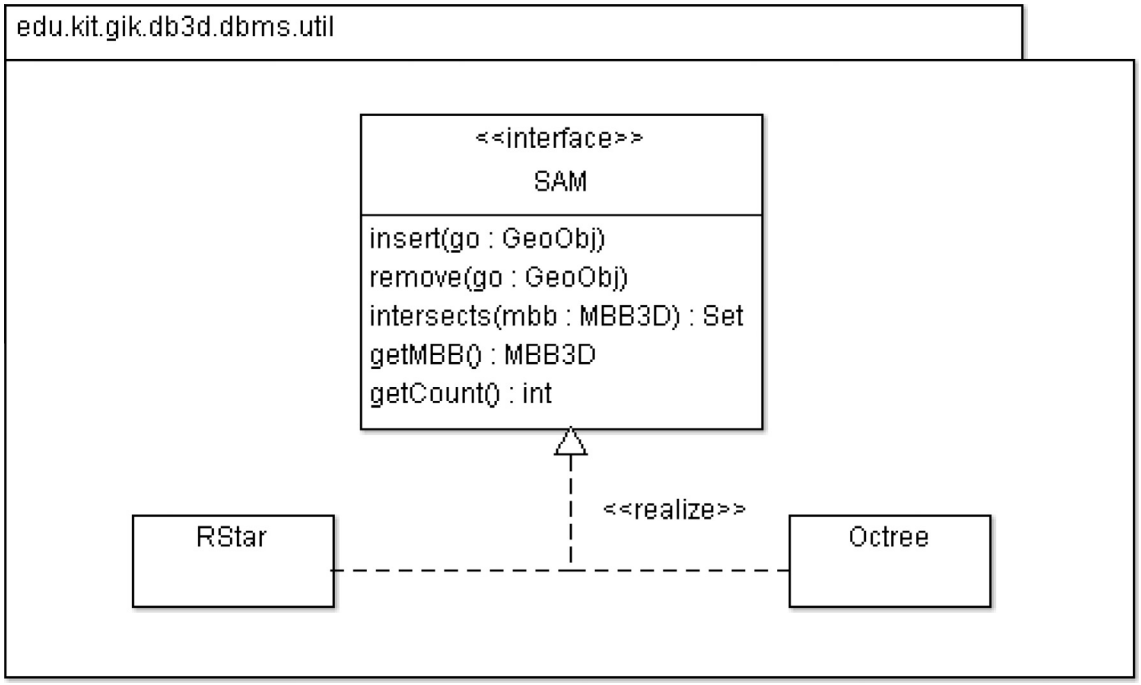

Fig. 14. Spatial index implementation in DB4GeO.

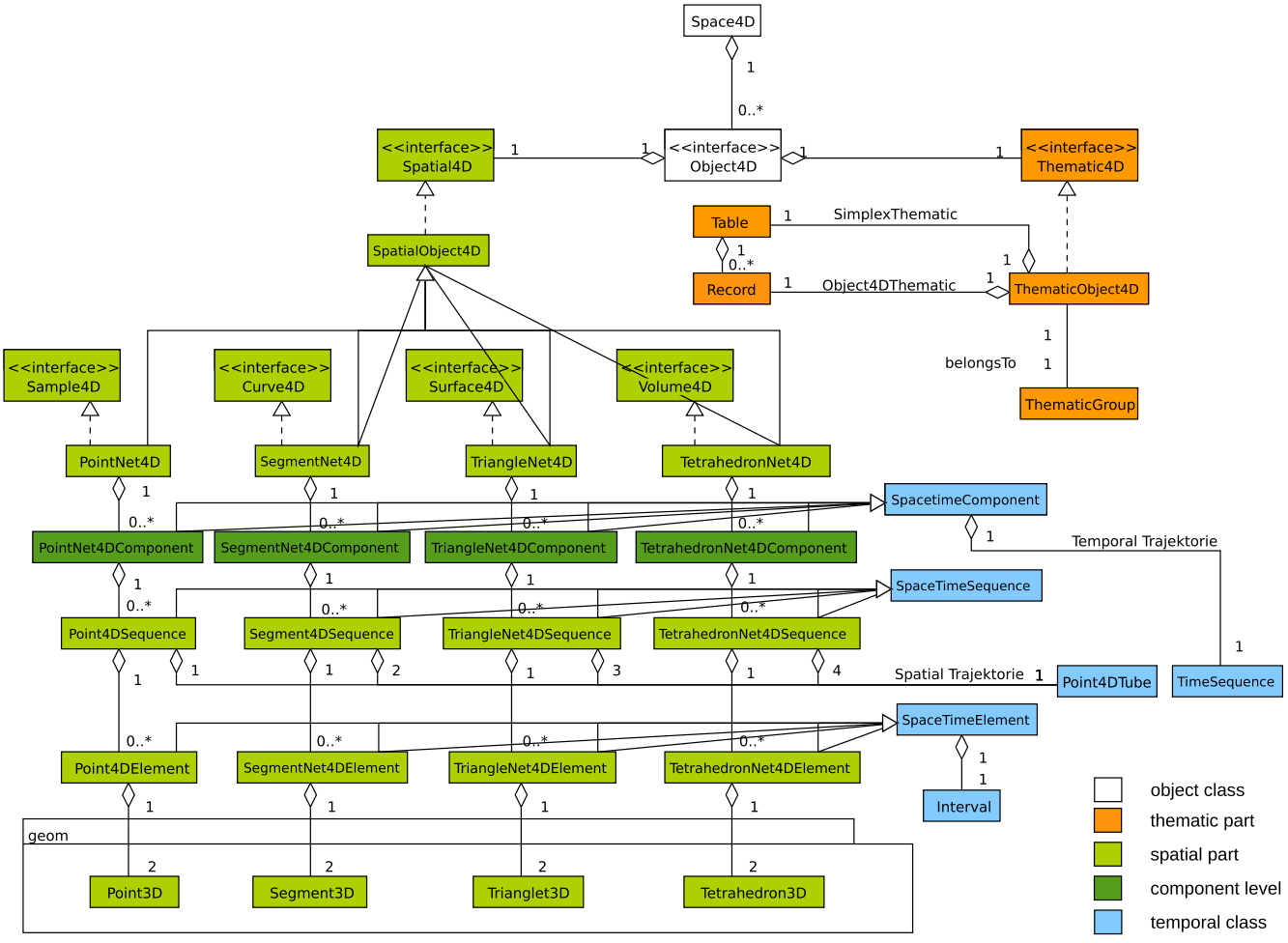

Fig. 15. Class hierarchy of DB4GeO's 4D Model.

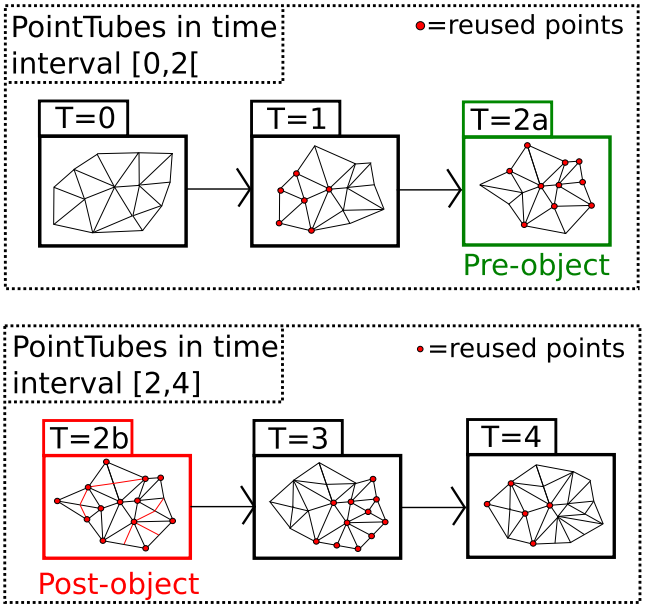

Fig. 16. Point-Tube concept, Polthier and Rumpf temporal model and Delta-Storage implemented in DB4GeO.

rally interpolated 3D object. Other useful spatio-temporal operations for the advanced 4D model shall be integrated in the `ServicesFor4DObjects` class and are part of ongoing research.

4.6. Extending the database by new data types

New data types – regardless of being geometric, topological, or spatio-temporal - can be added to the database besides the already existing data types by programming a new class in the corresponding component, i.e. in the Geometric Model, Topological model and Spatio-Temporal model, respectively. Finally, an interface has to be provided at the object model level. In future, - especially in the context of 3D tracks planning – it is also planned to use a scripting language for quick extension.

5. Using DB4GeO – experiences and extensions

5.1. Input options

DB4GeO already offers a number of importers and exporters which enable combining data from different sources and converting them into various formats. A further step towards geo-data conformity is the support of standards by the Open Geospatial Consortium (OGC). In order to enable providing and receiving data via OGC web services, support for the Geography Markup Language (GML) is required. Similarly to DB4GeO, GML handles objects as features. Geometry can be one of their properties, whereby others may include further information in form of text, images, etc. The geometry model of GML is based on boundary representations and thus is different from the simplex-based model of DB4GeO.

Therefore, it is necessary to create a way to adapt DB4GeO to handle more complex GML geometries. For this purpose, a method for triangulating non-simplex geometries, including concave geometries and geometries with holes, has been implemented in DB4GeO. However, simply substituting a GML geometry with a number of simplices is not enough, because it leads to data loss, e.g. of the attribute information assigned to the GML feature object as a whole. Some of these attributes might reflect geometry-related characteristics of that feature (e.g. area, length or density) and thus be misleading when applied to the elements of a new triangulated surface. Thus an additional level is needed to handle the representation of non-simplex structures typical for GML.

DB4GeO uses its topological model (see Section 4.3) to address these issues. The net level of the model establishes a direct relationship with the underlying simplex geometry, which may be a result of triangulating a more complex GML boundary representation. Various geometric operations offered by DB4GeO can be accessed at this level. The G-Maps-based object level models how the simplices of the net level are aggregated to non-simplex

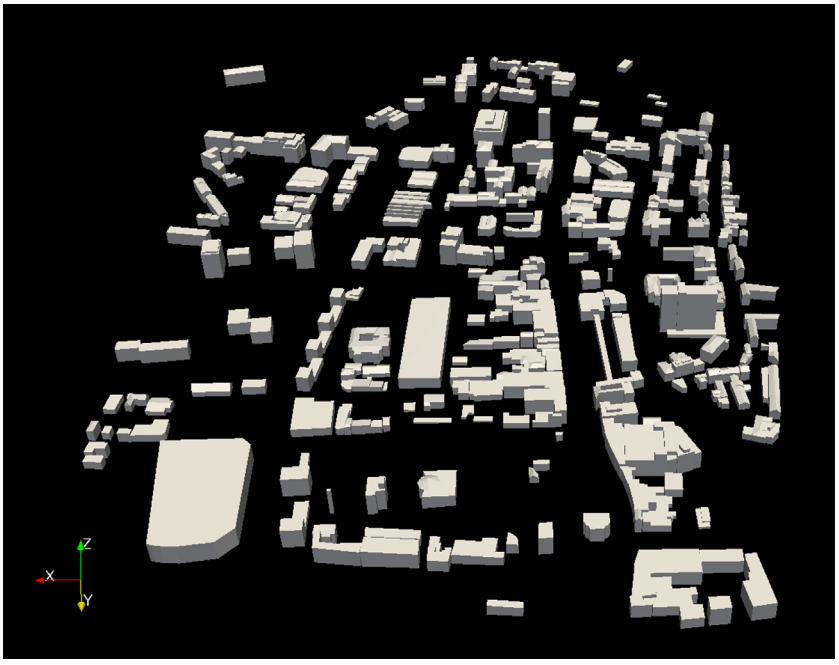

Fig. 17. CityGML data (©Ordnance Survey Great Britain, obtained from citygml.org) imported into DB4GeO and visualized in ParaViewGeo® after conversion to GOCAD® TSurf.

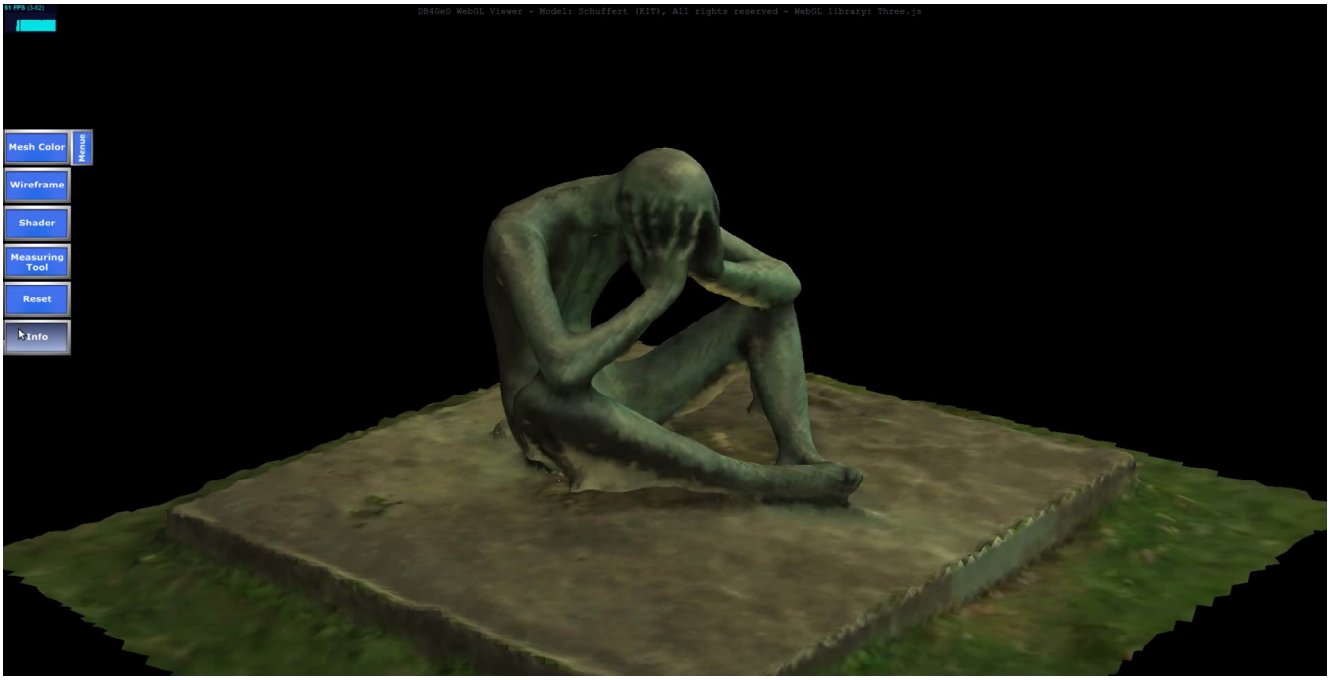

Fig. 18. WebGL model of the statue 'the thinker' [3D model has been constructed by Simon Schuffert, KIT, All rights reserved].

GML geometries. At this level, attributes of the GML feature or of its geometric elements can be assigned to the corresponding faces, for instance, a specific colour or texture can be stored for each wall of the building.

It is possible to import a city model available in the GML dialect CityGML into DB4GeO and then export it into the GOCAD® format in order to combine it with subsurface data (cf. Fig. 17). Vice versa, results of geoscientific modelling in GOCAD® can be imported into DB4GeO, further processed using the geometric functionality of the geo-database and then converted to GML to be provided via an OGC web service. The CityGML importer does currently not support the LoD model of CityGML, and it does not use the temporal functionality of DB4GeO, only static geo-models can be imported.

5.2. Output options

For a quick and straight forward visualization of existing 3D objects, a web-based visualization based on the WebGL library *Three.js* has been implemented for DB4GeO as an external application (WebGL, 2013). Fig. 18 demonstrates an example of a triangle based 3D Model of the statue 'the thinker'. The model (~100.000 triangles) was transformed directly to a HTML file in DB4GeO and can be displayed in all modern browsers (Firefox, Chrome, Safari) without using external plugins due to the use of WebGL.

In this example colour values are also processed and exported. Typical control tools are available for the user. Therefore it is possible to move, zoom and rotate the 3D object. It is also possible to change the transparency of the object directly in the viewer. Currently the visualization service is implemented for 0-simplex, 2-simplex and 3-simplex based 3D models.

5.3. Example of a DB4GeO service

As well known from application fields such as geology and geophysics, the attributes of volumetric objects may not only be attached to the hull i.e. the boundary of the objects, but also to their interiors. Therefore it is obvious that true 3D geo-database services are needed providing a clear interface with 3D functionality, possibly enabling the user to build own user-defined services, e.g. by combining or implementing own 3D geometric and topological database operations. 3D geo-database services may contain 3D spatial operations such as distance operations or intersections between volumetric objects. However, some desktop applications and mobile clients also need 2D results of spatial operations if their CPUs are not efficient enough to compute complex 3D models.

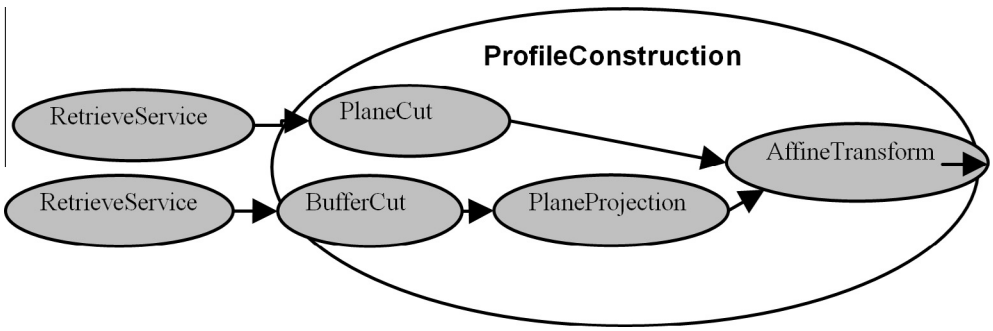

Fig. 19. Operation chain needed for the 3D-to-2D service.

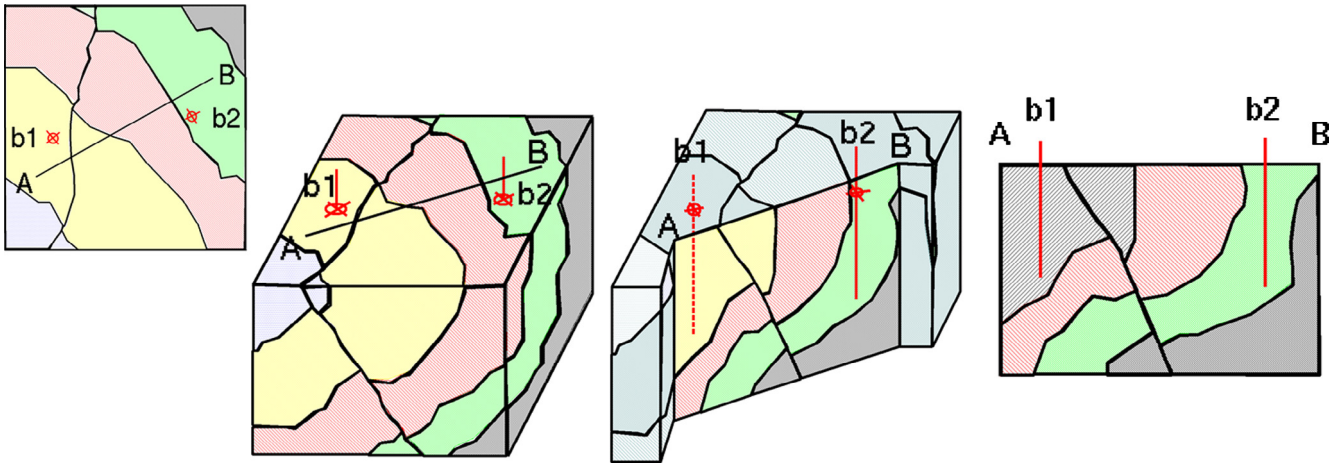

Fig. 20. Example of a geo-database service: 3D-to-2D service.

Fig. 19 shows schematically the operation chain needed to execute the so called "3D-to-2D service" as part of the DB4GeO database kernel. This geo-database service computes a profile section, specified by the user on a 2D map, from a geological 3D block and projects all vertical boreholes around a given distance. In addition to spatial operations, various transformations have to be provided for input and output to allow for further processing of the data in various formats as required by collaborating application programs.

The 3D-to-2D service works as follows. First, the location of a vertical profile section is defined on a 2D map. Secondly, the stored geometries are retrieved from a corresponding window in 3D space. Then the vertical intersection of the section plane with the 3D geometries is computed. Using a 3D buffer surrounding the plane, sample data are retrieved and projected onto the profile plane. Finally, for purposes of visualization and further analysis, the result can be projected onto a 2D vertical "paper plane" carrying the profile section. Fig. 19 describes the chain of operations, which are graphically illustrated in Fig. 20.

The result of this service is a vertical 2D geometry, which can be sent to mobile devices within a mobile client–server database architecture.

5.4. Web service interactions

In cooperation with the DB4GeO Web Feature Service capabilities as data resource, a prototypical app for mobile devices has been developed. The Android Operation System (OS) in version 4.0 (Ice Cream Sandwich) was selected as target platform, and OpenGL (Open Graphics Library) ES 2.0 is used as real-time rendering engine. The purely Java based application provides the following capabilities:

- Loading 3D data from the device's internal storage.
- Loading 3D data from an OGC web feature service.
- Visualizing 3D data interactively via touchscreen gestures (zooming in and out, panning an rotating the dataset in 3D space).

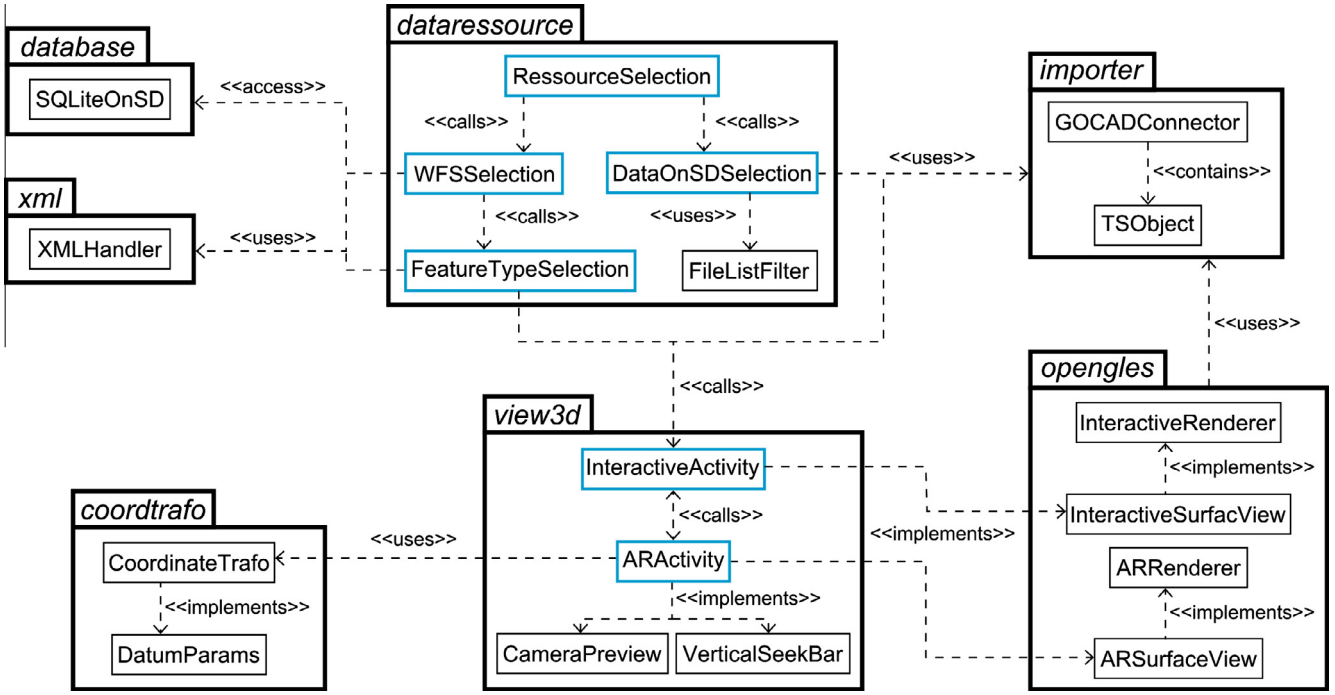

Fig. 21. Package diagram showing the modular architecture.

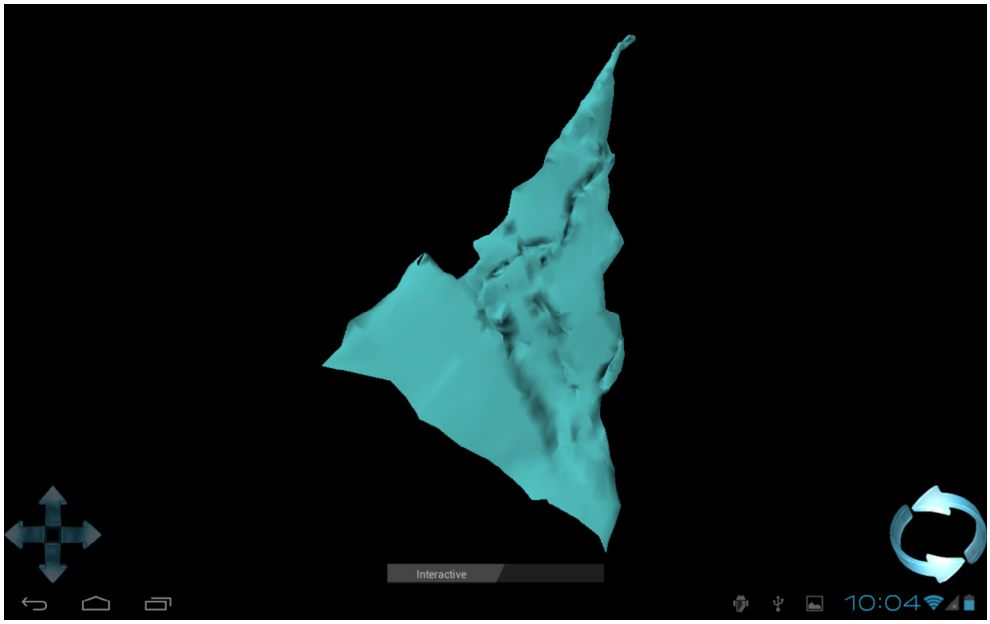

Fig. 22. Interactive real-time visualization with the visualization app.

- Overlaying the camera view of the device with a transparent representation of the data (i.e. *Augmented Reality* technique).

The architecture of the visualization app combines its components in a modular way. Thus, further enhancements can be integrated seamlessly. Fig. 21 shows the different packages with their Java classes and their various interactions. The package *coordtrafo* is responsible to transform the geographical coordinates from the GNSS (Global Navigation Satellite System) sensor, in relation to the World Geodetic System 1984 (WGS84) ellipsoid, to the respective type of coordinates and geodetic datum of the current dataset. Hitherto the implementation of the complex transformation algorithms is not fully optimized concerning accuracy and performance. The visualization app provides solutions for two different types of communication modes: Especially for areas with a weak or no wireless network connection, it provides a convenience solution; the user is able to load data from the device's internal storage. This method, however, is not always suitable for the visualization of several big datasets since the memory capacities of mobile devices are still limited. Therefore, the visualization app implements a second way of loading data into the viewer. The in-built client allows querying an OGC Web Feature Service compliant to version 1.0.0 and 1.1.0. Thus, the large datasets are only loaded on demand and are not stored persistently on the device. An integrated SQLite database table stores services with their corresponding connection information (ID, name, base URL – Uniform Resource Locator).

With the standardized OGC service as data channel, the application only needs to implement the appropriate interface and does not require information on the internal storage structure of the database server. The communication process between client and server is therefore based completely on standardized concepts such as the Hypertext Transfer Protocol (HTTP), Extensible Markup Language (XML) and XML-schemas. This process is a well-structured chronology of requests and responses from the client application to the Web Feature Service.

To support interactive real-time visualization, the visualization app provides operations to zoom in and out, to pan and to rotate the dataset in 3D space in real-time (see Fig. 22). These actions are implemented as the typical gestures known from other touch-screen applications (swipe, pinch, etc.). Thus, a high level of user interaction is ensured, which is necessary to perceive three-dimensional data.

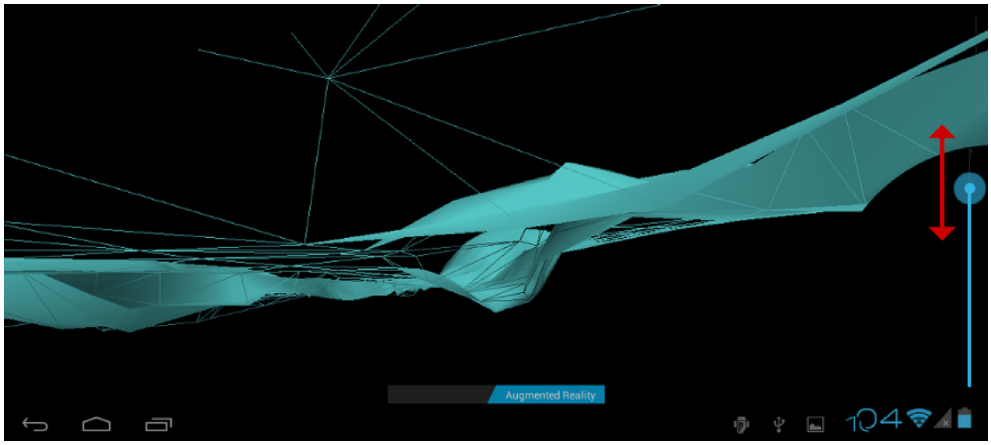

Fig. 23. Vertical slider to zoom "down" in DB4GeO's visualization app.

Fig. 24. Example session for AR modus.

The slider at the bottom of the application view (see Fig. 23) enables the user to switch into the "Augmented Reality" mode (see Fig. 24) that combines digital and real world content by overlaying the camera view of the device with mainly transparent representations of digital models.

The prototype of the app was tested in several successful example sessions with a strongly simplified, digital model of the front of Karlsruhe palace. In Fig. 24, three screenshots of these sessions are presented. The remaining drawbacks of the system are due to the sometimes insufficient accuracy of the GNSS coordinates (between 10 and 20 m). That results in a deficient overlay which is shifted away from the camera view. The second issue is the still limited processing power of mobile devices. The test devices, a Samsung Galaxy Nexus and a Motorola Xoom, both only delivered low frame rates (<15 frames per second) concerning big datasets.

To estimate its position and orientation and hence offering accurate geo-referencing, the mobile device needs internal or external sensors. In the visualization app, the GNSS sensor, a gyroscope and an accelerometer are used determine the positional parameters. These sensors detect the current movements of the user, modifying his position and field of view, in real-time. The outputs of the sensors are fused to benefit from their respective advantages and to balance their drawbacks. Within the fusion process, the GNSS-sensor is responsible for the 3D geographical coordinate, the gyroscope for a high-rate and accurate orientation and the accelerometer data is used to correct the cumulative shift of the gyroscope and to improve the azimuth measurement.

Considered the ongoing and fast progress in the development of mobile devices, however, these drawbacks will lose in impact. In particular the latest implemented GNSS sensors make use of more correction data transmitted through mobile networks. Thus, the positional accuracy can be improved to three meters or even less.

6. Conclusions and outlook

In this article we presented the story of DB4GeO and the different development steps during more than twenty years. The lessons learned from the original software systems GeoStore and GeoToolKit included a careful data models design and the requirement for easy web-based data access. It became evident that a strict separation between geometric, topological, and temporal model enlarges the flexibility to model semantics for applications on top of the geo-database architecture. The lessons learned pathed the way to the development of a service-based geo-database architecture communicating with other geo-modelling software. The system architecture of DB4GeO as well as its geometric, topological, and spatio-temporal model were described in detail. Furthermore, extensions to DB4GeO to model 3D city models and our experiences with real world applications such as a web-based and a mobile visualization app for DB4GeO to support data analysis were presented. As overall experience we can summarize that we would go this way again travelling through an exciting process of changing software development paradigms and upcoming geo-IT concepts with changing members of our team.

In our future research work we will especially optimize the access to hierarchies in the geometric and topological model. Furthermore, we intend to cooperate with Dubai Municipality GIS centre that is to be a central base of GIS data and GIS services for the emirate to provide support for government organisations. The selection of Dubai as a host for EXPO2020 and the decision of Dubai government in 2012 for Dubai to be a smart city confronts GIS experts in Dubai with an exciting challenge: The opportunity to launch over 100 projects which will integrate more than a 1000 government services affecting urban planning, communication, transport and many other services related to all aspects of life in Dubai. This will lead to the creation of new geo-spatial databases and intelligent networks at airports, customs and ports which has a positive effect on local economy. By adding a spatial component to services of government organisations such as Dubai Electricity and Water Authority, Dubai Police, Department of Economic Development, Dubai Municipality, and Dubai Road and Transport Authority (RTA), these services obviously will create added value. These projects are aimed at transforming Dubai into a smart city and improve the quality of life of its inhabitants. The strength of the internet infrastructure and application of GIS technology will enhance the data management by integrating geo-database operations with mapping applications and all their visual benefits as a service to identify and predict events saving a significant cost for public and private enterprises. Examples of such GIS applications are the Makani mapping system (https://www.makani.ae/desktop/?lang=E, 2015), the WOJHATI web based and mobile application developed by the Road and Transport Authority (RTA), and the GeoInfra program (http://wojhati.rta.ae/dub/XSLT_TRIP_REQUEST2?language=en, 2015). Data Scientists recognize the benefits of this process to the development of smart cities in the region with the deployment of mobile 3D GIS applications to support scientific investigations (http://www.gnproperty.com/a/real-estate-news/smart-living-city-dubai-smart-move-towards-future, 2015).

Acknowledgments

The funding of DB4GeO within the DFG grants BR2128/11-1 and BR2128/11-3, the BMBF grants 63064004 and 03G0644A by Projektträger Jülich Research Centre and within the Researchers Group "Computer-Aided Collaborative Subway Track Planning in Multi-Scale 3D City and Building Models" (grant No. BR 2128/14-1 and BR 2128/14-2 by the German Research Foundation DFG) is appreciated. We also thank Dubai Municipality for accompanying the researchers group and the anonymous reviewers for their valuable help to improve the manuscript.

References

Alms, R., Balovnev, O., Breunig, M., Cremers, A.B., Jentzsch, T., Siehl, A., 1998. Space-time modelling of the lower rhine basin supported by an object-oriented database. Phys. Chem. Earth 23 (3), 251–260.

Apel, M., 2004. A 3d geoscience information system framework. PhD Thesis. Faculty of Geosciences, Geotechnology and Mining, University of Technology Bergakademie Freiberg, 94p.

Babcock, B., Babu, S., Data, M., Motwani, R., Widom, J., 2002. Models and issues in data stream systems. In: Proceedings of 21st ACM Symposium on Principles of Database Systems, PODS, pp. 1–16.

Balovnev, O., Bode, Th., Breunig, M., Cremers, A.B., Müller, W., Pogodaev, G., Shumilov, S., Siebeck, J., Siehl, A., Thomsen, A., 2004. The story of GeoToolKit – an object-oriented geo-database kernel system. Geoinformatica 8 (1), 5–47, Kluwer Academic Publishers.

Bär, W., 2007. Management of geo-scientific 3D data in mobile database systems (in German). Ph.D. Thesis, University of Osnabrück, Germany, 166p.

Beckmann, N., Kriegel, H.-P., Schneider, R., Seeger, B., 1990. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. In: Beckmann, N. (Ed.), Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, SIGMOD '90. New York, NY, USA: ACM, pp. 322–331.

Bode, Th., Breunig, M., Cremers, A.B., 1994. First experiences with GEOSTORE, an information system for geologically defined geometries. IGIS1994, Proceedings of the Intern. Workshop on Advanced Research in Geographic Information Systems, Monte Verita, Ascona, Switzerland, Lecture Notes in Computer Science 884, Springer, Heidelberg, pp. 35–44.

Bohlen, M.H., Jensen, C.S., Skjellaug, B., 1998. Spatio-temporal database support for legacy applications. ACM Symp. On Applied Computing, pp. 226–234.

Breunig, M., Butwilowski, E., Golovko, D., Kuper, P.V., Menninghaus, M., Thomsen, A., 2013. Advancing DB4GeO. Progress and new trends in 3D geoinformation sciences. Lecture Notes in Geoinformation and Cartography. Springer, pp. 20–27.

Breunig, M., Cremers, A.B., Götze, H.-J., Schmidt, S., Seidemann, R., Shumilov, S., Siehl, A., 2000. Geological Mapping based on 3D models using an Interoperable GIS. GIS – J. Spatial Inf. Decis. Mak. 13 (2/2000), 12–18.

Breunig, M., Cremers, A.B., Götze, H.-J., Schmidt, S., Seidemann, R., Shumilov, S., Siehl, A., 1999. First steps towards an interoperable 3D GIS - an example form southern Lower Saxony, Germany. Phys. Chem. Earth, Part A 24 (3), 179–190.

Breunig, M., Schilberg, B., Thomsen, A., Kuper, P.V., Jahn, M., Butwilowski, E., 2010. DB4GeO, a 3D/4D geodatabase and its application for the analysis of landslides. Geographic information and cartography for risk and crisis management. Lecture Notes in Geoinformation and Cartography. Springer, Heidelberg, pp. 83–102.

Brisson, E., 1993. Representing geometric structures in d dimensions: topology and order. Discrete Comput. Geom. 9, 387–426.
Del Mondo, G., Rodríguez, M.A., Claramunt, C., Bravo, L., Thibaud, R., 2013. Modeling consistency of spatio-temporal graphs. Data Knowl. Eng. 84, 59–80.
Döner, F., Thompson, R.J., Stoter, J.E., Lemmen, Ch., Ploeger, H., van Oosterom, P., Zlatanova, S., 2011. Solutions for 4D cadastre – with a case study on utility networks. Int. J. Geogr. Inf. Sci. 25 (7), 1173–1189.
Egenhofer, M.J., Frank, A.U., Jackson, J.P., 1989. A topological data model for spatial databases. Proceedings ACM SIGMOD, LNCS 409. Springer, Berlin.
Frank, T., Apel, M., Schaeben, H., 2003. Web Integration of gOcad Using a 3d-Xml Application Server. GOCAD Technical Meeting, Nancy, France, 10p.
http://docs.oracle.com/javase/8/docs/api/java/rmi/server/package-frame.html (last accessed 31.08.2015).
http://wojhati.rta.ae/dub/XSLT_TRIP_REQUEST2?language=en (last accessed 31.08. 2015).
http://www.geotechnologien.de/index.php/en/fruehwarnsysteme.html (last accessed 31.08.2015).
http://www.geotechnologien.de/index.php/en/index.html (last accessed 31.08. 2015).
http://www.gnproperty.com/a/real-estate-news/smart-living-city-dubai-smart-move-towards-future/ (last accessed 31.08.2015).
http://www.opengeospatial.org (last accessed 31.08.2015).
https://www.makani.ae/desktop/?lang=E (last accessed 31.08.2015).
Le, H.H., Gabriel, P., Gietzel, J., Schaeben, H., 2013. An object-relational spatio-temporal geoscience data model. Comput. Geosci. 57, 104–115.
Lévy, B., 2000. Topologie Algorithmique – Combinatoire et Plongement PhD Thesis, INPL Nancy, 202p.
Lienhardt, P., 1991. Topological models for boundary representation: a comparison with n-dimensional generalized maps. Comput. Aided Des. 23 (1), 59–82.
Mallet, J.L., 1992. GOCAD: a computer aided design programme for geological applications. In: Three-Dimensional Modelling with Geoscientific Information Systems. Proceedings of NATO ASI 354. Kluwer Academic Publishers, Dordrecht, pp. 123–142.
Mallet, J.L., 2002. Geomodelling. Oxford University Press, p. 599.
Ohori, K.A., Ledoux, H., Biljecki, F., Stoter, J., 2015. Modeling a 3D City Model and Its Levels of Detail as a True 4D Model. ISPRS Int. J. Geo-Inf. 4, 1055–1075.
Paterson, J., Edlich, S., Hörning, H., Hörning, R., 2006. The Definitive Guide to db4o. Apress.
Polthier, K., Rumpf, M., 1994. A concept for time-dependent processes. In: Goebel et al. (Eds.), Visualization in Scientific Computing. Springer, Vienna, pp. 137–153.
Pouliot, J., Badard, T., Desgagné, E., Bédard, K., Thomas, V., 2008. Development of a web geological feature server (WGFS) for sharing and querying of 3D objects. Advances in 3D Geoinformation Systems. Lecture Notes in Geoinformation and Cartography. Springer, Heidelberg, pp. 115–130.
Pouliot, J., Roy, T., Fouquet-Asselin, G., Desgroseilliers, J., 2010. 3D Cadastre in the province of Quebec: a first experiment for the construction of a volumetric representation. In: Kolbe, Thomas H., Kolbe, Gerhard, Nagel, Claus. (Eds.), Advances in 3D Geo-Information Sciences, Lecture Notes in Geoinformation and Cartography. Springer, Berlin, pp. 149–162.
Samet, H., 1990. Design and analysis of spatial data structures. Addison Wesley, 993p.
Schmidt, A., Jensen, Ch., 2003. Spatio-temporal data exchange standards. IEEE Data Eng., 50–54
Schmiegelt, Ph., Behrend, A., Seeger, B., Koch, W., Seeger, Bernhard., 2014. A concurrently updatable index structure for predicted paths of moving objects. Data Knowl. Eng. 93, 80–96.
Shumilov, S., Thomsen, A., Cremers, A.B., Koos, B., 2002. Management and visualisation of large, complex and time-dependent 3D objects in distributed GIS. ACM-GIS 2002, 113–118.
Siebeck, J., 2003. Concepts for the representation, storage, and retrieval of spatio-temporal objects in 3D/4D geo-information-systems. PhD Thesis, Faculty of Mathematics and Naturlal Sciences at the Rheinischen Friedrich-Wilhelms-Universit, Bonn, 143p.
Siehl, A., Rüber, O., Valdivia-Manchego, M., Klaff, J., 1992. Geological maps derived from interactive spatial modelling. From geoscientific map series to geo-information systems. Geol. Jahrb. A (122), 273–290, Hannover.
Sonderforschungsbereich 350, Report, University of Bonn, Germany.
Tamminen M., Samet H.: Efficient octree conversion by connectivity labeling. ACM Computer Graphics, 18(3), 43–51.
Thomsen, A., Breunig, M., Butwilowski, E., Broscheit, B., 2008. Modelling and managing topology in 3D geoinformation systems. Proceedings of 3D Geoinfo 07, Delft, Advances in 3D Geoinformation Systems. Lecture Notes in Geoinformation and Cartograhphy. Springer, Heidelberg, pp. 229–249.
van Oosterom, P., Stoter, J.E., 2010. 5D data modelling: full integration of 2D/3D space, time and scale dimensions. Geograph. Inf. Sci. Lect. Notes Comput. Sci. 6292, 310–324.
WebGL, 2013. OpenGL ES 2.0 for the Web. <http://www.khronos.org/webgl/> (last accessed 31.08.2015).
Worboys, M., 1994. A unified model for spatial and temporal information. Comput. J. 37 (1), 25–34.