

INTERNET BASED DISTRIBUTION AND VISUALIZATION OF A 3D MODEL OF THE UNIVERSITY OF COLOGNE CAMPUS

Willmes, C.*, Baaser, U., Volland, K. and G. Bareth

Institute of Geography (GIS & RS), University of Cologne
Albertus-Magnus-Platz, 50923 Cologne, Germany

*corresponding author; Phone: +49-(0)221-470-6234, c.willmes@uni-koeln.de

Abstract

Visualization of 3D geodata integrated into a web browser based user interface is still a key issue in geoinformation science research. In a short overview of state of the art solutions for this demand, the key advantages and disadvantages of each of those solutions are lined out. As a solution which combines most of the advantages identified before, a CityGML based 3D model of the University of Cologne Campus and the setup of an OGC conform Web Perspective View Service for internet based visualization of this model is described.

Keywords

3D geodata visualization, 3D model, 3D buildings, CityGML, 3D-SDI, WPVS.

0. INTRODUCTION

In this paper an implementation of a CityGML (Kolbe et. al 2008) based 3D model of the University of Cologne Campus buildings and the setup of an OGC Web Perspective View Service (WPVS) based visualization of this model is presented. The considerations to choose the WPVS technique for the system setup and the implementation of this web service will be described in the following of this paper.

Study area for this project is the University of Cologne Campus area for which a GIS is available, the CampusGIS (CampusGIS 2010). The CampusGIS provides general and spatial campus information, e. g. visualization, advanced search functions, orientation, routing and facility management. The overall design approach is based on the linkage of several existing relational databases with spatial data in a WebGIS environment (Baaser et. al. 2008). The University of Cologne, founded in 1388, is one of the largest and oldest universities in Germany. The most buildings of the university are located in a 2km² large area (the campus) in the western centre of the city of Cologne.

1. STATE OF THE ART

The different available approaches to web based 3D geodata visualization have advantages and disadvantages. For a good user interface it is crucial to offer easy, fast and seamlessly integrated access to the visualization of the three dimensional geodata within the web browser user interface (Willmes 2009). This means, that the user should not have to obtain any additional software to what normally is present on an internet connected desktop computer, which is basically a state of the art web browser.

The process of visualizing geodata in distributed environments can be generalized through the model of the OGC portrayal pipeline (Cuthbert 1998). The portrayal pipeline divides the process of geodata visualization into four independent steps, which must not be implemented on the same system. These four steps are: i.) *Filter/Select*, ii.) *Display Element Generator*, iii.) *Render* and iv.) *Display*. Figure 1 shows, that for the three dimensional case of geodata visualization, some specific constraints to the portrayal pipeline are to be considered (Altmaier & Kolbe 2003). In step i.) the data which will be visualized is selected by database, web service or file backend queries. In step ii.) a model of the data, which will be visualized, is generated with information of how the data will be styled/visualized by defined rules. For example the color of the non-textured parts of buildings, will be applied in this step. The format of these compiled models can be for example in CityGML, KML, VRML or X3D. In step iii.) those 3D models are rendered from the geometry and styling informations given in this scene graph models. In step iv.) the rendered view of the data is displayed on the client system (Willmes 2009).

This partitioning of the visualization process allows the setup of distributed systems (Cuthbert 1998, Altmaier & Kolbe 2003). In the left illustration of figure 1, the different orchestration setups of the portrayal pipeline are shown. It shows, that it is possible to orchestrate data and display elements (scene graphs) from different sources, in 3D case it is not

possible to orchestrate the rendered views at the client because of line of sight constraints (Altmaier & Kolbe 2003), in opposite to the 2D case where this is possible (e.g. semi transparent WMS overlays). The right side of figure 1 shows four different client/server system configurations for service oriented architecture (SOA) based 3D geodata visualization, i.) *local system*, ii.) *thick client*, iii.) *medium client*, iv.) *thin client*. These configurations are different in the amount of portrayal pipeline steps processed client side.

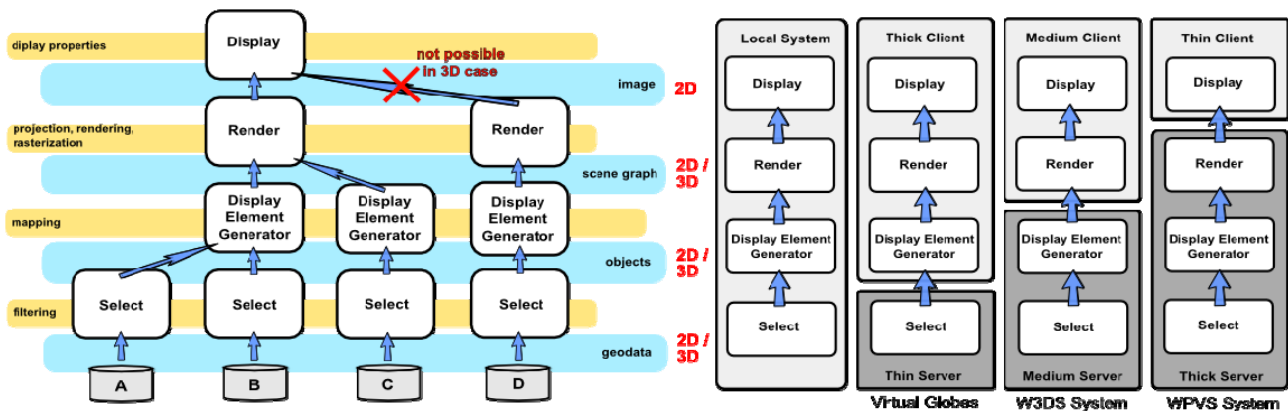


Fig. 1: OGC portrayal pipeline and client/server system setups for SOA based geovisualization (Willmes 2009 modified from Kolbe 2004).

An example for the *local system* would be desktop software systems like *ArcScene* or *LandXplorer*, where all four steps of the portrayal pipeline are processed locally. Examples for the *thick client* architecture would be virtual globes like *Google Earth* or *NASA WorldWind*, where the data selection, styling, rendering and display are done on the client, but the raw geodata is accessed over the network. An Example for the *medium client* approach is the OGC Web 3D Service (W3DS) technology (Quadt & Kolbe 2005), where the application renders server generated 3D scenes client side. A known implementation of this approach would be the GDI-3D technology (Zipf et. al. 2007, GDI-3D 2010). An example for the *thin client* architecture would be the presented system setup of a WPVS, in this case implemented with the degree3 framework (degree 2010).

The key advantages of the virtual globe thick client architecture implementations are the very good interactivity, very fast and good rendering and at least in case of *Google Earth* the massive available geodata backends. The disadvantages of this approach are the fact, that these systems are stand alone software programs, which cannot be integrated seamlessly into a web page and have to be obtained and installed on the client system. The key advantage of medium client systems is relatively lightweight client design, if for example implemented in Java, those clients are integrate able into websites as Java Applets or as Java webstart, which lowers the hurdle of installing extra software by the client to access the application. A disadvantage is that this Plug-In approach is critical for platform compatibility reasons and also requires the client user to install additional software, which may not always possible for different reasons.

The presented WPVS setup is an implementation of the thin client architecture approach, with only the display of the rendered view of the data on the client. The first three steps of the portrayal pipeline are processed on a single server, it would be possible to distribute the data backend of the WPVS on several servers to distribute server load for example. In this setup the user accesses the visualizations through a web browser based user interface, which generates the WPVS GetView request key value pairs (KVP) by several user input elements of the GUI (see 3.3 for a detailed description of the GUI). The server generates the rendered views of the data stateless from the parameters coded in the request string (Singh 2001, Hagedorn 2010).

The key advantages of this WPVS based setup for the visualization of three dimensional geodata are: i.) the visualization is accessible integrated into a website from a web browser based GUI, ii.) no third party Plug-Ins must be installed, iii.) the application is platform independent accessible from every system with a standard web browser, and iv.) this includes mobile devices with internet access.

None the less, there are also disadvantages to the WPVS technology approach. The main disadvantage of the WPVS system setup is the rather limited interactivity, because response times from the server to the client are just too long for "real" interactivity. On every change of the view, a new request is send to the server, the server accesses now all the data sources and renders the view, which then has to be sent back over the network to the client. On a setup with good network connectivity between client and server and fast server response times, the delay is about 2 seconds per frame. This is too high for a really interactive application, generally one speaks of instant feedback at a frame rate from above 10 frames per second (Döllner 2005).

2. SYSTEM SETUP

The CampusGIS-3D (CampusGIS-3D 2010) web visualization system contains a 3D model, in up to four Levels of Detail (LoD0 to LoD3), of the buildings of the University of Cologne Campus, a digital elevation model and a high resolution aerial image of the campus area. The system is built with a combination of free for use and open source software tools and own implementations of software for CityGML data conversion tasks and the WebGUI. The presented system setup follows the 3D-SDI (Bezema et. al. 2007, Zipf et. al. 2007) system architecture. The 3D-SDI technology is an architecture pattern to design SOA based, and OGC standard conform, spatial data infrastructures for three dimensional geodata (Zipf et. al. 2007). The deegree3 framework implements all building blocks necessary to implement OGC service based and standard compliant 3D-SDI applications (deegree 2010).

2.1 Software

The system setup of the CampusGIS-3D WPVS is based on open standards and open source software. Server side runs an open source Linux system with the well known web server software from the Apache Software Foundation. We use the Apache Http (Apache Foundation 2010c) and Apache Tomcat (Apache Foundation 2010a) server as a container for the deegree3 WPVS system and the WebGUI. For modeling the campus buildings, *Google's building maker* (Google Inc. 2010a) and *Google Sketchup* (Google Inc. 2010b) including *citygml.de Sketchup Plug-In* (CityGML-Toolchain 2010) were used. To convert the geodata for the use with deegree3 WPVS, we used the tools provided by the deegree3 framework for this purpose. Additionally we developed the WebGUI to navigate in the 3D model through the server side rendered images, and some tools for data conversion steps which were not implemented in the mentioned software.

2.2 Data

There are three sources of geodata combined in the given system setup, i.) an elevation model of the Cologne Campus area, ii.) an high resolution aerial image for texturing the digital elevation model (DEM) and iii.) the 3D building models of the University of Cologne Campus. The raster data of the aerial image and the DEM cover an area of 4000m x 4250m in the bounding box of 2562750E, 5642000N for the south east corner and 2567000E, 5646000N for the north west corner in the geodetic coordinate system Gauß-Krüger (zone 2).

Digital Elevation Model

In this first development phase of the CampusGIS-3D model we used freely available digital elevation models (DEM) of the campus area. In the final setup we used the SRTM90 (CIGAR-CSI 2010) data. We also tested with the freely available ASTER GDEM (ERSDAC 2010), which was not less good than the SRTM model, but we had to take only one. Given the fact, that the campus area of the University of Cologne is relatively flat, with an altitude difference of at max. 10 meters on the whole area, beside a little hill near the Aachener Wheier, which is nearly 15 meters higher than the rest of the campus area, the "quality" of the DEM is not that important for visualization purposes in this project. In the before mentioned CampusGIS, there is a 5 m DEM from the Land Survey Office of North Rhine-Westphalia available (Hennig 2008), which can be integrated into the CampusGIS-3D model in the future.

Aerial Image

The aerial image, used as the DEM texture in this project, is provided from the geodatabase of the before mentioned CampusGIS (Baaser et. al. 2008). This image is an orthorectified airborne photography, in a scale of 1:5000 (DOP5), provided by the land survey authority of North Rhine-Westphalia (LVermA) in 2003. The image has a spatial resolution of 0,30 m per pixel and an average accuracy of 1 m (Hennig 2008).

Buildings

In the CityGML standard specification five Levels of Detail (LoD) are defined (see Figure 2 for examples). LoD1 objects should have an accuracy better than 5m and do not have to implement roof type. LoD1 buildings are also referred to as *block buildings*. LoD2 objects should have a geometry accuracy of better than 2 m and the roof type and its orientation should be included in the model. The LoD3 models have an overall geometry accuracy of better than 0.5 m and the real object form of the roof, windows, doors, balconies, etc. of the buildings should be represented in the model (Kolbe et. al. 2008).



Fig. 2: The five Levels of Detail (LoD) defined by CityGML (Kolbe et. al., 2008).

The buildings of the CampusGIS-3D model are CityGML LoD1, LoD2 and partly LoD3. The LoD1 model of the campus buildings (see Figure 3) was originally created in a diploma thesis project (Hennig 2008) and converted into CityGML in another diploma thesis project (Willmes 2009) at the GIS & RS working group of the Institute of Geography of the University of Cologne.



Fig. 3: LoD1 building data set of the University of Cologne Campus (Hennig 2008).

The LoD2 buildings of the CampusGIS-3D model are created with the web based *Google Building Maker* (Google Inc. 2010a) tool. The textures of these buildings are also completely derived from this tool. See figure 4 for some examples of the LoD2 buildings of the Model.



Fig. 4: Two example LoD2 buildings of the CampusGIS-3D model, created with Google building maker tool and visualized in Google Sketchup.

The LoD3 buildings are modeled by extending the LoD2 buildings with help of ground and floor plans of some buildings, individual on-site investigation and taken photographs of the objects. The modeling process for the LoD3 buildings is done with the tools Sketchup (Google Inc. 2010b) and Blender (Blender 2010). See Figure 5 for an example of an LoD3 building. The modeling process of extending the LoD2 models to LoD3 models is very time consuming, but it is planned to have the complete Campus in LoD3 at some point.

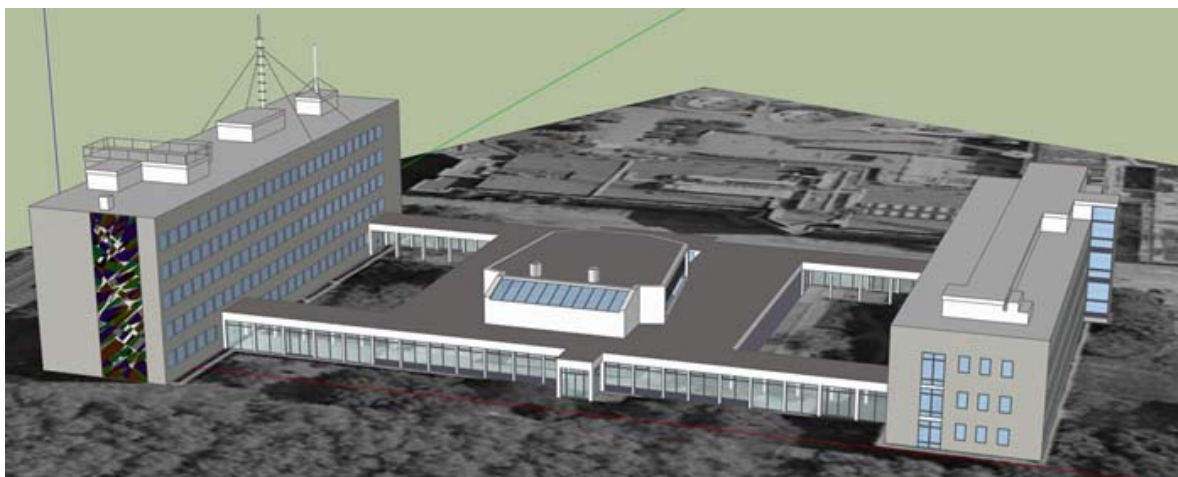


Fig. 5: LoD3 model of the geo science building visualized in Google Sketchup.

3. IMPLEMENTATION

For the implementation of the WPVS setup, the *deegree3 framework* (deegree 2010) was chosen. Deegree3 is a Java based open source framework for building spatial data infrastructures. As far as the authors know, it is the only open source implementation of an OGC WTS/WPVS (Singh 2001, Hagedorn et. al. 2008) available. Because the WTS/WPVS is still in development stage at OGC, there is no official standard yet (Bezema 2007).

3.1 3D-buildings to deegree3D conversion toolchain

For the conversion of the LoD1 building models (Hennig 2008), which were given in Shapefile 3D format, the software *LandXplorer Studio Professional* was used. The *LandXplorer* software is able to import the Shapefile 3D models and export a CityGML document. This CityGML document is not CityGML 1.0.0 schema valid. A Java tool *CityGMLConverter* (CityGMLConverter 2010) was implemented, to convert those non schema valid CityGML documents into CityGML 1.0.0 valid documents, to be able to import those models into the deegree3 WPVS backend. See Figure 6 for a schematic view of the conversion toolchain.

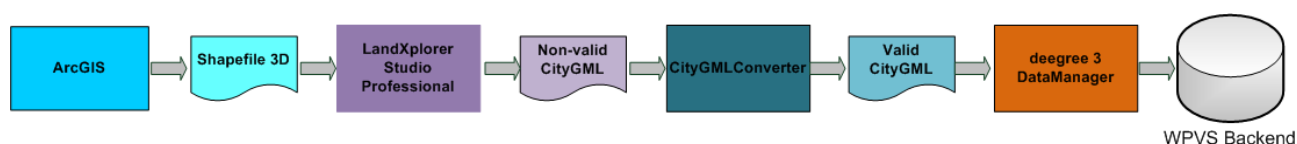


Fig. 6: ArcGIS to deegree3 3D building model conversion toolchain.

For the LoD2 models we used the *Sketchup CityGML Plug-In* (CityGML-Toolchain 2010) to export the Sketchup format models, created with the *Google building maker* (Google Inc. 2010a) tool, into the CityGML format. The CityGML documents, which are exported from the *Sketchup CityGML Plug-In* are also not CityGML 1.0.0 schema valid. The deegree3 DataManager tool demands schema validity to import given CityGML documents. In Figure 7 a schematic view of the conversion toolchain necessary to produce CityGML models ready for import into the deegree3 WPVS backend is shown.

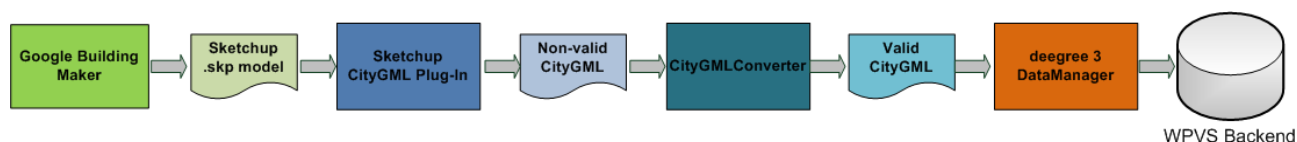


Fig. 7: Google Building Maker to deegree3 3D model conversion Toolchain.

The before mentioned *CityGMLConverter* tool converts the non-schema-valid CityGML models, exported from the *Sketchup CityGML Plug-In*, into CityGML 1.0.0 schema-valid models, by parsing the Sketchup exported CityGML document and adding missing CityGML elements and attributes, necessary for CityGML 1.0.0 schema validity, to the model.

The Sketchup based models are stored in a local metric coordinate system with a single world coordinate origin for each model. The coordinates of the local model origin are accessible through the *Sketchup* GUI in WGS84 decimal degree geographical coordinates (EPSG code 4326) for each model. With the world coordinates of the local model origins, the *CityGMLConverter* tool is able to translate the local metric coordinates of the models into a metric world coordinate system. For this CampusGIS-3D project the geodetic German Gauß-Krüger (Zone 2) coordinate system (EPSG code 31466) is used. The *CityGMLConverter* tool does first a coordinate conversion from the given EPSG 4326 geographical coordinates into the metric Gauß-Krüger EPSG 31466 system. For this conversion the coordinate conversion classes of the deegree framework are facilitated. Additionally a height value for the given coordinate origin is obtained during conversion over the web via *Google Elevation Web Service* (Google Inc. 2010c), because the Sketchup GUI does not provide the height value of the model origins. After the conversion the *CityGMLConverter* adds the metric local coordinates of the models to the Gauß-Krüger transformed coordinates of the *Sketchup* model world coordinate origins, to obtain all geometries of the model in world coordinates, which is demanded by the deegree WPVS.

3.2 Setup of the deegree3 WPVS

The deegree3 framework offers a set of Java tools to convert the geodata you want to serve with the WPVS into several suitable file or database based backends for the deegree3 WPVS. The deegree3 framework is not yet officially released, so there are no binary builds for download, which means, that one have to build the software from source.

The Elevation model of the campus area was originally provided in GeoTiff format, to import the data with the DEMDatasetGenerator tool of the deegree3 framework, the data has to be converted into the ASCII XYZ format. This is done with the tool *gdal_translate* of the well known open source Geospatial Data Abstraction Library (GDAL 2010).

The deegree3 WPVS uses a view dependent multi resolution model for triangular meshes for the internal representation of DEMs. This deegree3 DEM approach is a Batched-MT / BDAM variant and by this suitable for very large and detailed DEMs (Schneider 2008). The tool DEMDatasetGenerator of the deegree3 framework compiles XYZ ASCII DEMs into this view angle dependent multi resolution format. See figure 8 for a visualization of the data conversion toolchain to import a DEM into the deegree3 WPVS backend.



Fig. 8: DEM to deegree3 data conversion toolchain.

To import 2D raster data layers, for texturing the surface of the DEM, it is necessary to generate a deegree3 internal grid based file backend. This grid is a simple uncompressed data structure containing a grid of one or more raster files. This data structure is generated by first converting the input raster data into a RasterTree structure of resolution levels from the input raster. The deegree framework offers a tool *RasterTreeBuilder* for this task. In a next step, the before generated RasterTree is converted into the mentioned grid based data structure ready for use in the deegree3 WPVS. This conversion step is done with the deegree3 *RasterTreeGridifier* tool (see figure 9).

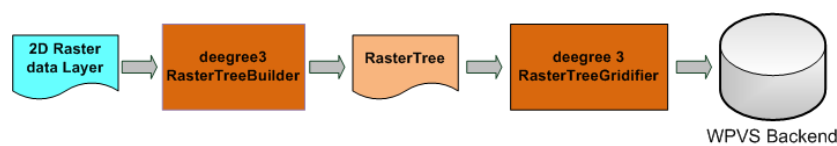


Fig. 9: 2D Raster data layer to deegree3 conversion toolchain.

The deegree3 valid CityGML building models are imported for the use with deegree3 WPVS into a file based backend. This backend has a deegree internal binary format suitable for fast access to the building models by the deegree3 framework. It is also possible to configure a data base backend, for example PostgreSQL/PostGIS as data store for the CityGML models. The deegree3 framework offers a tool named *DataManager* to import CityGML models into the file or database based backend.

The different data sources, resolution levels of the data, coordinate bounding boxes and other options, as well as some metadata about the author and the data itself, have to be configured for access from the deegree3 WPVS. This is done by authoring a deegree WPVS configuration XML document. Finally a web application archive is build and deployed into a Java Web Container, in this setup into an Apache Tomcat (Apache Foundation 2010a) Java Servlet-Container.

3.3 Implementation of the WPVS WebGUI frontend

A WebGUI frontend for navigation in a 3D scene using the WPVS interface was developed. This is facilitated by generating the WPVS GetView request from interacting with the GUI elements. The WPVS interface offers five parameters to navigate in the given 3D scene. These parameters are i.) point of interest (POI) to define the point on which the view is centered, ii.) the PITCH to define the angle of inclination, iii.) YAW to define the azimuth of the view direction, iv.) DISTANCE to define the direct distance between the point of camera (POC) and the POI, and v.) angle of view (AOV) to define the breath of landscape in the viewer's scene. See figure 10 for a visualization of these parameters.

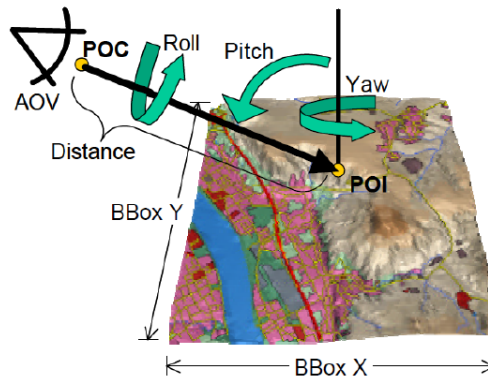


Fig.10: Definition of a perspective view of a 3D scene (Kolbe 2004).

To interact with these five parameters provided by the WPVS interface, three basic GUI elements are identified (see figure 11): i.) a slider element, named distance slider, for interacting with the DISTANCE parameter, ii.) a four direction cross, named view angle control, to interact with the YAW parameter in left/right axis and the PITCH parameter in up/down direction, and iii.) a second four direction cross, named POI control, to interact with the POI in north (up), east (right), south (down), west (left) direction.

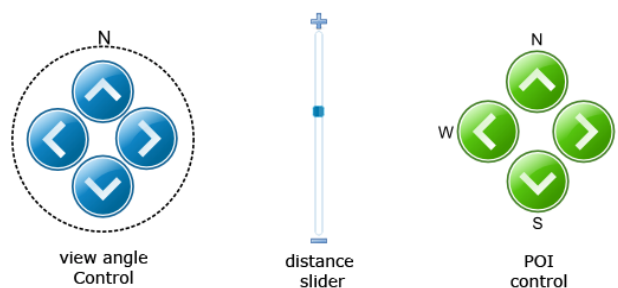


Fig. 11: Basic 3D scene navigation elements.

Optionally it is possible to implement further parameters (LAYERS, BBOX, STYLES, etc.) for interaction with the WPVS through the GUI. The GUI is a lightweight HTML/CSS/JavaScript based implementation. The processing of the WPVS parameters from the GUI interaction is implemented in JavaScript and completely processed client side. For UI element interaction handling and processing, the open source JavaScript Framework jQuery (jQuery 2010) is integrated into the GUI. The development of this WebGUI is in an early stage and it is planned to enhance this GUI in the future to interact with all WPVS provided parameters through the GUI and also some extra functionality like mouse wheel zoom, 3D panning and authoring of "fly through" animations by chaining a set of different views in a time step based loop over those by the GUI defined view frames.

Another planned feature, for orientation purposes, which is an integrated 2D overview map of the actually requested BoundingBox and the visualization of the view frustum of the current view generated from the WPVS by a 2D polygon overlay over a base map. This can be implemented by integrating a WMS client, for example OpenLayers (OpenLayers 2010), into the GUI to request WMS provided base maps for the requested BoundingBox simultaneously to a triggered WPVS request, and the 2D polygon of the view frustum seen from above, drawn as vector geometries client side over the 2D WMS base map.

4. RESULTS

A result of this work is the theoretical and practical grounding of reasons for the use of a WPVS setup, for internet based 3D geodata visualization. The main reason for the use of WPVS in this context is, that the client side 2D image based visualizations in well adapted formats like JPEG, PNG or Tiff, of this approach ensures best accessibility and interoperability in terms of client side software and hardware requirements. Further more a detailed documentation of the setup of an 3D-SDI (Zipf et. al. 2007, Bezema et. al. 2007) open standards based WPVS system to visualize 3D geodata web based, is given in this contribution.

The WebGUI developed for this project is another realization of the presented work. In Figure 12 a screenshot of the first development stage of the WPVS GUI is shown. The code is strictly separated by visualization and functionality. The visualization and placement of different UI elements and content containers is implemented in CSS. The functionality of processing the WPVS GetView request string is implemented in JavaScript. Both are integrated through the HTML markup of the GUI web site interface. This development approach allows relatively free templating and themeing/design of the GUI for different applications, which guaranties the reuse of this software for different WPVS based projects in the future.

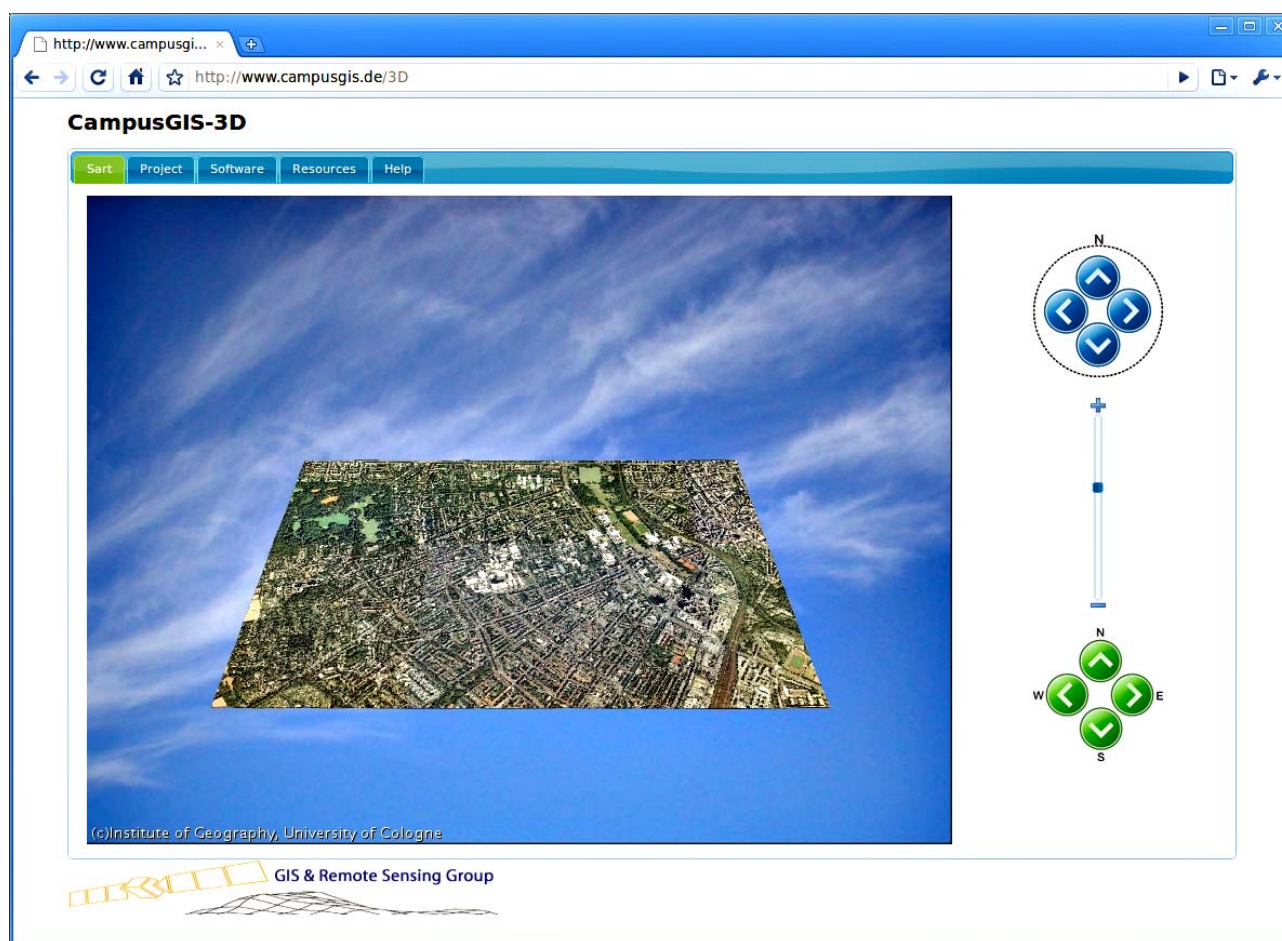


Fig. 12: Screenshot of the first development stage of the web browser based WPVS Client GUI.

Another result is the *CityGMLConverter* tool, which helps to modify and convert CityGML datasets exported from *Sketchup* or *LandXplorer*, for import into the backend of a deegree3 based WPVS system. The tool is implemented in Java and facilitates the coordinate transformation code of the deegree framework, as well as the *Apache Xerces* (Apache Software Foundation 2010b) libraries for XML processing and the *Google Maps Elevation Web Service* for obtaining height values for the world coordinate system origins of the *Sketchup* based building models. The *CityGMLConverter* tool will be published for download on the web sites of the CampusGIS-3D (CampusGIS-3D 2010) project in the future.

5. CONCLUSION AND FUTURE WORK

In this contribution it was shown, that it is possible to setup an open standards based application for internet based distribution and visualization of 3D geodata.

An adaption of the system shown in this paper, for the visualization of three dimensional models of archaeological excavation sites, is possible due to the development of a CityGML Application Domain Extension (ADE) for archeological excavation sites. A WPVS based visualization of these CityGML based excavation site models will be integrated into the spatial data infrastructure of the Collaborative Research Centre 806 (Bolten et. al. 2010, CRC806 Database 2010).

For the future the development of a medium client architecture based on *WorldWindJava* (WorldWind Java 2010), for the client application is planned. An OGC standard based infrastructure for delivering CityGML and KML documents of the excavation site models for rendering in the *WorldWindJava* based client, will also be designed for this system. To implement this medium client approach, *WorldWindJava* will be extended to directly render CityGML in its interface, functionality to render KML already exists for WorldWindJava. This technology will also be integrated into the CRC806 data base, as a more interactive interface to visualize the mentioned CityGML based models of archaeological excavation sites.

At the moment a further development of WTS/WPVS technology is announced as Web View Service (WVS) at the OGC as a discussion paper (Hagedorn 2010). The WVS is designed to overcome the restricted visualization and interaction capabilities of WTS/WPVS-based approaches. As a major extension, the WVS provides additional geometrical and thematic data, such as depth information and object identity information, which are encoded in the retrieved multi-layer images. Additionally, the WVS provides operations for retrieving information on visualized objects at specific image positions, measurement functionalities, and enhanced navigation support (Hagedorn 2010). This are promising new developments to the here presented thin client architecture technology and will be taken into account for the development of future projects on internet based 3D geodata visualization.

ACKNOWLEDGEMENTS

The authors would like to thank the deegree framework core developers Markus Schneider and Rutger Bezema for their support regarding issues with the deegree3 framework.

REFERENCES

- Altmaier, A. and Kolbe, T. H. (2003): Applications and Solutions for Interoperable 3D Geo-Visualization. In Fritsch, D. (ed.): Proceedings of the Photogrammetric Week, Wichmann, Stuttgart.
- Baaser, U., Hennig, S. D., Aasen, H., Dornauf, E., Gnyp, M. L., Hoffmeister, D., Köhn, N., Louwen, B., Laudien, R. and Bareth, G. (2008): AJAX-based linkage of databases for location-based-services: The Online-CampusGIS of the University of Cologne. In: J. Chen; J. Jiang; S. Nayak (eds.) (2008): ISPRS Congress Proc. XXXVII, Part B4, Commission IV. Beijing, China: 745-750.
- Bezema, R., Müller, M. U., Rubach, H. Poth, A. and Taddei, U. (2007): Interoperabilität für 3D-Geodaten - Erfahrungen mit CityGML und OGC Web Services. In: Strobl, J., Blaschke, T. & Griesebner, G. (eds.): Proc. AGIT 2007, Wichmann, Heidelberg.
- Bolten, A., Hoffmeister, D., Willmes, C., Bubenzer, O., and Bareth, G. (2010): Adapting the data management concept of the CRC/TR32 "SVA-Patterns" to CRC 806 "Our way to Europe". In: Curdt, C. and Bareth, G. (eds.): Proceedings of the Data Management Workshop, 29.-30.10.2009. Kölner Geographische Arbeiten, Heft 90, Köln: 7-11.
- Cuthbert, A. (1998): User interaction with geospatial data. OGC Technical Report 98-060.
- Döllner, J. (2005): Geovisualization and Real-Time 3D Computer Graphics. In: Dykes, J., MacEachren, A. M. & Kraak, M.-J. (eds.) Exploring Geovisualization, Elsevier, Amsterdam, 325-343.
- Hagedorn, B., Schilling, A., Neubauer, S. and Zipf, A. (2008): 3D Portrayal Services – Use Cases. Discussion Paper OGC 08-140, Open Geospatial Consortium Inc., URL: http://www.webviewservice.org/_media/2009-09-17_3d_portrayal_services_-_use_cases.pdf.
- Hagedorn, B. (2010): Web View Service Discussion Paper. OGC 09-166r2. URL: http://portal.opengeospatial.org/files/?artifact_id=37257.
- Hennig, S.D. (2008): Prozessierung von Laserscandaten zur Erstellung eines 3D-Stadtmodells: CampusGIS-3D. Unpublished Diploma Thesis at Institute of Geography, University of Cologne.
- Kolbe, T. H. (2004): Interoperable 3D-Visualisierung (3D Web Map Server). In: Tagungsband zum Symposium Praktische Kartographie, Nummer 9 in Kartographische Schriften, Kirschbaum Verlag, Bonn.
- Kolbe, T. H., Gröger, G., Czerwinski, A. and Nagel, C. (2008): OpenGIS City Geography Markup Language (CityGML) Encoding Standard. Technical Report OGC 08-007r1, Open Geospatial Consortium Inc., URL: <http://www.opengeospatial.org/standards/citygml>.

Quadt, U. and Kolbe, T. (2005): Web 3D Service (W3DS) Discussion Paper. Technical Report OGC 09-018, Open Geospatial Consortium, URL <http://portal.opengeospatial.org/files/?artifact-id=8869>.

Schneider, M. (2008): Integration adaptiver Multiresolution-Mechanismen in ein 3D-Typsistem für erweiterbare RDBMS. Unpublished Diploma Thesis at Institute of Computer Science, Department III, University of Bonn.

Singh, R. (2001): Web Terrain Server. Open Geospatial Consortium Discussion Paper, OGC 01-061. URL: http://portal.opengeospatial.org/files/?artifact_id=1072.

Willmes, C. (2009): 3D-Geodatensvisualisierung im Internet: Methoden und Anwendungen. Unpublished Diploma Thesis at Institute of Geography, University of Cologne.

Zipf, A., J. Basanow, P. Neis, S. Neubauer and A. Schilling (2007): Towards 3D Spatial Data Infrastructures (3D-SDI) based on Open Standards - experiences, results and future issues. In: "3D GeoInfo07". ISPRS WG IV/8 International Workshop on 3D Geo-Information: Requirements, Acquisition, Modeling, Analysis, Visualization. Delft, Netherlands.

Internet References

Apache Software Foundation (2010a): Apache Tomcat. URL: <http://tomcat.apache.org/>. 4-28-2010.

Apache Software Foundation (2010b): Apache Xerces Java Parser. URL: <http://xerces.apache.org/xerces-j/>. 4-28-2010.

Apache Software Foundation (2010c): Apache HTTP Server Project. <http://httpd.apache.org/>. 4-29-2010.

Blender (2010): Blender content creation suite. URL: <http://www.blender.org/>. 4-28-2010.

CampusGIS (2010): The CampusGIS of the University of Cologne. URL: <http://www.campusgis.de/>. 4-29-2010.

CampusGIS-3D (2010): CampusGIS-3D web interface. URL: <http://www.campusgis.de/3D/>. 4-29-2010

CIGAR-CSI (2010): SRTM 90m Digital Elevation Data. URL: <http://srtm.csi.cgiar.org/>. 4-26-2010.

CityGMLConverter (2010): CityGMLConverter tool. URL: <http://campusgis.de/3D/CityGMLConverter/>. 4-28-2010.

CityGML-Toolchain (2010): CityGML-Toolchain. URL: <http://www.citygml.de/>. 4-28-2010.

CRC806 Database (2010): Database of the Collaborative Research Centre 806. URL: <http://sfb806db.uni-koeln.de/>. 4-28-2010.

deegree (2010): The deegree framework. URL: <http://www.deegree.org/>. 4-28-2010.

ERSDAC (2010): ASTER GDEM. URL: <http://www.gdem.aster.ersdac.or.jp/>. 4-26-2010.

GDAL (2010): Geospatial Data Abstraction Library. URL: <http://www.gdal.org/>. 4-26-2010.

GDI-3D (2010): Spatial Data Infrastructure for 3D-Geodata. URL: <http://www.gdi-3d.de/>. 4-28-2010.

Google Inc. (2010a): Google Building Maker. URL: <http://sketchup.google.com/3dwh/buildingmaker.html>. 4-28-2010.

Google Inc. (2010b): Sketchup. URL: <http://sketchup.google.com/>. 4-28-2010.

Google Inc. (2010c): Google Maps Elevation Web Service. URL: <http://code.google.com/intl/en/apis/maps/documentation/elevation/>. 4-28-2010.

jQuery (2010): jQuery JavaScript Library. URL: <http://jquery.com/>. 4-28-2010.

OpenLayers (2010): OpenLayers: Free Maps for the Web. URL: <http://openlayers.org/>. 4-28-2010.

World Wind Java (2010): NASA World Wind Java SDK. URL: <http://worldwind.arc.nasa.gov/java/>. 4-28-2010.



Christian Willmes is a research fellow and PhD candidate of the GIS & Remote Sensing Working Group at Institute of Geography of the University of Cologne. He studied at the Universities of Bonn and Cologne and has a diploma degree in Geography with minor in Computer Science and Cartography. His main research interests are in geovisualization and spatial data infrastructures.