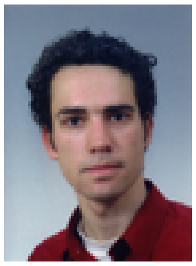
GISdevelopment.net ---> Technology ---> Geographic Information System

Print

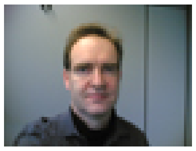
Updating geo-information in a heterogeneous networked environment – Experiences and evaluation of OpenGIS Web Feature Services



T.J. Brentjens
GIS Technology, OTB/TBM, TU Delft
The Netherlands

M.E. de Vries
GIS Technology, OTB/TBM, TU Delft
The Netherlands

C.W. Quak
GIS Technology, OTB/TBM, TU Delft
The Netherlands



C. Vijlbrief
GIS Technology, OTB/TBM, TU Delft
The Netherlands

P.J.M. van Oosterom
Kadaster, Apeldoorn, The Netherlands.

Though Internet GIS has been very popular for nearly a decade and is becoming more popular all the time, it is most often limited to simple map viewing or retrieval. Today there are many Internet servers, often conforming to the standard OpenGIS Web Map Server (WMS) protocol, mainly delivering (raster) images, which can be viewed by clients. There are also a few ‘closed’ Internet GIS applications, which allow editing of features by clients based on proprietary communication protocols. In practice this means that both the server and the client have to be of the same vendor and no heterogeneous situations are feasible; e.g. GeoShop (van den Berg et al 1997). However with the availability of the standard OpenGIS Web Feature Server (WFS) protocol, it is now possible, for the first time ever, to realize Internet based geo-information processing environments which include multiple servers offering data layers and different client types specifying the updates. This will be illustrated via a case study ‘notary drafts cadastral parcel boundary’, a relatively simple distributed editing prototype. The WFS protocol will be analysed for more advanced edit scenarios and a number of improvements of the WFS protocol are suggested.

1. Introduction

This paper evaluates the strengths and weaknesses of the OpenGIS Web Feature Services (WFS) protocol for creating distributed heterogeneous, yet interoperable geo-information systems. The evaluation is based on our experiences with the development of a WFS environment for editing cadastral data by notaries: using a simple Web client notaries can sketch new or changed cadastral parcel boundaries or edit ownership information (in their offices or in the field, with a PDA) and submit these changes to the central cadastral geo-database via an OpenGIS Web Feature Service that supports transactions.

In order to realize interoperable systems, standards must be used. The OpenGIS Consortium (OGC) has issued a number of Web service interface specifications in order to standardize the requests and responses between a Web service and a Web client. The scope of the Web Feature Service specification is not only the retrieval of geo-information over the Web, but also the editing of geo-data (both the spatial and the thematic attributes). Section 2 will give a short overview of the WFS protocol (and protocols used by WFS, such as GML and filter encoding). In section 3 our case study will be introduced. The interoperability aspect of the systems is discussed in section 4, where also a number of alternative server and client implementations are tested in cooperation with our case study prototype server and client. Based on these experiences an evaluation of the WFS protocol is given in section 5, accompanied by suggestions of future extensions/improvements of the WFS protocol. The conclusions can be found in the last part of this paper.

2. Short WFS overview

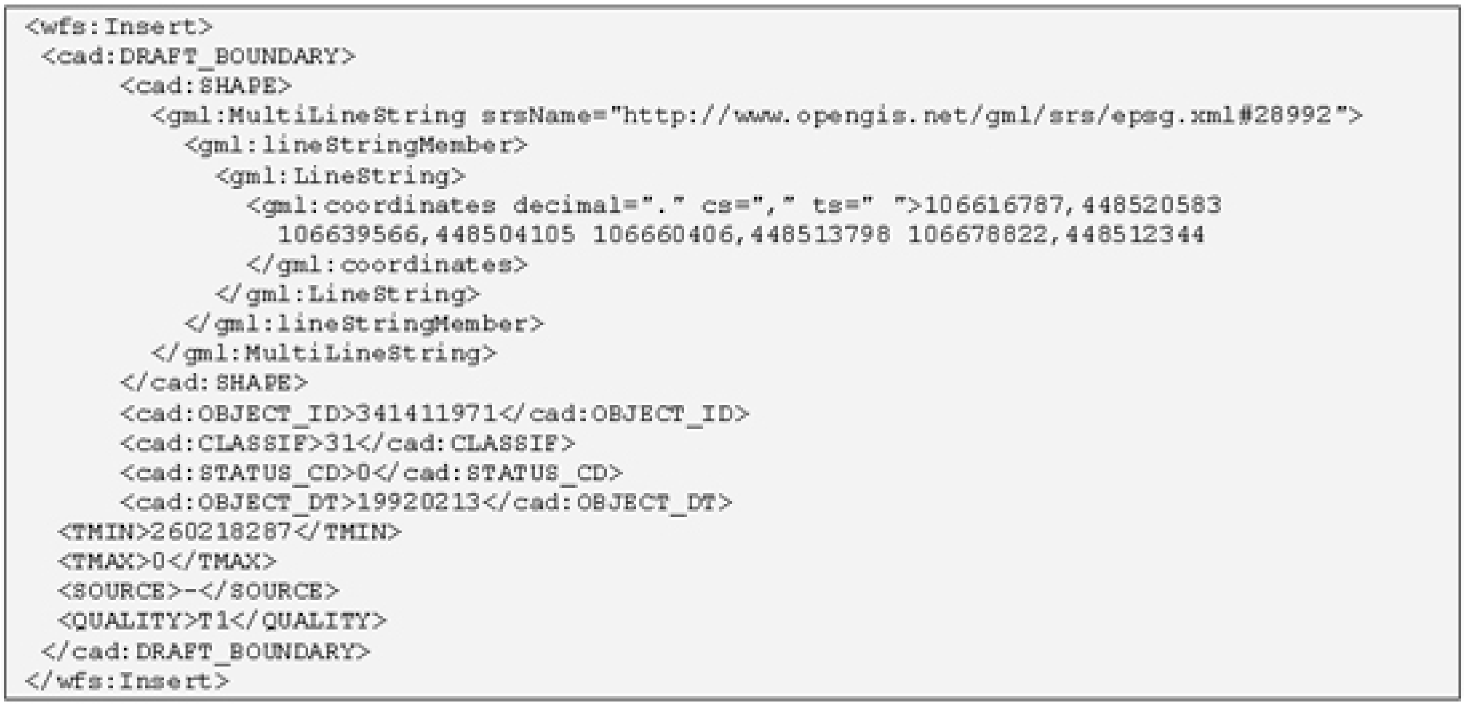
Two classes of Web Feature Services are defined by OpenGIS: Basic WFS (needed for retrieving features) and Transaction WFS (needed for editing geo-data) (OGC, 2002). The WFS protocol allows a client to retrieve geospatial (vector-) data encoded in Geography Markup Language (GML) from multiple Web Feature Services. GML is an XML encoding for the modelling, transport and storage of geographic information, including both the spatial and non-spatial properties of geographic features (OGC, 2003). The current version WFS (1.0) is based on GML 2.1.2. This has some disadvantages as also GML 3 is available with more functionality (e.g. topology, metadata, 3D primitives,…) and GML 3 will also become a ISO standard (in contrast to GML 2.1.2).

A Basic WFS implements the GetCapabilities, DescribeFeatureType and GetFeature requests. A client can request for an XML encoded capabilities document (indicating which feature types and what operations are supported) by sending the GetCapabilities request to a web feature server. The function of the DescribeFeatureType request is to generate an XML-schema (2004) description of the feature types serviced by a WFS implementation. The GetFeature request allows for the retrieval (of feature instances and selected set of their attributes) and uses filters from the OpenGIS Filter Encoding (OGC, 2001) to constrain the data to retrieve; see Figure 1.

```
http://www.someserver.com/servlet/wfs
?request=GetFeature
&FEATUREID=TEST_BOUNDARY.1000
```

Figure 1 Request to retrieve the TEST_BOUNDARY with FeatureId 1000 (KVP request using HTTP GET)

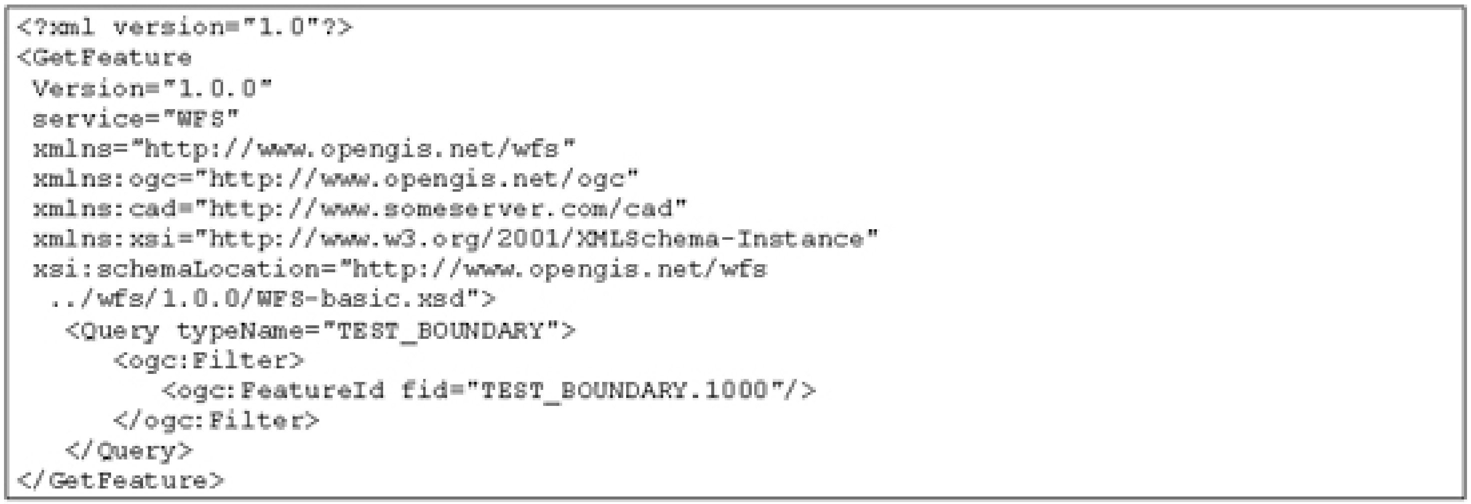
A Transactional WFS offers functionality to modify geographic features as well; that is insert, update, and delete geographic features (see Figure 2 for an insert). In order to do so, a Transactional WFS implements the Transaction request (a set of insert, delete, and update actions that belong together). It could optionally implement the LockFeature and the GetFeatureWithLock request. When the transaction has been completed, a Web Feature Service will generate an XML response document indicating the completion status of the transaction (and a list of newly generated feature identifiers assigned to the new feature instances). The purpose of the LockFeature request is to expose a long-term feature locking mechanism to ensure consistency and avoid editing by other users at the same time. A Lock element uses a filter to specify what feature instances should be locked. Finally, by using GetFeatureWithLock instead of the GetFeature request, a client requests for features to be retrieved and locked at the same time.



```
<wfs:Insert>
 <cad:DRAFT_BOUNDARY>
      <cad:SHAPE>
        <gml:MultiLineString srsName="http://www.opengis.net/gml/srs/epsg.xml#28992">
          <gml:lineStringMember>
            <gml:LineString>
              <gml:coordinates decimal="." cs="," ts=" ">106616787,448520583
                106639566,448504105 106660406,448513798 106678822,448512344
              </gml:coordinates>
            </gml:LineString>
          </gml:lineStringMember>
        </gml:MultiLineString>
      </cad:SHAPE>
      <cad:OBJECT_ID>341411971</cad:OBJECT_ID>
      <cad:CLASSIF>31</cad:CLASSIF>
      <cad:STATUS_CD>0</cad:STATUS_CD>
      <cad:OBJECT_DT>19920213</cad:OBJECT_DT>
  <TMIN>260218287</TMIN>
  <TMAX>0</TMAX>
  <SOURCE>-</SOURCE>
  <QUALITY>T1</QUALITY>
 </cad:DRAFT_BOUNDARY>
</wfs:Insert>
```

Figure 2 Fragment of the Transaction-request for a new draft_boundary.

WFS requests and responses are sent between client and server using the Hypertext Transfer Protocol (HTTP). There are two methods of encoding WFS requests. The first uses XML as the encoding language, the second uses keyword-value pairs to encode the various parameters of a request. An example of a keyword value pair (KVP) was already given in Figure 1. The same request but now as an XML encoding is given in Figure 3. In general KVP requests are shorter, but using KVP for transaction requests is limited. KVP-requests are sent using HTTP GET, while XML-requests have to be sent with HTTP POST. In both cases (XML and KVP), the response to a request or the exception report must be identical.



```
<?xml version="1.0"?>
<GetFeature
 Version="1.0.0"
 service="WFS"
 xmlns="http://www.opengis.net/wfs"
 xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:cad="http://www.someserver.com/cad"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-Instance"
 xsi:schemaLocation="http://www.opengis.net/wfs
  ../wfs/1.0.0/WFS-basic.xsd">
   <Query typeName="TEST_BOUNDARY">
      <ogc:Filter>
         <ogc:FeatureId fid="TEST_BOUNDARY.1000"/>
      </ogc:Filter>
   </Query>
</GetFeature>
```

Figure 3 Same request as in figure 1, but now as XML encoded request (using HTTP POST)

3. Case study: notary drafts parcel

As an example of cadastral transactions, one could think of a notary who sketches a new boundary because a parcel has to be divided into two new parcels as the result of a property transaction. Instead of drawing the new boundary on a paper map and sending it by postal mail to the cadastre, it would be more efficient when the notary could sketch the new boundary on a digital map in a Web client and send the digital boundaries to the cadastral database via a Web service. These draft boundaries are stored in the cadastral database as preliminary boundaries, and can be used as input when the exact boundaries are surveyed by the cadastral surveyor.

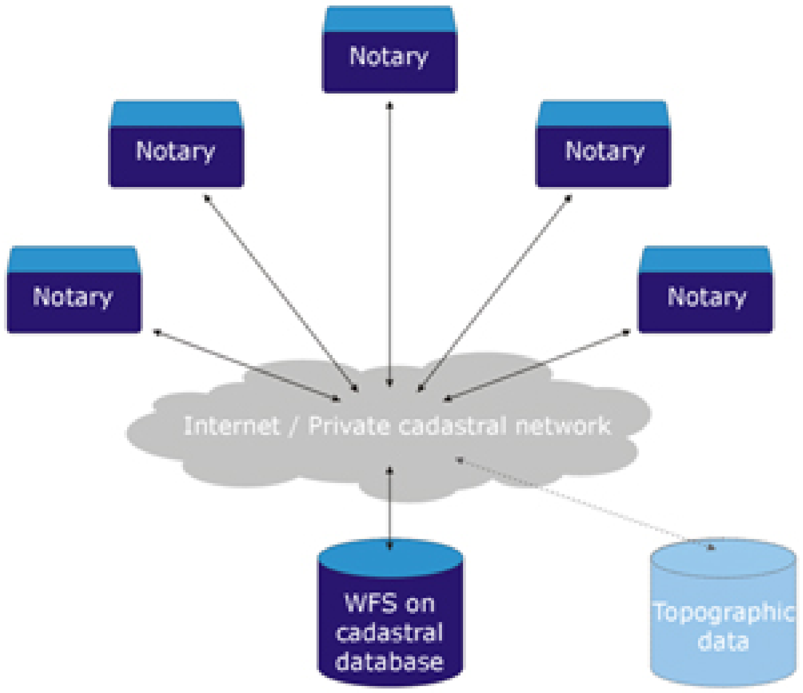

Notary
Notary
Notary
Notary
Notary
Internet / Private cadastral network
WFS on cadastral database
Topographic data

Figure 4 Web Feature Service for notaries to draft new parcels and boundaries in cadastral database, optionally using topographic data as background.

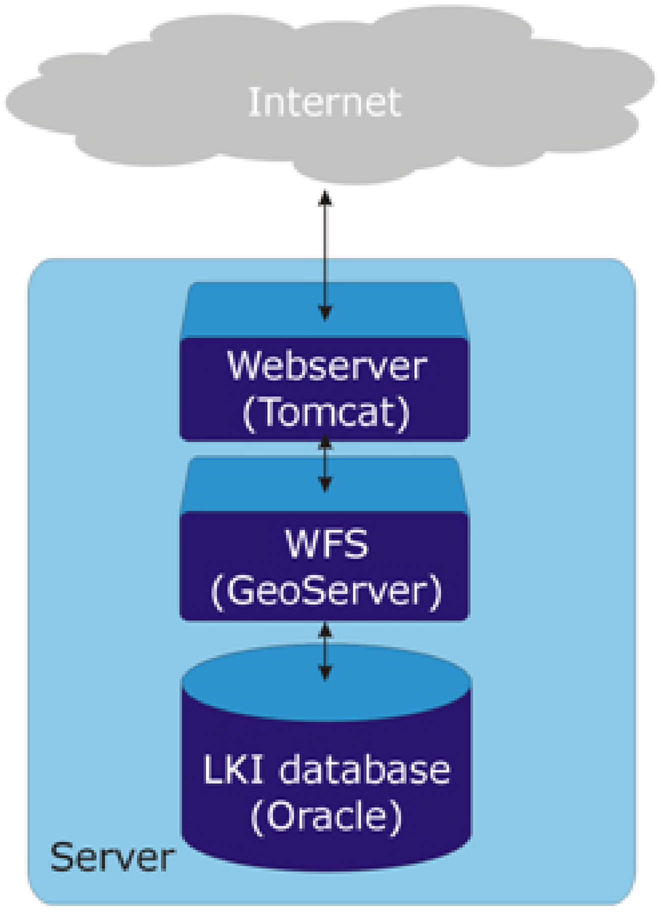
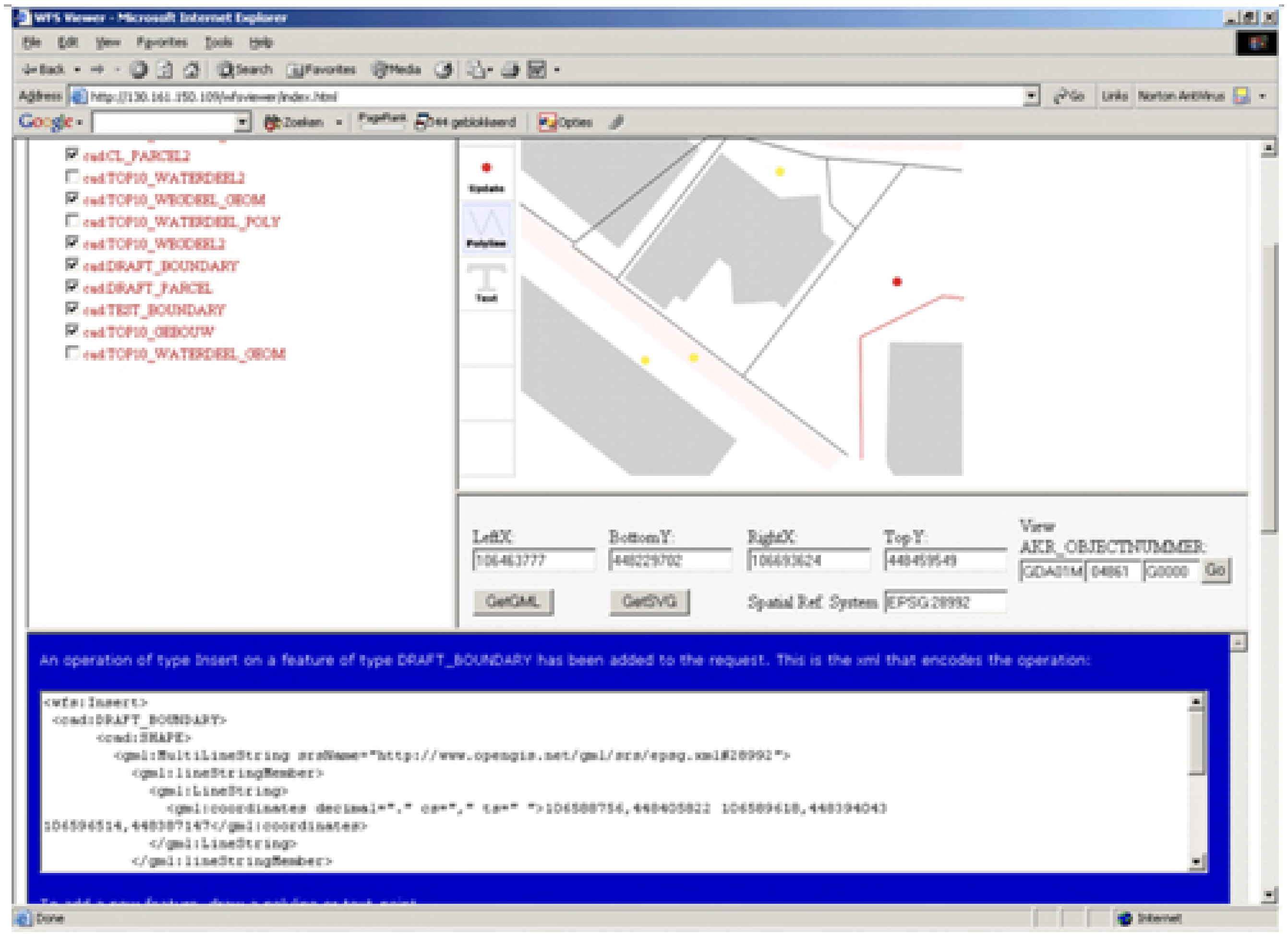


Figure 5 The server consisting of Oracle (DBMS), GeoServer (WFS) and Tomcat (webserver, Java platform)

In the Web client the data from the Dutch cadastral geo-database LKI serves as background for drafting new boundaries and parcels. With a spatial query (based on the current bounding box) also other data, like orthophotos or topographic data from other WFS or WMS services, can be added in the client as additional background information. The WFS service should make it possible for notaries to access the LKI database over the cadastral network (Internet), as illustrated in figure 4. Since the notaries do not need advanced GIS tools for complex analyses of data, a simple front-end for viewing and editing geodata would be sufficient. The client should consist of a viewer/editor and some layer that "talks" WFS, i.e. transforms operations from the viewer/editor to valid WFS requests and handles communication of requests and responses with the web feature server. The server has a data layer (consisting of an Oracle database with cadastral data), which is accessed by the WFS layer and "responds" according to the WFS protocol.

The open source WFS server GeoServer (http://geoserver.sourceforge.net) has been used in the case study. GeoServer is a full implementation of the WFS specification of the OpenGIS Consortium. GeoServer can be configured as a Transactional web feature server on several data formats, including Oracle Spatial (the cadastral data used in this case study is stored in an Oracle database). We used GeoServer in combination with Tomcat as webserver and (Java) servlet engine, as in Figure 5. Because existing (open source/ freeware) WFS clients are either not fully compliant with the specification or not Transactional, a prototype client oriented at developers needed to be developed. The developed client uses SVG (2004) for visualization. SVG can be generated by transforming the GML output stream with an XSLT (2004) stylesheet (see Figure 6). More information can be found in (Brentjes, 2004).



LeftX: 106463777
BottomY: 448229702
RightX: 106693624
TopY: 448459549
View AKR_OBJECTNUMMER: GDA01M 04861 G0000
Spatial Ref. System EPSG:28992

An operation of type Insert on a feature of type DRAFT_BOUNDARY has been added to the request. This is the xml that encodes the operation:

```
<wfs:Insert>
 <cad:DRAFT_BOUNDARY>
      <cad:SHAPE>
        <gml:MultiLineString srsName="http://www.opengis.net/gml/srs/epsg.xml#28992">
          <gml:lineStringMember>
            <gml:LineString>
              <gml:coordinates decimal="." cs="," ts=" ">106588756,448405822 106589618,448394043
106596514,448387147</gml:coordinates>
            </gml:LineString>
          </gml:lineStringMember>
```

Figure 6 From SVG to GML / WFS. The white box below – in the blue frame - contains a part of the WFS-request to add to the Transaction request.

4. WFS interoperability

A basic principle for interoperable services, in this case WFS, is that any WFS client should be able to communicate with any WFS service. This means that both client and server have to comply with the OpenGIS WFS specification. For interoperability of the server it is important that the web feature server can provide valid GML. Validating the GML from GeoServer against its schemas showed that the produced GML is valid. All kinds of WFS compliant transaction requests have been constructed with the developed client. Transactions consisting of insert, update and delete operations, in random order and different quantities have been sent to the server. These transactions were all processed successfully.

Besides the case study client, other WFS clients have been tested. GeoMedia Viewer does not accept restrictions (see Figure 7) that are defined in the XML-schemas of feature types. If these parts of the schema are removed, then GeoMedia Viewer accepts the features, although there still is a problem with handling the Dutch spatial reference system (EPSG:28992). The fact that features from an 'unknown' data source can be retrieved, visualized and queried by just any WFS compliant Web client shows the power of interoperable Web Feature Services. Unfortunately, no other Transactional clients than the developed client could be tested, simply because no open source or freeware Transactional WFS clients were available.

```
…
 <xs:element name="CLASSIF" nillable="false"
                  minOccurs="1" maxOccurs="1">
    <xs:simpleType>
        <xs:restriction base="xs:integer">
            <xs:maxLength value="11"/>
        </xs:restriction>
    </xs:simpleType>
 </xs:element>
…
```

Figure 7 Example of a part of an XML-schema. Restrictions are defined by the -element

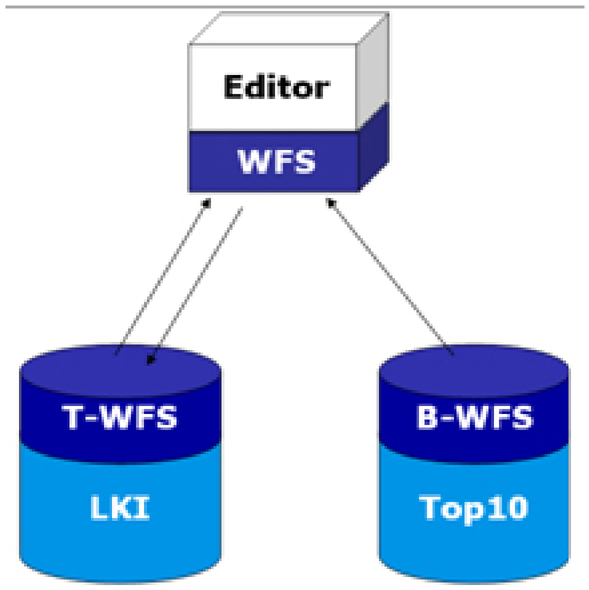


Figure 8 Two (independent) Web Feature Services accessed by one WFS client

Besides the GeoServer WFS server also other server software is being tested, i.e. the Ionic RedSpiderWeb WFS. Also this test was successful after solving some minor issues. Figure 8 shows the situation in which our WFS client accesses both the GeoServer WFS and RedSpiderWeb WFS. In this case the cadastral data is available at GeoServer WFS server and some background topographic data is available at the RedSpiderWeb WFS server.

5. Evaluation of WFS

The unique aspect of Transaction WFS is that, now for the first time ever, it is possible to edit data in a heterogeneous distributed environment. However, though 'simple' editing does go well (sketching new parcel boundaries without maintaining the topology of the final and approved parcels), we would like to share a number of observations related to editing in more complex real world situations, such as handling topology (van Oosterom, 1997). These observations are usually related to current limitations of the WFS protocol, but once identified they may be solved in future versions of WFS:

1. *GML3/Topology:* WFS is not yet advanced enough to support transactions on complicated geographic data sets based on GML3 (e.g. with topology). Incorporating GML 3.0 in WFS would allow WFS to deal with topology, temporal aspects and default styling.

2. *New object identifiers:* When the client creates new features, these should be assigned unique identifiers. The client needs identifiers in order to be able to refer from one object to another object; e.g. a boundary may refer to the parcel on the left and on the right side. Of course, the client can generate locally unique identifiers (for new boundaries and parcels), but there is no guarantee that these are also unique at the server (in a multi-user environment). Two possible solutions are:
    - A new WFS request type is added 'GetNewIds' in which a client can request one or a range of new unique object-identifiers.
    - The WFS request type 'Transaction' has built in functionality to translate the local id's of new features into global id's and send back a report to the client with these translation details. Note that this translation also the referring local id's (foreign keys in RDBMS terminology) should be replaced by global id's; e.g. in the boundary feature not only the local id of the boundary itself should be replaced by a global id, but also the left and right references (local id's) to parcels should be replaced by global id's in the left and right references.
3. *Multiple attribute identifiers:* The current WFS (and Filter encoding) specification assumes one attribute to be the feature identifier. In real world applications the identifiers may be composed of several attributes; e.g. in case of a cadastral parcel this could be: municipality, section parcel number and time (parcel version). A solution for this would be that the WFS request 'DescribeFeatureType' (perhaps a better name for this request would be 'GetFeatureTypeDescription') returns in its description the definition of identifiers (possibly composed of multiple attributes); both for the features own identifiers (primary key) and identifiers of other features types used in references within this feature (foreign key).
4. *Area Locking:* Though it is possible to lock all features overlapping with a specified lock area via the WFS request type 'LockFeature' (and specifying the actual area via the filter encoding), this does not truly locks an area. Other users may for example insert new features (which could overlap with some of the locked features). Certain application may require a true locking of the area (and not only locking the features within the area).
5. *True (atomic) transactions:* To a client it is unknown (and up to a certain extend it does not matter) whether a server is based on a DBMS or on files to manage the data. One important aspect of an atomic (DBMS) transaction is that either all actions within one transaction (inserts, deletes, updates) succeed or none of the actions succeed. This in order to bring the system from one consistent state into the next consistent state (which may require several basic actions at the level of insert/delete/update). However, WFS has defined the status of partial successful (in addition to completely successful and fail), to allow for web feature servers that don't support atomic transactions (file based) to give reports which actions succeeded and which failed. Note that the web feature server cannot advertise whether transactions are dealt with as atomic transaction (thus as one entity) or not. This should be enhanced in the future versions of the WFS request 'GetCapabilities'.
6. *Error reporting:* This should be enhanced and common error messages - especially in the case of edit operations - should be standardized, in order to give useful feedback to end-users.
7. *Integrity constraints in transactions:* Validation of (changes in) features should prevent that a data set will contain invalid features, that is, features that violate topological rules or other spatial or non-spatial restrictions. The WFS specification defines some

operations and mechanisms that can be used for validation of single features. It is not so easy however to enforce integrity constraints that concern combinations of features (as in the case of topologically structured data or certain rules implied by business logic).

8. *Clients defining new feature types:* Currently, transaction WFS allows clients to add, delete and update feature instances of feature types known at the server. In DBMS terms this is related to the Data Manipulation Language (DML) operations. One could image situations in which a client wants to define a new feature type (from scratch or based on inheritance from an existing feature type). Again in DBMS terms, this would then be related to the Data Definition Language (DDL) operations. Therefore, the future WFS specification should consider including a new request: 'DefineFeatureType', through which a client can submit a GML schema defining a new feature Type.
9. *Transferring constraint knowledge to client:* The server may check certain integrity constraints after the client posts a transaction and as a result the transaction may fail. However, for the client it is unknown what the constraints are (except for the conditions implied by the GML schema defining the individual feature types). It may be quite frustrating for a client trying to update data and getting back (unexpected) errors. In fact, dealing with all kinds of constraints in a data model is a generic problem. Using a Model Driven Approach (MDA) constraints have to be modeled first (similar to object classes, attributes and relationships), for instance in UML (Unified Modeling Language) class diagrams and OCL (Object Constraint Language). The actual implementation is a derivative of the model. For more information on MDA, UML and OCL, visit the website of the Object Management Group (http://www.omg.org/, [49]). One interesting question is: is it possible (and meaningful) to translate constraints in the data model to constraints related to the structure of valid transactions (e.g. a parcel split always implies at least deleting one old parcel, inserting a new boundary and two new parcels).

6. Conclusions and future work

The case study presented in this paper shows that the retrieval and combination of geo-data from multiple, heterogeneous data sources in one Web client is relatively easy with OpenGIS WFS services. One reason is that the WFS specification clearly describes the requests and responses that a WFS service should support. This way it is possible to 'decouple' client and server and e.g. build an application-specific Web client that still can communicate with (open source or commercial) server software developed by others. Another reason is that WFS uses standard web technologies as HTTP and XML (GML and WFS-requests/responses). Common web technologies can be used, like Servlets/Java Server Pages for application logic and SVG for cartographic visualization.

Not only web-based clients, but also thicker clients (like GIS software for analysis or viewers) can use data from WFS servers. This makes exchanging and sharing geo-data a lot easier: whether the data is stored in a local file system or in a remote database has become (almost) transparent to the end-user.

Besides retrieving, also editing of data in an interoperable web environment has been tested in this case study. In the case study 'notary drafts cadastral parcel' a relative simple edit procedure has successfully been realized in our prototype. Based on these experiences and requirements of more complex cadastral editing, an evaluation of the WFS protocol has been given, together with a number of suggestions for future extensions/improvements of the WFS protocol. For developing fully functional Internet-GIS based applications, application logic can be divided between the web feature server, the client and other (mediating) services like application services. An interesting research topic is how and where to the check integrity constraints (and other application logic) in WFS based distributed systems.

References

- Berg, C. van den, Tuijnman, F., Vijlbrief, T., Meijer, C., Oosterom, P. van, Uitermark, H., Multi-server internet GIS: Standardization and practical experiences. In Goodchild, M.F., M.J. Egenhofer, R. Fegeas, and C.A. Kottman, editors (1999), Interoperating Geographic Information Systems, Boston, USA, (International Conference and Workshop on Interoperating Geographic Information Systems, Santa Barbara, California, USA, December 3-4 and 5-6, 1997) pages 365-378.
- Brentjens, T., OpenGIS web feature services for editing cadastral data. Analysis and practical experiences, MSc thesis, TU Delft, section GIS technology, May 2004.
- OGC, 2001, Vretanos, P. (editor), Filter Encoding Implementation Specification, version 1.0, Reference number: OGC 02-059, OpenGIS Consortium Inc., USA.
- OGC, 2002, Vretanos, P. (editor), Web Feature Service Implementation Specification, version 1.0, Reference number: OGC 02-058, OpenGIS Consortium Inc., USA.
- OGC, 2003, Cox, S., Daisey, P., Lake, R., Portele, C., Whiteside, A. (editors), OpenGIS Geography Markup Language (GML) Implementation Specification, version 3. Reference number: OGC 02-023r4, OpenGIS Consortium Inc., USA.
- Oosterom, P. van, 1997, Maintaining consistent topology including historical data in a large spatial database. In Chrisman, N. (Ed.), Proceedings of Auto-Carto 13, Bethesda: ACSM & ASPRS, pp. 327-336.
- SVG, Scalable Vector Graphics, http://www.w3.org/TR/SVG/intro.html, last visited 13 April 2004.
- XML Schema, http://www.w3.org/XML/Schema, last visited 21 April 2004.
- XSLT, XML Stylesheet transformations, http://www.w3.org/TR/xslt, last visited 16 June 2004.