Multi-format semantically rich XML based visualization of urban structures

PETER A. HENNING, Karlsruhe

Reprint of an article that appeared in:
Zeitschrift für Photogrammetrie und Fernerkundung **2/2008** (2008), 121-132

Keywords: eGovernment, CityGML, XSLT, visualization.

Summary: The article describes the architecture of a completely XML-based geoinformation system, whose output follows a generator-transformer-serializer concept. This allows to generate tables and graphics as well as virtual reality models from a data base filled with land register data. Internal calculations and transformations are done with XSLT, which allows the integration with latest web technologies also for editing purpose. The modular system architecture may incorporate arbitrarily heterogeneous data sources and therefore is well suited for the integration of GIS and general workflow management systems anticipated in public administration.

Zusammenfassung: Vorgestellt wird die Architektur eines vollständig XML-basierten Geoinformationssystems, bei dem die Darstellung auf einem modularen Generator-Transformer-Serializer-Konzept beruht. Damit lassen sich aus einer mit Katasterdaten gefüllten Datenbasis sowohl Tabellen und Grafiken, als auch Virtual Reality-Darstellungen ganzer Städte erzeugen. Interne Berechnungen und Transformationen werden durch XSLT realisiert, dies erlaubt die Integration mit den neuesten Web-Technologien auch für die Dateneingabe. Die modulare Systemarchitektur kann beliebig heterogene Datenquellen bedienen und eignet sich darum für die im öffentlichen Bereich erwartete Integration mit allgemeinen Workflow Management Systemen.

1 Introduction

Computer scientists have envisioned the so-called *Semantic Web*, where all data carries meta-data (=semantic annotation) and is linked and integrated into a huge, globally accessible system. At the core of this Semantic Web are documents based on the Extensible Markup Language XML – or rather its *applications,* since XML is a meta-language and useful only for deriving consistent problem-oriented sets of tags, called applications [W3CXML]. While the Semantic Web might be the only way to handle the increase in the world wide data volume [LYMAN, P. &VARIAN, H. , 2004], its methods have also opened new possibilities for the interoperability of applications. For this latter reason, XML technologies have become an increasingly important paradigm in any part of information science, and consequently also in geoinformation science.

This concerns, first of all, the handling and storage of geodata [ABDUL-RAHMAN, A., ZLATANOVA, S., COORS, V. , 2006]. The relevant applications here range from generic (and therefore, from the standpoint of the Semantic Web, *low-level*) languages like the Geometrical Markup Language GML[GML] to very ambitious projects like CityGML [GRÖGER, G., KOLBE, T.H. & CZERWINSKI, A. , 2007; CITYGML].

Secondly, XML applications have become increasingly important also for visualization purpose, such as the vector graphics format SVG (Scalable Vector Graphics [W3CSVG]) and 3D modeling languages such as X3D [WEB3D], KML [KML] and COLLADA [COLLADA].

There is however a third aspect of geoinformation science where XML is bound to play a major role. Most of the geo-administrative tasks carried out in modern cities are embedded in complicated *workflows* involving several partners and various data sources. Consequently, these parts of a city administration are slow to operate and expensive to maintain. For similarly complicated workflows in the private business area, generic electronic *workflow engines* have come up in the past few years [WFMC]. It may therefore be anticipated that such workflow engines will eventually also emerge in (and of course revolutionarize) the public administration. They are, of course, operated by XML-based workflow descriptions – and therefore XML applications are also at the core of the upcoming

workflow revolution [BASTIDA MERINO, L., BENGURIA ELGUEZABAL, G. , 2005; HENNING, P.A. , 2007; BPDL].
In the following we describe a prototype system where this latter aspect of XML has been put to use. In the workflow oriented geoinformation system WB3, all calculations *and* transformations are XML-based, and the workflow is described in an XML file as a set of generator-transformer-serializer pipelines, implementing the three layers of data-workflow-visualization of GIS. It is therefore conformant to the OGC Reference model [PERCIVALL, G., ET. AL., 2003]. However, our system goes beyond this scheme in the aspect that the term "generator" may also apply to a user interface form – and the "serializer" then may consist of a database entry operation, or even in the presentation of another user interface page.
It must be emphasized however, that for performance reasons we have not built our system around a *generic* workflow engine, and are not using a high-level workflow description language like BPDL. Nevertheless WB3 allows to cross-link a multitude of applications, data sources, customer requirements and handling agents into a complex system. Let us note, that this is a different type of complexity than found in traditional large monolithic software systems – rather, it may be seen as the direct implementation of Semantic Web technology in geoinformation science and is well in line with current attempts to create geodata infrastructures.
For all the internal calculations we have used XSLT, the Extensible Stylesheet Language – Transformations [W3CXSL]. XSLT is a set of tags that allows for a formal description of data transformations, with input and output of this transformation being XML compatible documents. This has been shown to be quite useful in geoscience, and already a considerable number of research efforts has been devoted to using XSLT in transforming geodata structures.
Much less well known is the fact that XSLT is indeed a full-fledged universal programming language, which follows the so-called *functional programming paradigm* [HENNING, P.A: &VOGELSANG, H. , 2006]. Consequently by using XSLT any formally describable operation on data may be performed. This includes the generation of semantic meta-data by tapping into other data sources, but also allows to treat legacy systems. Just describe the operation that has to be performed, implement the description as XSLT program – and be done with it, such is the theory.

2 Technology

The technological core of the system WB3 is an Open Source product, the Apache Cocoon Server [COCOON]. It is uniquely well suited for the purpose addressed here because it exhibits a generator-transformer-serializer concept: Data is read e.g. from an XML skeleton file filled with some parameters from a Web page, then passed through one or more transformer stages and then finalized by a serializer into an output format, e.g. another Web page. Such a sequence is called a *pipeline* in the Cocoon vocabulary, Fig. 1 shows a typical pipeline. Numerous modules exist for generation, transformation and serialization of data – including "fancy" code to present calendar inserts or to augment MP3 files. For our purpose we currently employ only a few of these possibilities, since most of the data generation is done by filling a predefined XML file with some parameters of the calling Web request.

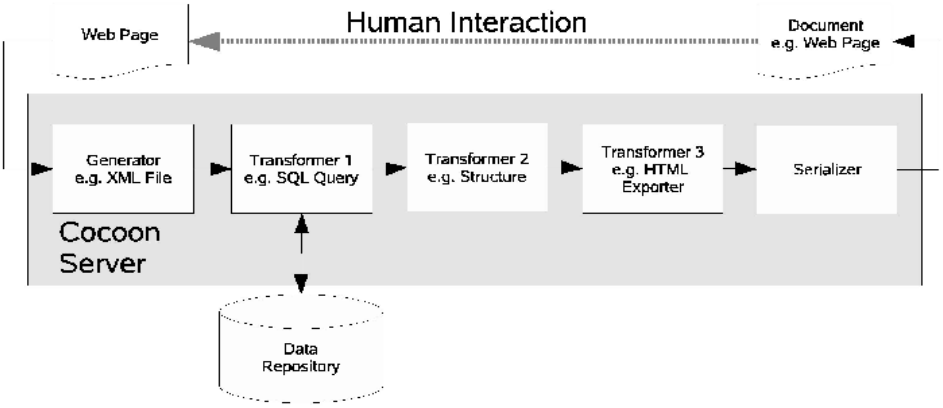

Fig.1: Typical Cocoon pipeline consisting of generator, three transformer blocks and a serializer.
In most cases the first transformation following the generator is an SQL transformation: The initial skeleton file together with the actual parameters passed by Cocoon forms an XSQL query, which is run through a database. While this might be an unorthodox view in the sense of usual programming paradigms, it fits well into the *functional programming paradigm* – where a database is nothing but a function performed on some parameters (the database query).
The database addressed in the WB3 project is another Open Source product, a mySQL database with several tables containing data on grounds, buildings, streets and other objects. It should be noted, that Cocoon works independently of the database used – and therefore might be run on top of any other geoinformation database.
Results of database queries are, from the viewpoint of modern markup languages, rather flat and structureless objects – one or more simple tables. For the purpose of WB3, some structure has to be brought into these results, requiring another transformation specified in the XSLT language.
For our example pipeline, the structured XML file resulting from this transformation is then exported into a certain format – a page, which is run through a simple HTML serializer to produce another Web page. This simple example however does not reflect the various possibilities of the Cocoon system. In particular, it is possible to merge results of two different pipelines in the Cocoon server. As part of our system, this feature is used to include complex image maps into HTML pages. One may also address another pipeline by referring to an image in a generated Web page, thereby merging different pipelines outside the Cocoon server. This is depicted in Fig. 2.

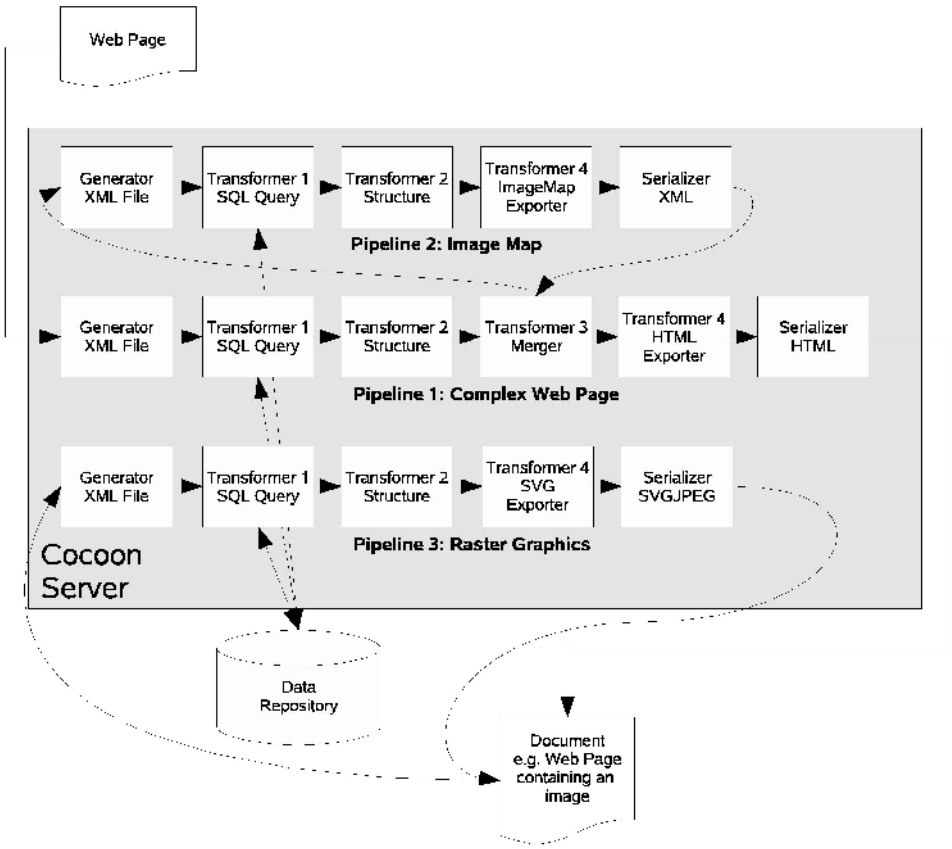

Fig.2: Merging several Cocoon pipelines for the production of a complex Web page.

In WB3, the XSL transformations also contain the *business logic,* which involves string manipulations as well as numerical calculations. The functional programming possible with XSLT has certain advantages over conventional programming logic:

1. Since no state variables exist in functional programming, function calls are, in the language of computer science, free from side effects. This implies that function calls cannot influence each other – and consequently, one of the core tasks in geoinformation systems, namely coordinate transformations, may be implemented much more safely.
2. Function calls are possible with arbitrary recursion, and the concept of an operator, i.e., a function action on functions is quite natural. It is not by chance that the first computer algebra systems have been realized using the functional programming language LISP.
3. According to industrial power users, functional programming leads to more abstract and easier verifiable source code, and consequently reduces the error rate in software development dramatically [ARMSTRONG, J. , 2003].
4. Since XSLT code itself comes in XML format, it is structurally verifiable. This means, that syntactical as well as semantical errors are even more easier to discover.

Fig. 3 shows a few lines of XSLT source code (not relevant for the WB3 system) to demonstrate some of these features.

```
<template name="ggt">
     <param name="x"/>
     <param name="y"/>
     <choose>
        <when test="$y=0"><value-of select="$x"/></when>
         <otherwise>
               <call-template name="ggt">
                      <with-param name="x" select="$y"/>
                      <with-param name="y">
                          <call-template name="modulo">
                              <with-param name="xx" select="$x"/>
                              <with-param name="yy" select="$y"/>
                          </call-template></with-param>
               </call-template></otherwise>
      </choose>
 </template>
```

Fig.3: Example for XSLT source code – here a simple implementation of an algorithm to find the greatest common divisor of two integer number by recursive call of the function (=template) named "ggt".

A price one has to pay for this "secure" development is the much higher amount of writing needed - but on the other hand the code is quite readable even to non-developing persons. Another drawback is the fact that so far XSLT code is *interpreted* – which means, that much of modern computing power is lost. However, XSLT "compilers" seem to be around the corner – which would then translate XSLT code into Java classes that have a much higher performance.
A natural question arising in the context of technology is the performance of the system. The Cocoon server exhibits a nice caching mechanism – which up to now is not synchronized with the caching mechanism of the connected data base. Consequently we have found it advantageous to keep the result of database queries as temporary files, e.g., to manually extend the caching of Cocoon to the database. With such a component in place we are able to render any of the output formats for approximately 1.000 houses within a fews seconds on an Athlon 64 computer. It is therefore not a far-fetched

guess to estimate, that some 90.000 buildings may be a manageable data volume for the current prototype system running on a single PC.

3 Data Acquisition and Storage

The WB3 data repository is - for each city - organized in two main tables which may be filled automatically by point and line data obtained from a state agency [LVABW].

1. A table of grounds, each carrying a unique identifier. Its edge points are specified in a city-local coordinate system as meters "right" (=east) and "up"(=north) from a central Gauss-Krueger reference point. While we found this very convenient to handle, it is of course easily adopted to a standard system, where coordinate points are kept in a separate table and referenced by name.
2. A table of building data with edge points specified in a ground-local coordinate system. These ground-local systems are automatically aligned with the northwest corner of the ground and a building side– thereby allowing for easy user-friendly measurements local to the ground while maintaining the geospatial reference. Since the building data – even when obtained from state agencies - is of mixed quality, these reference points may be modified on a per-building basis. Building parts on the same ground are automatically grouped.

Let us note, that this storage in a ground-local system is not necessary for the application – one might easily switch to an absolute coordinate system also for the building data. The transformation concept of WB3 allows to implement this within a few lines. However, when presenting data to the *owner* of a building, he might be more familiar with the data in a ground-local system than with absolute data. In this sense, the building database was found easier to maintain in its present form. Furthermore, although the usage of ground-local coordinate systems is certainly not an OGC conformant service and therefore might seem unusual for geographers, it is still compliant to OGC standards on coordinate referencing [LOTT, R., ET. AL., 2004].

Fig.4 depicts the geometrical organization of this repository. However, these initial data structures are far from complete enough to perform even the most basic public services. The missing data items, like e.g. ownership information, addresses and building heights are, to a large extent, found in data bases run by the city administration. From these they may be imported semi-automatically. Manual data entry for the city is also necessary to fill optional additional structures:

3. A table of street names and their reference to certain ground identifiers.
4. Tables for vegetation covers and solitary vegetation objects, like e.g. trees.
5. A table of underground objects in the form of piecewise linear curves, like e.g. sewage pipes.

Since it is clear, that manual data entry must be part of a useful city information system, a security concept as well as an editing concept are needed. The WB3 system therefore contains a role based access control (RBAC) security system: *City editors* may read, write and delete any data item, while *administrative employees* may delete and write only meta data – but read the full data available. The general public may read only part of the data, for example in the form of graphical overviews of the whole city - but never the tabular data on individual buildings and grounds. *Home owners* may be given a special password access to read their own full geometrical and meta data – and to make suggestions for improval of data items. Official data on buildings used for the prototype turned out to be incomplete (missing buildings), outdated (missing addenda to buildings) and imprecise (some points being as much as 0.5 m off reality), such a concept might indeed improve the data quality without measurable investment.

Since this security system is implemented as part of the WB3 server, it serves as an envelope for other security issues. Let us, for example, consider a heterogeneous distributed system where part of the necessary data is acquired via a third-party web service (like e.g. a particular 3D architectural model). The special *role* of a WB3 user then may or may not include access to this web service.

Roles therefore have to be seen orthogonal to dedicated user names and passwords. The RBAC system may be extended to an arbitrary fine granularity, solving every conceivable problem of access rights propagation. On the other hand, due to its modular structure, WB3 may be easily modified to run a user authentication against a Radius server or an LDAP system that is part of a larger eGovern-

ment system. In any case it may be safely stated that the RBAC system of WB3 is fully conformant to the digital rights management (DRM) gatekeeper model of the OGC [VOWLES, G. ET.AL., 2006].

4 Editing data

For writing data into WB3, the entitled editor is presented with comfortable forms via internet, consequently no special software is needed. Standard internet encryption techniques (secure encryption and digital signatures for documents) allow for usage even in non-secure environments. Moreover, an entitled editor may enter coordinate data necessary for these tables graphically, since he is presented with a map generated from the initial data. Since this map is rasterized from an SVG representation, it may be enlarged almost indefinitely – and a person entering, for example, a "tree" into the vegetation table may do so by clicking into the exact spot on the map. Even for this graphical data entry, no special software is necessary. The coordinates are acquired from the rasterized map by JavaScript functions which are part of the editor forms, see Fig.5 for an overview.
It appears noteworthy, that the graphical editor functions are purely based on JavaScript. Consequently, the data transfer into the back end is minimized and all relevant coordinate transformations from pixel into the WB3 system are performed on the client side, i.e., they are running in the *editor's* computer.
Not only does this conform to the latest Web technologies like e.g. Ajax, it also allows for an easy extension. In particular, this may involve requirements like e.g. grid snapping and other geometrical and topological construction aids – which are not yet implemented in the prototype WB3 system. One may also envisage an extension of the editor functionality that couples it to completely different third party software – which would allow to use arbitrary CAD functionality for editing purpose. This is currently a trend also in other areas, where e.g. ready-made editor components for text are available for inclusion into any system as "Java beans".

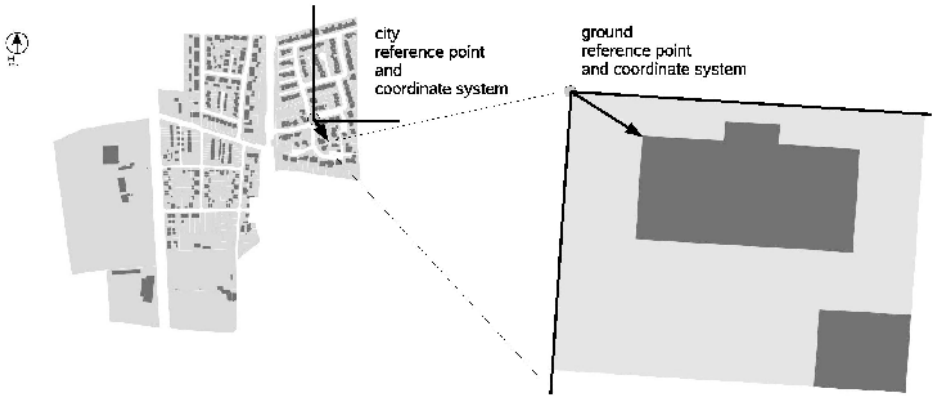

Fig.4: Example for the Scaled Vector Graphics (SVG) output of the city database after automatic import of ground and building data. Streets are removed for clarity of presentation.

With this editing concept, WB3 does no longer fit into the OGC reference model [PERCIVALL, G., ET. AL., 2003] – rather, it closes an interactive loop between data presentation and data production. From the standpoint of Semantic Web technology, this should be considered a hint towards a possible extension of such reference models.
As a final remark concerning the graphical editor component it must be emphasized, that although its display currently is based on a rasterized picture, one could also couple the JavaScript functions into an SVG display. This would allow for a graphical editor component that – while still being completely web-based – can be of any desired precision.

5 Visualization of City Data

Of course, each data layer may be presented as web pages, including two-dimensional graphical representations. These pages can also be transformed into yet another XML application, the XSL Formatting Objects. These are then serialized by a standard component of Cocoon into a PDF document, thus producing printable output of the data collected in WB3. Since standard methods allow for a digital signature of these PDF documents, the system is well suited to print official documents without requiring manual intervention – an important aspect for eGovernment.

The ground and building parts of the two-dimensional graphical data are obtained from the official land register. Additional data entered manually falls in two categories: Vegetation structures are drawn as (semitransparent) areas in the two-dimensional plans, whereas underground pipes are drawn as collections of points and piecewise linear curves. Apart from their presentation in interactive city maps, the 2D maps are also needed for the graphical editing forms.

Three-dimensional representations are obtained from imported land register data using a default height for buildings, as e.g. contained in the city regulatory codes. Underground pipes are included in these as cylinder arrays, i.e. piecewise linear tubes, and vegetation as blocks or artificial tree-like structures. Even in this LOD (Level of Detail) 1, the output may be useful for planning purpose – ranging from an underground view showing sewage and water pipes and electrical cables to simulations on noise and wind propagation. We consider this underground view as another innovative step, since it provides an added value for administrative purpose.

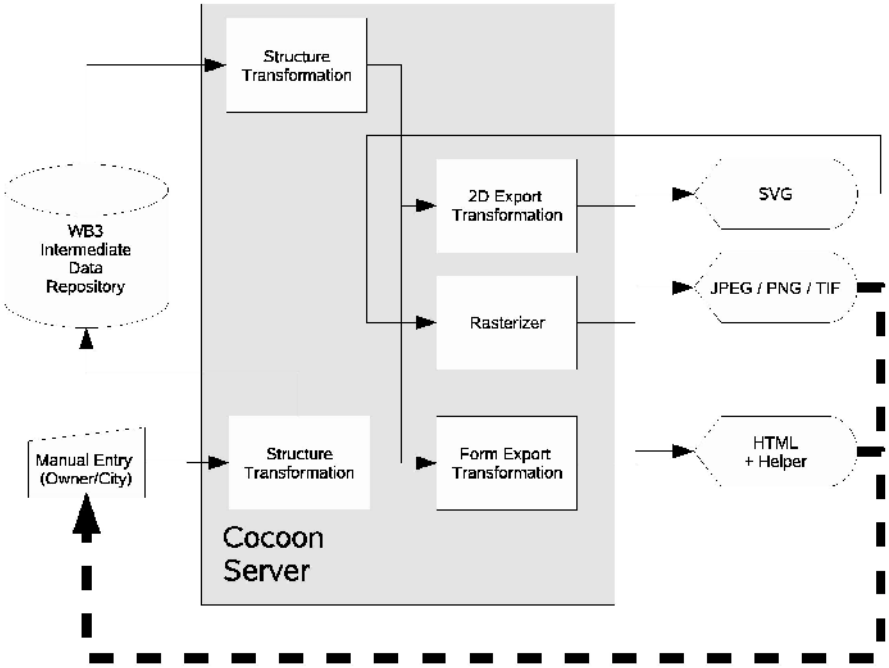

Fig.5: Two-dimensional representations and editing concept of WB3.

If a *city editor* enters coordinates for a roof line, a roof type and height values for storeys, the system generates a realistic form for the outside of the building, including a roof according to specification. While such an LOD 2 model could be calculated from the data in real-time, we found it to be more useful to do this calculation only once when the building editor form is saved – hence *caching* the results of the geometrical calculations performed to acquire a roof in a special database field. It must be emphasized, that this is not a return to old-fashioned storage methods – since there is no way to dir-

ectly edit this “roof and side” cache, and since it will be overwritten with each modification of the building. It is solely maintained for performance reasons.
Finally, the system WB3 also allows to place links in the building table of the data repository. Two different links are possible. One of them points to arbitrary internet location for information purpose, hence 2D as well as 3D data may be part of a city information system. Such a system has recently been implemented for the City of Berlin based on a 3D model in Google Earth [BERLIN].
The second type of links connects the system to three-dimensional models. In accordance with established standards, we label this as LOD 4, see Fig. 6. LOD 3, which would be a more detailed model than LOD 2 (but without interior pieces), as well as textures are not yet fully implemented in WB3 (see note in the summary).
Clearly, this linking is not limited to static 3D visualization models (as we have implemented it) – but could also involve semantic links into other systems, like e.g. CAD data bases or CAD files. We find it too early to speculate about the nature of future 3D standards for *architectural* building models (apart from the fact that they will be XML based). However, on one hand it is safe to state that for many years to come the vast majority of buildings in public database systems will *not* encompass LOD 4 data. On the other hand even in case LOD 4 data *is* available, it would unwise to incorporate CAD data into WB3 – partly because WB3 lacks features for re-use of models and version control, partly because even CAD models are written for a certain purpose and will most certainly never be fully standardized. The linking in WB3 then might also involve external transformation scenarios for these CAD models. Consequently, the method of linking to external LOD 4 models chosen here is semantically correct, flexible to implement, effective to maintain and future safe.

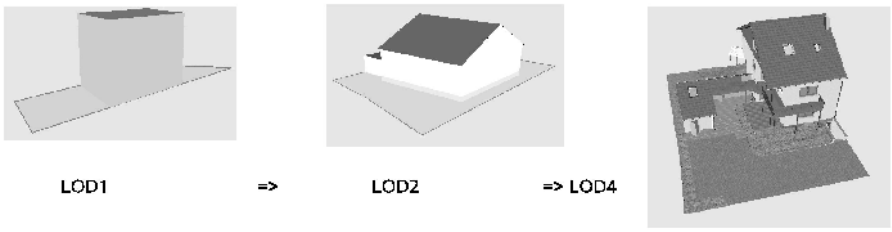

Fig. 6: Buildings are represented in different Levels of Detail (LOD)

Three-dimensional data nowadays, even if limiting oneself to textual non-binary form, comes in a variety of formats. The following formats are produced by WB3:

- X3D is an XML formulation for the Virtual Reality Modeling language VRML [WEB3D]. It is tailored to describe virtual worlds, which may be explored interactively by the user. The virtual world may contain animations and other time dependent effects. The X3D file produced in LOD 1 and LOD 2 will be placed in a so-called LOD node, if a link to a static VRML or X3D file is contained in the database. If the virtual world is explored by the user, and he or she approaches the LOD node, the rather simple LOD 1or LOD 2 model is replaced by the embedded VRML file. The combined X3D/VRML world is centered around the city-local Gauss-Krueger reference point.
- KML, the Keyhole Markup Language, is a rather simple X3D format heavily used for 3D building models in the application Google Earth [KML]. Its particularity is the geo-referentiation of the coordinates used. They are derived from WB3 internal coordinate representations (see Fig. 4) by an XSL transformation.
- COLLADA is a rather new XML format for 3D models [COLLADA]. While it is also displayed in Google Earth, it is mainly targeted to easy and fast rendering in game programming situations. Compared to the X3D representation, it is therefore closer to rendering en-

gines and open graphics standards like OpenGL. An OpenGL renderer for COLLADA may be seen similar to a rasterizer for SVG data. COLLADA is most easily used in simulations where city data is needed – like e.g. driving simulators.

- CityGML is more than just another three-dimensional format, because it is targeted at reproducing the complete semantical information known about a city and its objects. It is therefore not really a display format, but a format to exchange data with other applications [KOLBE & GROEGER , 2005; GRÖGER, G., KOLBE, T.H. & CZERWINSKI, A. , 2007].

Fig. 7 depicts the various export formats and their relationship. This picture also emphasizes the modular structure of the WB3 system, composed of re-usable transformation blocks written in XSLT.

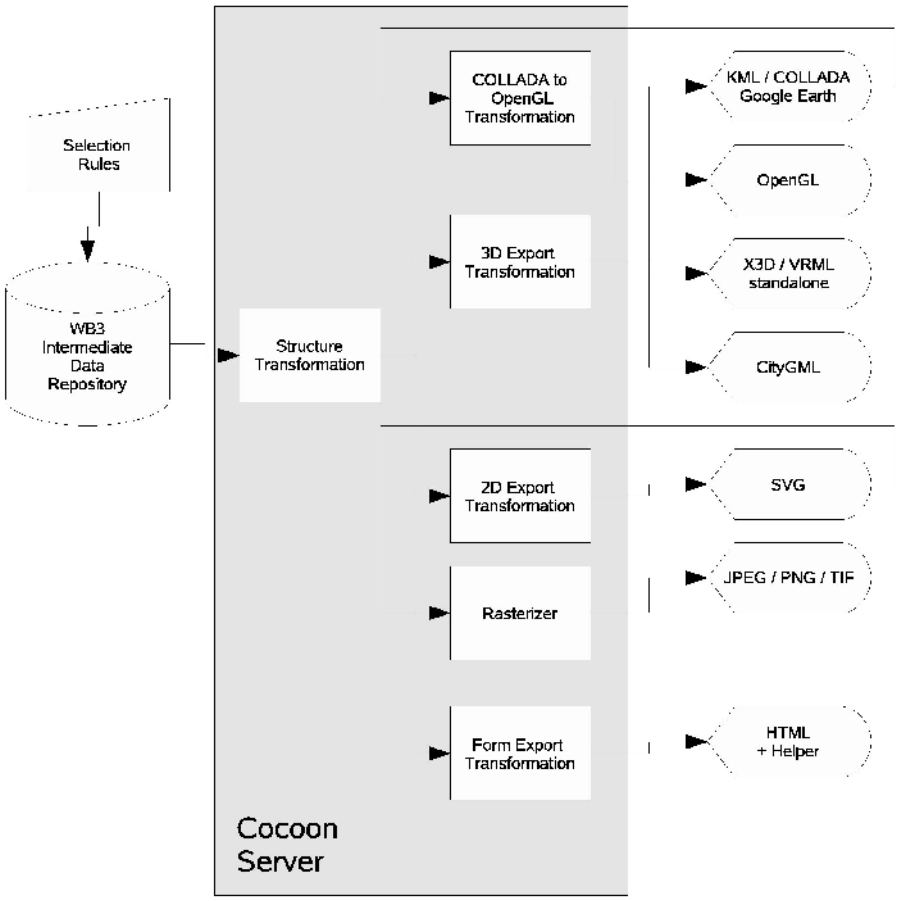

Fig. 7: Different 2D and 3D export formats

6 Summary

First of all it should be made clear, that the prototype implementation still lacks a lot of functionality that is already available in state-of-the-art commercial geoinformation systems, like e.g. spatial analysis. However, his is just a temporary limitation bound to disappear when more functionality is added as described below.

The existing prototype system therefore has to be distinguished from the general (informational) architecture of WB3. Whereas the former is, well, a prototype – the latter is indeed more than "just an-

other" geoinfomation system. In the following we summarize our findings from the more general standpoint.

1. The modular generator-transformer-serializer concept fits into the *thin client* picture of the OGC [PERCIVALL, G., ET. AL., 2003] and is a clean implementation of the usual visualization pipeline. Within this framework, the web browser together with its standard plugins serves as *thin client*, whereas the WB3 system serves as integrator and partially as renderer. However, adding editor functionality in terms of JavaScript functions to the thin client as is done in WB3 then shifts it somewhat more into the direction of a *thick client*. Therefore, as stated above it may be concluded, that the the OGC model for portraying is slightly inadequate for WB3 as the former does not combine data presentation and data entry, e.g., it lacks the interactive aspects of the Semantic Web.
2. One might argue, that the data storage layer implemented in the prototype system is not OGC compliant. However, due to the modularity it is easily possible to implement another storage system as bottom layer of the system. This adaptability is due to the use of XSLT and combines naturally with the concept of functional programming and workflow orientation of the system.
3. Since WB3 is completely modular and based on XML even for the internal calculations, it can be expanded or modified very easily. In particular, it may be customized to a large extent - which sets it apart from large monolithic software systems It was proven over the past two years, that even the open source development model of distributed software development is feasible for such a system.Within the past six weeks before publication, for example, a rudimentary texturing tool was added. Planned extensions include a Web Map Service and a Web Feature Service – which, in the view of WB3 are just another XML output.
4. As argued above, this strict XML orientation forms a unique opportunity to integrate it with generic *workflow management systems* that are bound to appear in public administration. Such generic systems are much more flexible than the dedicated workflow concepts of existing and upcoming commercial GIS, because they may also incorporate completely different steps outside the usual city geographers' tasks. One could envisage a city information system, where also sociological and archaeological informations are kept along with modern data. In this context it seems worthwhile to mention that in one of the prototype villages available in WB3 we have already entered roman building structures from 200 A.D. - and conversely integrated specially processed WB3 graphics output into a virtual museum application.
5. Since many data formats useful for visualization nowadays come as XML applications, WB3 forms a natural base for an export into any of these formats. This encompasses both semantically poor formats like COLLADA and semantically rich formats like CityGML. In other words, WB3 is also a tool to transform among these formats. Since they differ in their ontological basis, WB3 therefore may be considered an ontological mapping tool.
6. As we have shown, XSLT serves as an integrating glue, binding together various types of data from a multitude of sources without ever modifying the legacy systems. Semantic meta-data necessary to switch the ontological basis is then acquired in the XSLT transformation descriptions – which may change with time more easily than Document Type Definitions or XML Schema files. We therefore conclude, that the rise of XML based declarative languages for geoinformation purpose will be followed by an increasing usage of XSLT to ensure interoperability.

While the idea of a *Semantic Web* in its pure form seems to be a rather theoretical concept interesting mainly to computer scientists, its core idea of managing a world wide data repository by semantic annotation (XML) and transformation (via XSLT) may become to be a key element to the future of geodata management.

The prototype of the system WB3 is accessible at http://www.ilias-karlsruhe.de:8888/wb3/

References

ABDUL-RAHMAN, A., ZLATANOVA, S., COORS, V. , 2006: Innovations in 3D Geo Information Systems. Lecture Notes in Geoinformation and Cartography (Springer, Heidelberg 2006)

ARMSTRONG, J. , 2003: Making reliable distributed systems in the presence of software errors, (Dissertation, KTH Stockholm 2003)

BASTIDA MERINO, L., BENGURIA ELGUEZABAL, G. , 2005: Business Process Definition Languages Versus Traditional Methods Towards Interoperability, *in:* COTS-Based Software Systems (Springer, Heidelberg 2005), 25-35

GRÖGER, G., KOLBE, T.H. & CZERWINSKI, A. , 2007: City Geography Markup Language (CityGML) OGC Best Practices Document, Version 0.4.0, (OGC 07-062 2007).

HENNING, P.A. & VOGELSANG, H. , 2006: Handbuch Programmiersprachen (Hanser, München 2006)

HENNING, P.A. , 2007: Open Source Lösungen zur Integration von Hochschulmanagement und eLearning, Proceedings der 21. DFN-Arbeitstagung Kommunikationsnetze, Kaiserslautern 2007 (GI Lecture Notes, im Druck)

KOLBE, T.H & GRÖGER, G. , 2005: 3D-Stadtmodellierung auf der Basis internationaler GI-Standards. *in:* COORS&ZIP (Hg.): 3D-Geoinformationssysteme (Wichmann, 2005) 243-249

LOTT, R., ET. AL., 2004: OGC Abstract Specification Topic 2, Spatial referencing by coordinates (OGC 04-046r3)

LVABW: Landesvermessungsamt Baden-Württemberg, Copyright for geodata used in WB3

LYMAN, P. & VARIAN, H., 2004: How much information ?, http://www.sims.berkeley.edu/research/projects

PERCIVALL, G., ET. AL., 2003: The OGC Reference Model (OGC 03-040, 2003)

VOWLES, G., ET. AL., 2006: The OpenGIS® Abstract Specification, Topic 18: Geospatial Digital Rights Management Reference Model (GeoDRM RM) (OGC 06-004r4, 2006)

BERLIN:	3D Model of Berlin, http://www.3d-stadtmodell-berlin.de
CITYGML:	Resources for CityGML, http://www.citygml.org/
COCOON:	Resources for the Apache Cocoon Server, http://cocoon.apache.org/
COLLADA:	Resources for the COLLADA Community, http://www.collada.org
GML:	Resources for the Geography Markup Language, http://www.opengis.net/gml
KML:	Resources for the Keyhole Markup Language KML, http://code.google.com/apis/kml/documentation/
WEB3D:	Resources of The Web 3D Consortium, http://www.web3d.org
WFMC:	Resources of the Workflow Modeling Coalition, http://www.wfmc.org
W3CXML:	XML Resources of the W3C, http://www.w3c.org/XML
W3CXSL:	W3C Resources on XSL, http://www.w3.org/Style/XSL
W3CSVG:	SVG Resources of the W3C, http://www.w3.org/Graphics/SVG/

Anschrift des Autors: Prof. Dr. rer. nat. PETER A. HENNING, media::lab, Hochschule Karlsruhe, Moltkestrasse 30, 76133 Karlsruhe, Tel.: +49-721-925-1477, Fax: +49-721-925-1509, eMail: peter.henning@medialab-karlsruhe.de