

DESIGN, IMPLEMENTATION AND APPLICATIONS OF 3D WEB-SERVICES IN DB4GEO

M. Breunig^a, P. V. Kuper^a, A. Dittrich^b, P. Wild^c, E. Butwilowski^a, M. Al-Doori^d

^a Geodetic Institute, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany -
(martin.breunig, paul.kuper, edgar.butwilowski)@kit.edu

^b Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany -
andre.dittrich@kit.edu

^c Institute of Environmental Systems Research, University of Osnabrück, 49069 Osnabrück, Germany -
pwild@uni-osnabrueck.de

^d School of Engineering, American University in Dubai, Dubai, United Arab Emirates -
maldoori@aud.edu

Commission III, WG III/4

KEY WORDS: Geo-database, spatial database, database service, 3D service, 3D database, database architecture.

ABSTRACT:

The object-oriented database architecture DB4Geo was originally designed to support sub-surface applications in the geo-sciences. This is reflected in DB4Geo's geometric data model as well as in its import and export functions. Initially, these functions were designed for communication with 3D geological modeling and visualization tools such as GOCAD or MeshLab. However, it soon became clear that DB4Geo was suitable for a much wider range of applications. Therefore it is natural to move away from a stand-alone solution and to open the access to DB4Geo data by standardized OGC web-services. Though REST and OGC services seem incompatible at first sight, the implementation in DB4Geo shows that OGC-based implementation of web-services may use parts of the DB4Geo-REST implementation. Starting with initial solutions in the history of DB4Geo, this paper will introduce the design, adaptation (i.e. model transformation), and first steps in the implementation of OGC Web Feature (WFS) and Web Processing Services (WPS), as new interfaces to DB4Geo data and operations. Among its capabilities, DB4Geo can provide data in different data formats like GML, GOCAD, or DB3D XML through a WFS, as well as its ability to run operations like a 3D-to-2D service, or mesh-simplification (Progressive Meshes) through a WPS. We then demonstrate, an Android-based mobile 3D augmented reality viewer for DB4Geo that uses the Web Feature Service to visualize 3D geo-database query results. Finally, we explore future research work considering DB4Geo in the framework of the research group "Computer-Aided Collaborative Subway Track Planning in Multi-Scale 3D City and Building Models".

1. INTRODUCTION

For Geographical Information Systems predominantly processing geo-spatial data in two-dimensional space, the problem of standardizing data and operations was first addressed 20 years ago focusing on the interoperability aspect at the Open GeoSpatial Consortium (OGC 1994).

However, standardized access to 3D geo-spatial databases also is an important issue to be considered by the 3D GeoInfo Community. Pouliot and her group (Pouliot et al. 2007; 2010) worked on the problem by developing the concept of an open system architecture called Web Geological Feature Service (WGFS). In this work the objective was to realize a closer integration between 3D modeling tools and a spatial database. The architecture uses the client GOCAD (Mallet, 1992; 2002) working together with a database management system (MySQL) via an application server (Apache Tomcat and Deegree). Thus the OGC compliant Web Feature Service and spatial schema are directly used in the architecture. This approach simplifies the transfer from GOCAD data (Mallet, 1992; Mallet, 2002) into a geological database. Without any doubt, an extension of this approach to other geological applications such as interactive geological modelling (Mallet 2002; Siehl et al. 1998) or basin restoration (Thomsen et al., 1998) would be of high interest.

Obviously, the standardized access to spatio-temporal data is an open problem. However, the modelling and managing of spatio-temporal data has been well reflected in different communities (Worboys, 1994; Koubarakis et al., 2003; Schaeben et al., 2003; Balovnev et al., 2004; Le et al., 2013). The group of Peter van Oosterom is even considering scale as separated dimension plus space and time in their model (Van Oosterom and Stoter, 2010). The problem of a standardized nD-data access seems to be how to bring different models together and of course how to implement an integrated approach as a data exchange model. All in all, the problem seems to be more on the efforts of implementation than on general understanding of the data structures to be used in standardized spatio-temporal data infrastructures.

In this paper, approaches are presented on how DB4Geo (Bär, 2007; Breunig et al., 2012a; Breunig et al., 2012b) addresses standardized 3D data access. The paper is structured as follows: In section 2 DB4Geo's REST-based service architecture is introduced followed by the OGC-based services in section 3. In section 4 we explore application approaches before giving a future outlook in section 5 for implementing DB4Geo technology in Dubai and neighbouring countries. Finally, section 6 summarizes the contents of the paper.

2. DB4GEO'S REST-BASED FRAMEWORK

In (Breunig et al. 2012) we presented the external web interface of our REST-based data access and processing framework. In this section, we will present a rough insight into internal implementation aspects of the framework.

To internally handle data access and processing requests that are received from the internet, DB4GeO uses the open source Java Restlet API which is an implementation of the concepts of RESTful web services. In the REST “world of view”, the internet consists – on the highest level of abstraction – of *components* (cf. Component class in Figure 1).

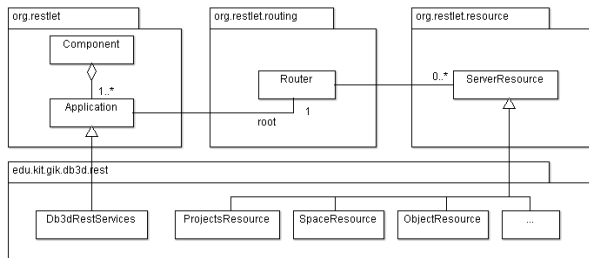


Figure 1: Class diagram showing implementation aspects of DB4GeO's REST-based framework

A component is a container that combines a set of systemically related distinguishable pieces of software that participate in a network (thus, a component can be a web server, a mail server, or a client web browser). Usually, every component resides on a dedicated internet-connected computer but it is also possible to run multiple components on the same machine. Moreover, there is no clear distinction between a server and a client component, since a component can render both roles – e.g. a mail server can also be a mail client if it is configured to collect e-mails from other accounts. The whole DB4GeO REST service infrastructure can be seen as one component.

Individual components of the internet communicate with each other with the help of *connectors* (which represent all properties of a communication link, like port numbers, communication protocols, etc.). In the DB4GeO services, we are currently using only the HTTP protocol for communication which anyway practically has displaced most of the other data exchange protocols of the internet.

A component usually consists of several *applications* which belong together (e.g. a flight reservation component might provide two completely different web applications, one for self-service customers, another one for professional travel agents). The DB4GeO REST service infrastructure currently provides exactly one application which is *Db3dRestServices*. This web application is a data import/export hub for all data types that can be processed by the DB4GeO core API. One advantage of the Java Restlet API is that the DB4GeO services can be easily extended at a later time by new applications and even new components in the code without even the need for reconfiguring the web server.

A RESTful application itself consists of a set of *resources*. A resource is a self-contained part of an application. This resource-centric view of applications is a key element in the REST concept. REST considers an application as a collection of referenceable resources with a unique Uniform Resource Identifier (URI). In the case of DB4GeO for example, we need to make several resources available to provide a useful application. These resources are – to name a few – the *ProjectsResource* (that contains general information on

all the available projects in the component), the *SpaceResource* (that contains property information on a certain geometry space), or the *ObjectResource* (which actually contains the information on the internal structure of a geo-object, like its property values and its actual geometry). An application points to exactly one root router that performs the routing from URI to the registered resources of the application. The resources are simply attached to the router by providing a URI schema in combination with a class instance of a resource. In our case, the resources are attached to the root router of the *Db3dRestServices* application:

```
...
router.attach("/projects", ProjectsResource.class);
router.attach("/projects/{projectname}/{spacename}",
    SpaceResource.class);
router.attach("/projects/{projectname}/{spacename}/
    {objectname}", ObjectResource.class);
router.attach("...", ...class);
...
```

Each of the referred application resources can be accessed individually through its unique URI.

Finally, every resource can have several *representations*. In a web application setting, it is probable that different types of clients want to access the data of the application server. Different types of clients will also need different types of representations of the same resource. For example, a web browser would prefer an HTML/XHTML representation of the aforementioned DB4GeO projects resource to directly display a list of available projects in the browser. Instead, a 3D geo-modelling client like GOCAD would prefer a geo-object resource to be represented as a GOCAD file in order to directly visualize the 3D model of the geo-object. This concept of multi-representation resources leads to an ongoing negotiation of exchange formats between clients and servers which in our case brings great flexibility into the web of geo-data.

3. OGC-BASED WEB SERVICES

The implementation of the first basic Web Feature Service in DB4GeO uses the REST module (see Figure 2).

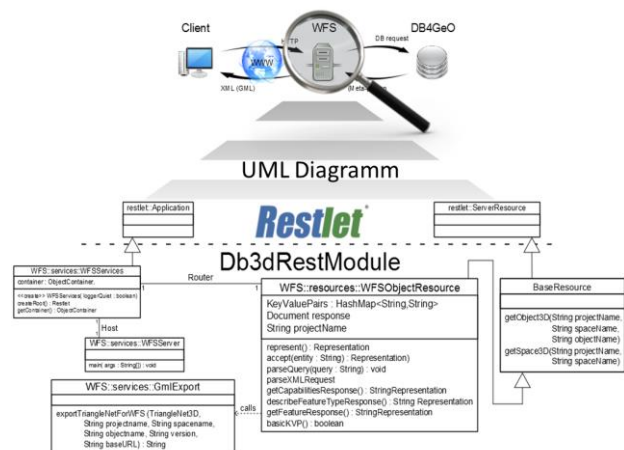


Figure 2: UML-Diagram of DB4GeO's Basic-WFS

3.1 Web Feature Service

To prepare the essential process in the source code, at first the class *WFSObjectResource* is extended by the method *getTransactionResponse()*. The XML-document, which is delivered via POST-request and which contains GML

information of the to be saved object, is being prepared here for further processing. For this purpose the method `parseXMLRequest()`, which overtakes the parsing of the XML-documents transferred via POST, is extended, so that the method `getTransactionResponse()` can be provided with the necessary information.

To ensure the essential server functionality, the classes `OGCServer` and `OGCServices` are outsourced to a new package in the Java Project `Db3DRestModul` (from `WFS.services` to `OGC.services`). Figure 3 shows the connection of the described classes between `RESTlet` and the underlying object-oriented database management system `db4o`.

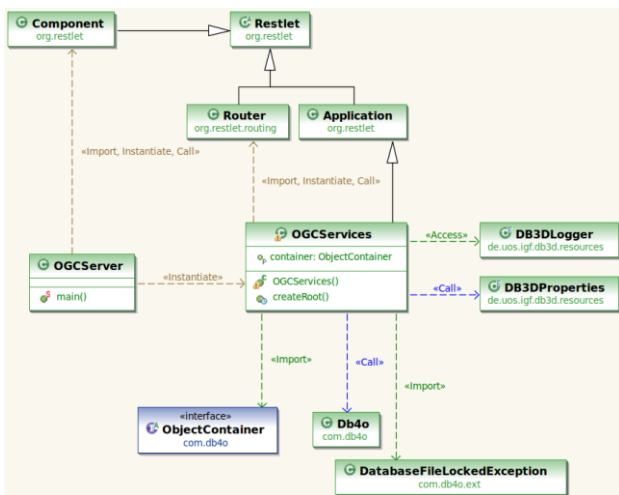


Figure 3: Connection of the classes `OGCServer` and `OGCServices` between `RESTlet` and `db4o`

The class `OGCServer` starts the server via the `RESTlet` class `org.restlet.Component` on `http://localhost:8182`. It instantiates the class `OGCServices` where it is then added to the server as a component of the `org.restlet.Application`. It also takes over the routing of the server for incoming requests and forwards them (either at the WFS or at the WPS). The class `OGCServices`, which inherits from `org.restlet.application`, configures the database and opens a connection between Web services and the database. An example of a `DB4GeO` database will be created and filled with a test file (e.g. “contains the Space Test3D which contains the Object O1”).

Extension by various output formats

To ensure the support of different output formats, the methods `getGocadResponse()` and `getWebGL()` call the methods `writeGocad3D()` or `writeWebGL()` in the class `BaseResource`. The requested object will be passed and the user receives either a `GOCAD`-file or a `WebGL`-file as output.

Implementation of `getTransactionResponse()` method

In the current implementation of the `WFS-T`, first an XML-document is delivered via the `WFS` (via `POST`-request). For this reason the method `parseXMLRequest()` is extended for filtering out the `GML` information from the incoming XML-document if the `REQUEST`-parameter `TRANSACTION` is set. Then the `HashMap` `transinsert` is filled with this information. Therewith, the name of the object to be stored is used as the key, and the `GML` information of this object is represented by the value. Next follows the implementation of the query for the

`REQUEST`-parameter `TRANSACTION` in the method `getWFSResponse()`. Should this parameter be set, the method `getTransactionResponse()` will be called in the following step. Here, the information stored in the `HashMap` `transinsert` will be committed to the `OGC`-compliant `GML-Importer`, which will be developed in our future work. Thus, the passed object can then be saved in `DB4GeO`.

3.2 Web Processing Service

The implementation of the `WPS` is based on the `OGC` specification called `WPS` Version 1.0.0. In addition to the methods “`GetCapabilities`”, “`DescribeProcess`” and “`Execute`”, the access to the `DB4GeO` database services “`3D-to-2D`”, “`ProgressiveMeshes`”, “`Intersect`”, and “`Contains`” have been implemented. The service infrastructure has been designed in a way that the future integration of other database services is possible without problems. The `UML`-diagram in Figure 3 shows the structure of the service infrastructure of the `WPS`.

The `UML`-diagram of Figure 4 shows the binding to the `RESTlet` module. The class `WPSObjectResource` inherits the functionality of `BaseResource`. `BaseResource` inherits the functionality from `ServerResource` of the package `org.restlet.resource`, whereby the first connection to `RESTlet` is started. Also a connection exists via the request of the class `OGCServices`. Methods that take over the processing of `http`-requests (for `GET`- and `POST`-requests) can be found in the main class `WPSObjectResource`. Additionally, a method that performs the validation of the key-value pairs and methods that take over the actual functionality of the `WPS` operations (in the form of a representation after `RESTlet`: `org.restlet.representation.Representation`) are able to take over processing requests..

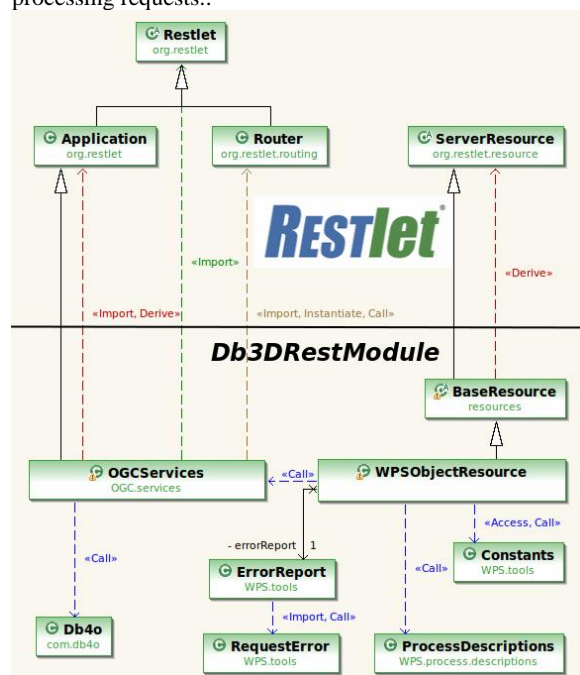


Figure 4: `UML`-Diagram of `DB4Geo`'s `WPS`

The currently available database services are divided into simple- and complex-services. `DB4GeO` starts its own instance of `WPS` (or `WFS`) for each project. Therefore, the project name is extracted from the `URI` and does not need to be specified. The project name is important for sourcing the requested objects from the database. “`Intersect`” and “`Contains`” belong to the simple database services: “`Intersect`” checks whether a

specific geometry intersects a given 3-D object and “Contains” checks whether a specific geometry in a specific 3D object is contained. The complex database services are represented by the “3D-to-2D” service, which generates a sectional view of certain 3D objects to a requested plane - and by “ProgressiveMeshes” which reduces the number of used triangles in a triangelnet. The output of the WFS is returned in the basic services in the form of a boolean value. With the complex services, either a GML-, GOCAD- or WebGL-file can be returned, depending on the user's intentions.

4. APPLICATIONS FOR OGC WEB SERVICES

4.1 Android AR Viewer

Geoscientific researchers frequently need to investigate and analyze their existing spatial data during their work in the field. The standard representations of these data sets have mostly been paper maps or, more recently, Geo-Information Systems on laptops. With maps only providing data in a static way and laptops often being impractical for field work, the high importance of a flexible and convenient solution becomes apparent. As the development of smartphones and tablet computers with high processing power progresses, new possibilities of in situ data-visualization arise. By using the intuitive touch-sensitive interface of the mobile devices for example, the user is able to explore different three-dimensional datasets interactively. Furthermore, the location sensors can be employed to realize an augmented reality system that superimposes the real world view with digital data to enhance the perception of the investigated area.

To demonstrate the potential of such a system in cooperation with the DB4GeO Web Feature Service capabilities as a data resource, a prototypical app for mobile devices has been developed. It was first built against version 4.0 of the Android OS (Ice Cream Sandwich) and currently uses OpenGL ES 2.0 as real-time rendering engine. Apart from the widespread use of Android devices, the mainly open-source concept of the platform was the reason to use the Android framework for the prototype development.

The app provides the following capabilities which will be explained below:

- 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 and rotating the dataset in 3D space)
- overlaying the camera view of the device with a transparent representation of the data (i.e. augmented reality)

One of the main issues of a mobile system in geosciences is the data connection. Therefore, the app provides two different types of resources for 3D datasets. On the one hand, the user can load 3D data from the device's internal storage as a convenient solution for areas with a weak or no wireless network connection.

But this method is not always appropriate for the visualization of several big datasets since the memory capacities of mobile devices are still limited. Therefore, the app implements a second way of loading data into the viewer. The built-in client allows for querying of an OGC Web Feature Service compliant to versions 1.0.0 and 1.1.0. Thus, the often large datasets are only loaded on demand and are not stored persistently on the device. Hence, the application does not require the receipt of information on the internal storage structure of the database

server but only needs to implement the suitable interface. The interaction between client and server is based completely on standardized concepts like the HTTP-protocol, XML (Extensible Markup Language) and XML-schemata/schemas. Any number of services with their corresponding connection information (ID, name, base URL) can be stored in a SQLite database table. The user can add and delete entries in the table and of course query for available services (see Figure 5).

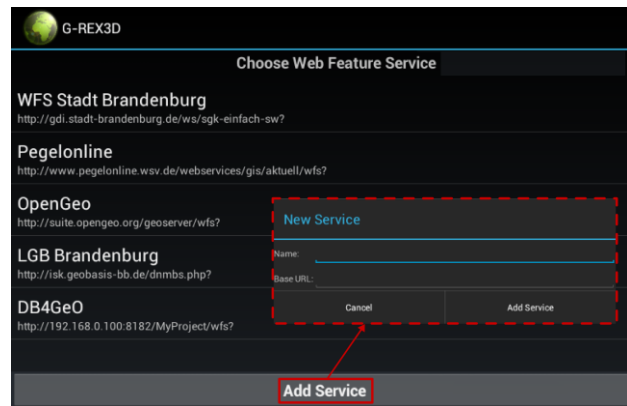


Figure 5: Adding a new service to the application

The communication process is a well-structured sequence of requests and responses between the client application and the Web Feature Service. In detail, the sequence can be divided into three major steps (see Figure 6).



Figure 6: WFS communication steps

First of all, the application translates the user input, given in the form of a selected list item (via touchscreen), to an HTTP GET request and transmits it to the service. In terms of OGC services this request is called *GetCapabilities* and shall provide general information on the server, its capabilities, and most importantly the available datasets.

The service creates a database query fitting its underlying database server, which, in turn, responds with a list of its stored datasets and corresponding metadata. The WFS formats this information as an XML-File (Capabilities-Document) and sends it back to the client, where it will again be presented as a list for the user. The list displays the name and the EPSG-Code of the dataset's geometry.

The EPSG-Code, as a structured collection of geodetic reference systems, is necessary for the correct transformation of coordinates and a possible geodetic datum shift. Subsequently, the user can select a dataset and the client application will produce two more HTTP requests in a background thread. The first one (*DescribeFeatureType*) will be answered with an XML-schema explaining the structure of the requested dataset. The second (*GetFeature*) will enable the WFS to transmit the

dataset as *StringRepresentation* in GOCAD format. The client imports the given structure into an OpenGL ES 2.0 fitting format and displays the result on the screen. All the communication between the application and the server is not visible for the user and is processed via asynchronous tasks in background threads.

For a high degree of user interaction, which is necessary to perceive three-dimensional data, the app provides possibilities to zoom in and out, to pan and to rotate the dataset in 3D space in real-time. To facilitate the usability, these actions are implemented as the typical gestures known from other touchscreen applications (swipe, pinch, etc.). Therein, the user inputs are translated according to the current zoom level of the dataset. Thus, the amount of translation, rotation or scaling is always suitable for the extent of the user's touch gesture.

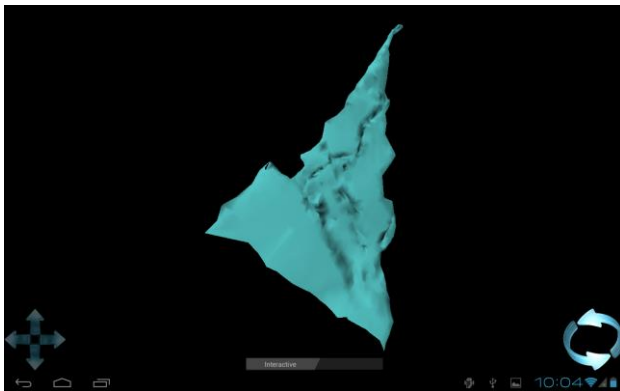


Figure 7: First view of digital dataset

As our geological test dataset only contains 3,000 triangles, there have been no critical technical problems visualizing subsurface structures in real-time. However, for larger data sets we will have to support the real-time visualization by techniques such as progressive meshes to visualize different levels of detail for complex subsurface structures.

Using the slider at the bottom of the application view (see Figure 7), the user can switch into the “Augmented Reality” mode. Augmented Reality can be interpreted as a type of information fusion providing an exceptional high level of perception. It combines digital and real world content by overlaying the camera view of the device with mainly transparent representations of digital models.

Apart from the camera sensor and accurate georeferencing of the digital dataset, the mobile device has to feature internal or external sensors to capture its position and orientation. The sensors that are used in this application are the GNSS sensor, a gyroscope and an accelerometer. The real-time data of these three MEMS-sensors are fused to estimate the current movements of the user, modifications to his position and field of view. Therein, the GNSS-sensor is responsible for the 3D geographical coordinate, the gyroscope for the high-rate and accurate orientation, and the accelerometer data is used correct the cumulative shift of the gyroscope and to improve the azimuth measurement.

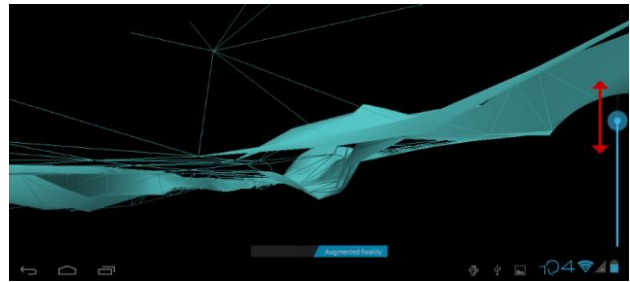


Figure 8: Vertical slider to zoom "down"

As an additional feature, the application enables the user to zoom “down” on his current position and explore the underground data surrounding him. This vertical adaption of the application's view is implemented as a shift bar on the right side when the user is in the Augmented Reality mode (Figure 8). This function could be especially helpful for a geologist to examine datasets of geological layers and improve their perception on site.

In Figure 9, three screenshots of one of the very successful example sessions are displayed. The digital dataset is a strongly simplified model of the front of Karlsruhe Palace. At the moment the drawbacks of the system can be traced back to the sometimes poor accuracy of the GNSS coordinates which resides averaging inconsistency between 10 to 20 meters. Thus, the model is sometimes shifted away from the camera view producing a deficient overlay. Another issue is the limited processing power of mobile devices which results in low frame rates (< 15 fps) concerning big datasets.

Both of these described drawbacks, however, will decrease with the ongoing development of mobile devices and the implementation of high class GNSS sensors. Most notably, the latest implemented location sensors are able to use more correction data provided via mobile network and hence can improve the positional accuracy to 3 meters or even better.



Figure 9: Example session at Karlsruhe Palace

4.2 Working with qGIS

An additional application example which demonstrates the benefits of interoperability gained by the use of OGC standards is the coupling of DB4Geo with the popular, classical Geo-Information System software qGIS¹. The software is a popular free open-source GIS-Software used to display and model classical 2D raster data. It is also capable of working with vector-based spatial data like points, lines and polygons from various sources.

In our application scenario we are only interested in the capabilities of dealing with vector-based data. One of the main issues of a Geo-Information System is the integration of spatial data sources. It is already possible to connect qGIS with popular spatial databases like PostGIS or Spatialite directly. Additionally there is also an option to establish connections to standard-based servers including the WebFeatureService.

This application scenario consists of the coupling of the WFS of DB4Geo and the WFS-interface of qGIS.

Despite the fact that QGIS is a Geo-information System, which is only capable of visualizing 2D spatial data (in the standard version), it is still possible to capture real 3D data from the new WFS-interface of DB4Geo and visualize it in 2D (see Figure 10).

¹ <http://www.qgis.org/>

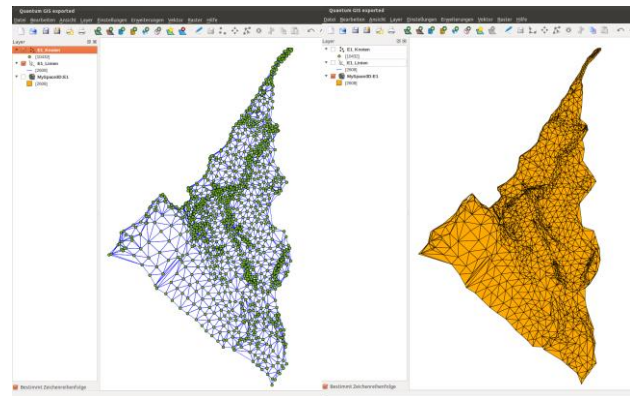


Figure 10: Visualization of a 3D object in qGIS

According to the rules of the WFS standard (OGC 2005), first a GetCapabilities request is sent in the form of an XML document to the WFS server of a specified DB4Geo project.

The response document contains information about the existing 3D spaces and their 3D objects (in this case the spaces MySpace3D and AnySpace3D and the objects E1 and A2). In the next step qGIS automatically processes and displays the information and it is possible to add one or more of these objects as a new layer (Figure 11).

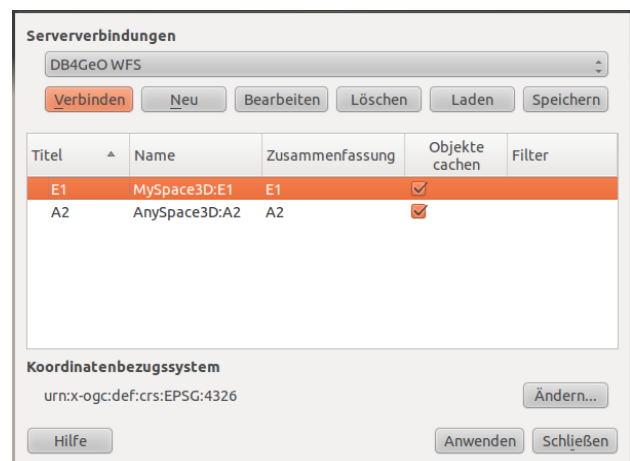


Figure 11: Selecting an object out of DB4Geo via WFS

After selecting a specific object, qGIS queries the 3D spatial data with the help of a DescribeFeatureType and a GetFeature request from the DB4Geo WFS server. Internally the result of such a query is returned in a GML format (Figure 12) but is displayed as a new layer in qGIS. Internally, the Z values of the 3D coordinates are ignored, so that the 3D geo-object can be represented in a 2D workspace.



Figure 12: Result of a DB4Geo WFS query

Due to this application scenario, it was demonstrated that the WebFeatureService of DB4Geo can work together with standards-based GIS clients like qGIS. Therefore by the consistent implementation of OGC standards in DB4Geo the

interoperability of this spatial database has been increased enormously.

5. FUTURE APPLICATIONS IN THE GULF REGION

3D data modelling, management, and analysis are likely to become key issues in the Gulf region over the next few years. Research will focus on ways to support 3D city and infrastructure applications as well as 3D sub-surface modelling e.g. for oil exploration. Especially in the United Arab Emirates (UAE), first 3D visualization examples show the demand for a continuous 3D workflow concerning data collection, management, visualization, and analysis.

Obviously, an increasing number of commercial enterprises are using geospatial information as an essential element in their business strategies. Geospatial information provided to the government, industry and citizens, will serve as a central instrument to support the ambitious city and infrastructure development projects in the UAE. Thus data utility in the region will critically depend on intelligent and scalable software systems. That is why the time has come for a new culture of collaboration between Computer, Geoinformation, Earth, and Civil Engineering Scientists to connect their different approaches to 3D modelling.

Obviously, combined 3D models for cities, infrastructure, and geological subsurface structures that are being processed by different geo-software coupled with a 3D geo-database architecture such as DB4Geo, could significantly improve the continuous long-term management and analysis of 3D knowledge. These combined 3D models will create new ways to understand and analyse urban and landscape planning. Furthermore, there is a strong need for new geo-services providing meta-data and 3D data for selected regions.

6. CONCLUSION

The service-oriented database architecture DB4Geo has been presented. This paper focused on the open access to DB4Geo via standardized web-services. Implementations of REST- and OGC-based approaches demonstrated new ways to access DB4Geo's 3D data stores and operations. The OGC Web Feature and Web Processing Service were used to access DB4Geo functionality. Complex services such as the 3D-to-2D service and mesh-simplification (Progressive Meshes) through a Web Processing Service have been illustrated. The Android-based mobile 3D augmented reality viewer for DB4Geo revealed how the Web Feature Service could be used to visualize 3D geo-database query results. Finally, an outlook is given on future research work considering future applications of DB4Geo in the Gulf Region.

7. REFERENCES

Balovnev, O., Bode, T., Breunig, M., Cremers, A.B., Müller, W., Pogodaev, G., Shumilov, S., Siebeck J., Siehl, A., Thomsen, A., 2004. The Story of the GeoToolKit - An Object-Oriented Geodatabase Kernel System. *Geoinformatica* 8:1, Kluwer Academic Publishers, 5-47.

Breunig, M., Butwilowski, E., Golovko, D., Kuper, P.V., Menninghaus, M. Thomsen, A., 2012. Advancing DB4Geo. *Proceedings 3D Geoinfo 2012*, Quebec, Canada, Lecture Notes in Geoinformation and Cartography, Springer, 193-210.

Breunig, M., Butwilowski, E., Kuper, P.V., Golovko, D., Thomsen, A., 2012. Topological and Geometric Data Handling

for Time-Dependent Geo-Objects Realized in DB4Geo. *Proceedings Spatial Data Handling (SDH) 2012*, Bonn, Germany. 16p.

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, 83-102.

Bär, W., 2007. *Management of geoscientific 3D data in mobile database systems*. In German. Ph.D. Thesis, University of Osnabrück, Germany, 166p.

Fielding, R.Th., 2000. *Architectural styles and the design of network-based software architectures*, Ph.D. thesis, University of California, Irvine, USA.

Koubarakis, M., Sellis, T., Frank, A.U., Grumbach, S., Güting, R.H., Jensen, C.S., Lorentzos, N., Manolopoulos, Y., Nardelli, E., Pernici, B., Schek, H.-J., Scholl, M., Theodoulidis, B., Tryfona, N. (Eds.), 2003. *Spatio-temporal databases. The CHOROCHRONOS Approach*. Lecture Notes in Computer Science, Vol. 2520, 358p.

Le, H.H., Gabriel, P., Gietzel, J., and Schaeben, H., 2013. An Object-Relational Spatio-Temporal Geoscience Data Model. *Computers & Geosciences*, accepted.

Mallet, J.L., 1992. GOCAD: a Computer Aided Design Programme for Geological Applications. Turner A.K. (Ed.), *Three-Dimensional Modelling with Geoscientific Information Systems*. Proceedings of NATO ASI 354, Kluwer Academic Publishers, Dordrecht, 123-142.

Mallet, J.L., 2002. *Geomodelling*. Oxford University Press, 599 p.

Polthier, K., Rumpf, M., 1994. *A concept for Time-Dependent Processes*. In: Goebel et al., (Eds), *Visualization in Scientific Computing*, Springer, Vienna, 137-153.

Pouliot, Desgagné, Badard, Bédard, Thomas, 2007. Development of a Web Geological Feature Server (WGFS) for sharing and querying of 3D objects. *2nd International Workshop on 3D Geo-Information*, December 14-16, Delft, Netherlands.

OGC ® Inc. (ed.), 1994. Open Geospatial Consortium, Spatiotemporal Geometry and Topology, <http://ormdev.opengeospatial.org/node/95>.

OGC ® Inc. (ed.), 2005. Web Feature Service Implementation Specification. 1.1.0.,

<http://www.opengeospatial.org/standards/wfs>.

Pouliot, Roy, Fouquet-Asselin, Desgroseilliers 2010. 3D Cadastre in the province of Quebec: A first experiment for the construction of a volumetric representation. *In: Advances in 3D Geo-Information Sciences*. Lecture Notes in Geoinformation and Cartography, Volume Editor(s): Kolbe, Thomas H.; König, Gerhard; Nagel, Claus. 3DGeoInfo conference, Berlin, Nov. 3-4.

Schaeben, H., Apel, M., v.d. Boogart, G., Kroner, U., 2003. GIS 2D, 3D, 4D, nD. *Informatik Spektrum*, Vol. 26, No. 3, June 2003, 173-179.

Siehl, A., Rüber, O., Valdivia, M., and Klaff J.: Geological Maps Derived from Interactive Spatial Modeling, 1998. In: R. Vinken. *Geolog. Jb., From Digital Map Series in Geosciences to Geo-Information Systems*, vol. 11, 273-289.

Thomsen, A., Jentzsch, T., and Siehl, A., 1998. Towards a balanced kinematic model of a faulted domain in the lower rhine graben. In Proc. of the *GOCAD ENSG Conference: 3D Modeling of Natural Objects - A Challenge for the 2000's* (Nancy, France, June 20-25 2000), p. 12.

van Oosterom, P., Stoter, J.E., 2010. 5D Data Modelling: Full Integration of 2D/3D Space, Time and Scale Dimensions. *GIScience 2010*, 310-324.

Worboys, M., 1994. A Unified Model for Spatial and Temporal Information. *The Computer Journal*, 37(1):25–34.