EVALUATING THE OPENGIS WEB FEATURE SERVICES PROTOCOL WITH THE CASE STUDY 'DISTRIBUTED CADASTRAL TRANSACTIONS'[1]

Thijs BRENTJENS (1), Marian de Vries (2), Wilko Quak (2), Tom Vijlbrief (3) and Peter van Oosterom (2)

(1) Geodan B.V.
The Netherlands
(2) GIS Technology, Research Institute for Housing, Urban and Mobility Studies/
Faculty of Technology, Policy and Management, Delft University of Technology
The Netherlands
(3) Kadaster, Apeldoorn
The Netherlands

ABSTRACT

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 'read only' retrieval. Today there are many Internet servers, often conforming to the standard OpenGIS Web Map Service (WMS) protocol, mainly delivering (raster) images, which can be viewed by clients. There are also a few 'closed' Internet GIS applications that allow editing of features by clients based on proprietary communication protocols. In practice, this means that server and 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 Service (WFS) protocol, it is now possible, for the first time, to realize Internet-based geo-information processing environments, where clients can not only view but also edit (update) data stored at multiple servers, and where client products of one vendor can be combined with server products of another vendor. This will be illustrated via a case study 'notary drafts cadastral parcel boundary', a relatively simple distributed editing prototype. The use case and the actual test scenarios are based on procedures and data model of the Dutch cadastre. The WFS protocol will be analysed for more advanced edit scenarios and a number of improvements of the WFS protocol are suggested.

[1] This research has been carried out within the framework of the Spearhead ´Sustainable Urban Areas´ (SUA) of the Delft University of Technology.

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 editing of geo-data via Web 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 for future extensions or 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 in the OpenGIS WFS specification: *Basic WFS* (for retrieving geographic 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 of the WFS specification (version 1.0) is based on GML 2. This has some disadvantages, as now also GML 3 is available, with more functionality (e.g. topology, metadata, 3D primitives, temporal model). A second argument for fast incorporation of GML 3 in WFS is the fact that GML 3 will become an ISO standard, which is not the case with GML 2.

A 'Basic WFS' service implements the *GetCapabilities*, *DescribeFeatureType* and *GetFeature* requests. A client can request an XML encoded capabilities document (containing the names of feature types that can be accessed via this service, the spatial reference system(s) and spatial extent of the data, plus information about the operations that are supported) by sending the *GetCapabilities* request to the WFS service. The function of the *DescribeFeatureType* request is to generate an XML Schema document (*XSD, 2004*) with a description of the data structure of the feature types serviced by that WFS service. The *GetFeature* request allows for the retrieval of feature instances (with all or part of their attributes) and uses filters from the OpenGIS Filter Encoding specification (*OGC, 2001*) to constrain the data to retrieve. For 'simple' selection requests also keyword-value pairs can be used, 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 (edit) geographic features as well, i.e. insert, update, and delete geographic features (see Figure 2 for an insert operation). 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 *GetFeature WithLock* request. When the transaction has been completed, a Web Feature Service will generate an XML response 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 invoke a feature locking mechanism to ensure consistency by preventing editing on these features by other users at that same moment. 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) request 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 in case of an error) is 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 BOUNDARY

As an example of cadastral transactions, one could think of a notary who sketches a new parcel boundary because a parcel is split into two new parcels as the result of a property transaction. It is important to understand that in the Dutch context the exact boundaries of the resulting parcels are determined *after* the legal (administrative) property transaction. Cadastral surveyors ask the parties involved in the transaction to 'point' at the boundaries in the actual terrain ("aanwijs vragen" in Dutch). This means that it is sufficient for a notary to create a sketch without exact measurements and with additional comments in the text specifying the transfer. If there is a disagreement then a judge will decide (*not* the Dutch Cadastre). 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 cadastral surveyor surveys the exact boundaries.

Other applications of a Cadastral WFS could be: the selection of a parcel that is transferred as a whole ('would you like to transfer *this* parcel'); pointing to parcels to be joined; or requesting information (administrative data of a map) concerning specific parcels. These applications do however not require explicit editing of cadastral data and could also be implemented with an OpenGIS Web Map Service (WMS).
In summary the functional requirements for the interface between the Cadastral WFS server and the (notary) client consist of the following requests:

1. **`Transfer`** of whole parcel: *input is a parcel number*; result is a GML dataset with the requested parcel (if this parcel exists) and some context in the form of neighbor parcels, buildings and annotations (street names, house numbers, parcel numbers). The client action, after inspecting that this is indeed the parcel to be transferred, is to send back an OK signal to the Cadastral WFS server.
2. **`Merge`** of two or more parcels: *input is a set of parcel numbers*, which are all to be joined into one parcel; result is again a GML dataset with all the parcels and some context information (if the parcels are existing and indeed all neighbors, otherwise an error is raised). The client action is after inspection to send back an OK signal.
3. **`Split`** of a parcel: *input is a parcel number*; result is a GML dataset with the parcel and context (if parcel does exit). The client action is to add one (or more) parcel

boundary/ies within the area of the parcel to be split and further every implied part of the parcel is uniquely labeled with a text string (usually just one letter 'A', 'B',..).

4. **`Get`** a reference cadastral map (not intended for the transfer, merge or split of parcels): *input is a bounding box*; result is a GML dataset of a cadastral map covering the requested area. The server expects no further client action.

Only the third operation requires real geographic editing and therefore this operation is further investigated and described in this paper. The fourth operation does not even have a direct link to the transfer, merge or split of parcel, but could be a convenient building block for all kinds of cadastral Web applications.

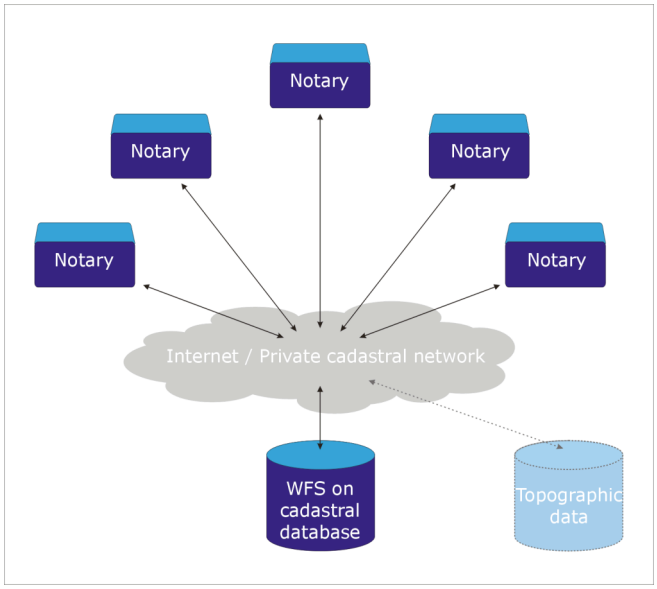

Figure 4: Web Feature Service for notaries to draft new parcels and boundaries in cadastral database optionally using topographic data as background.

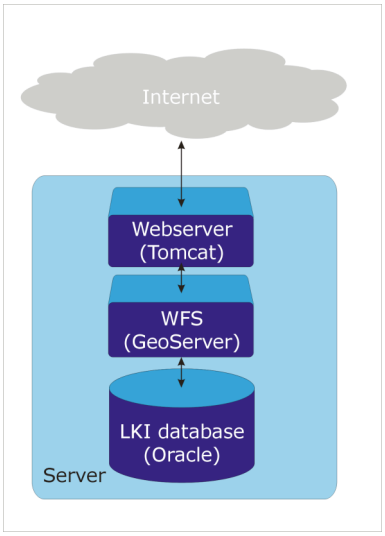

Figure 5: The server consisting of Oracle (DBMS), GeoServer (WFS) and Tomcat (Web server, Java platform).

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 data analysis, 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 Web server and Java servlet engine (see Figure 5).
Because existing (open source/ freeware) WFS clients are either not fully compliant with the WFS specification or not Transactional, a prototype client needed to be developed. The developed client uses SVG (*SVG, 2004*) for the visualization of the GML output of the WFS service, and also for temporarily 'storing' the geometric edits of the user (the parcel boundaries and label points drawn 'on screen'. SVG can be generated by transforming the GML output stream with an XSLT (*XSLT, 2004*) stylesheet (see Figure 6).
More information about the prototype client can be found in (*Brentjens, 2004*).

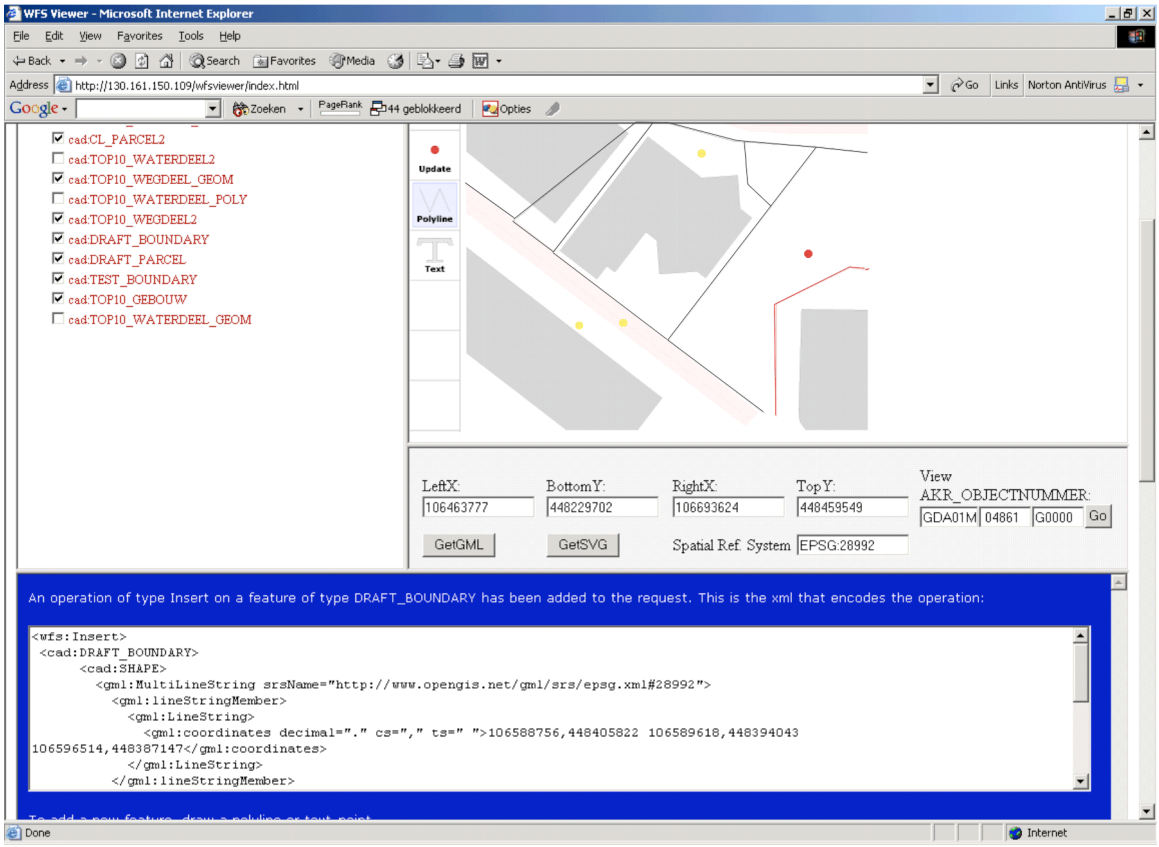

Figure 6: From SVG to GML/WFS. The white box below – in the blue frame - contains a part of the WFS-request to add to the Transaction request.

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 it 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.

We used the prototype client to access two other WFS services, i.e. a beta WFS service of Intergraph, and the RedSpiderWeb WFS service with topographic data mentioned in the previous Section.
Besides the case study Web client, we tested another WFS client: Intergraph's GeoMedia Viewer. The GeoMediaViewer did not accept the 'simpleType' restrictions (see Figure 7) in the XML schema definitions of the feature types. If these parts of the schema are removed, then GeoMediaViewer accepts the GML output of the GeoServer WFS service, although there appears to be 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: XML Schema fragment. Domain value restrictions are defined by the <xs:restriction>-element.

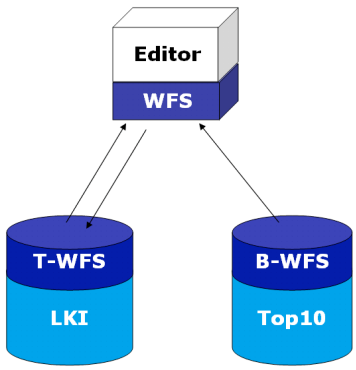

Figure 8: Two (independent) Web Feature Services accessed by one WFS client.

Figure 8 shows the situation in which our prototype WFS-T client accesses both the GeoServer WFS and the RedSpiderWeb WFS. The cadastral data is available at the GeoServer WFS server and some background topographic data (1:10.000) is available at the RedSpiderWeb WFS server.

5. Evaluation: WFS-T Functionality

The unique aspect of Transaction WFS is that, now for the first time ever, it is possible to edit data in a multi-vendor, 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 mostly related to current limitations of the WFS protocol, but once identified they may be solved in future versions of WFS:

1. *Complex data:* Current WFS implementations are based on GML version 2. GML 2 is not yet advanced enough to support transactions on more complex geographic data, e.g. data sets with topology or with 3D information. Using GML 3.1 in WFS would allow WFS services to deal with topology, more complex geometry types (BSpline, circle, 3D solids) and with 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 in 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 (composite keys):* 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 feature's own identifiers (primary key, candidate keys) and for identifiers of other features used in references from this feature (foreign keys).

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 lock an area. Other users may still insert new features (which could overlap with some of the locked features). Certain applications may require a true locking of the area (and not only locking the features within the area).

5. *True (composite) transactions*: To a client it is unknown (and up to a certain extent it does not matter) whether a server is based on a DBMS or on files to manage the data. One important aspect of a composite (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 'partially successful' (in addition to 'completely successful' and 'fail'), to allow for Web Feature servers that don't support composite transactions (file based) to give reports about which actions succeeded and which failed. Note that the Web Feature server cannot advertise

whether transactions are really dealt with as one 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 constraints that follow from rules implied by business logic).

8. *Transferring constraint knowledge to the 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 application 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 see http://www.omg.org.) 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).

9. *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 imagine 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.

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 in setting up the client/server architecture, like (Java) servlets and Java Server Pages (JSP) for application logic and SVG for cartographic visualization.

Not only Web clients, but also thicker clients (like GIS software that has more possibilities for analyzing the data) 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.

In the research presented in this paper the focus was on the editing of geo-data in an interoperable Web environment. In the case study 'notary drafts cadastral parcel' a relatively 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 Service, the client and other (mediating) services like application services. An interesting research topic is how and where to the check integrity constraints (and other business rules) in WFS based distributed systems.

REFERENCES

Berg, C. van den, F. Tuijnman, T. Vijlbrief, C. Meijer, P. van Oosterom and H. Uitermark (1997): Multi-server Internet GIS: Standardization and practical experiences. In Goodchild, M.F., M.J. Egenhofer, R. Fegeas, and C.A. Kottman (eds.), Interoperating Geographic Information Systems, Boston: Kluwer Acad. Publ., 1999 (International Conference and Workshop on Interoperating Geographic Information Systems, Santa Barbara, Ca., USA, December 1997) pp. 365-378.

Brentjens, T. (2004): 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. (ed.), Filter Encoding Implementation Specification, version 1.0, Reference number: OGC 02-059, OpenGIS Consortium Inc.

OGC (2002): Vretanos, P. (ed.), Web Feature Service Implementation Specification, version 1.0, Reference number: OGC 02-058, OpenGIS Consortium Inc.

OGC (2003): Cox, S., P. Daisey, R. Lake, C. Portele, A. Whiteside (eds.), OpenGIS Geography Markup Language (GML) Implementation Specification, version 3. Reference number: OGC 02-023r4, OpenGIS Consortium Inc.

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 (2004): Scalable Vector Graphics, http://www.w3.org/TR/SVG/intro.html, last visited 13 April 2004.

XSD (2004): XML Schema, http://www.w3.org/XML/Schema, last visited 21 April 2004.

XSLT (2004): eXtensible Stylesheet Language for Transformations, http://www.w3.org/TR/xslt, last visited 16 June 2004.

CVS OF THE AUTHORS

Thijs Brentjens (born in 1978 in Amersfoort, the Netherlands) studied Geodetic Engineering at Delft University of Technology from 1997 to 2004. In May 2004 he got his Master of Science degree, after specializing in GIS in the field of OpenGIS Web Feature Services. From 1998 until 2004 he has been working as a database and web developer for The DataFactory/Overtocht.nl, a travelling organisation. From Octobre 2004 he works for the Dutch company Geodan as GIS Software Engineer.

Marian de Vries is scientific researcher at the section GIS Technology of Delft University of Technology (the Netherlands). Focus of her research is on Web service/client architectures that are based on interoperable software components and open (data exchange) standards. She was co-developer of the GML prototype of the new TOP10NL product of the Dutch Topographic Service. At the moment she is involved in a number of geo-information integration projects for large data providers in the Netherlands.

Wilko Quak is scientific researcher at the section GIS Technology of Delft University of Technology (the Netherlands). Focus of his research is on geo-DBMS technology components and open (data exchange) standards. He was co-developer of the GML prototype of the new TOP10NL product of the Dutch Topographic Service. At the moment he is involved in a number of geo-information integration projects for large data providers in the Netherlands.

Tom Vijlbrief graduated in Computer Science at the University of Amsterdam (the Netherlands). From 1986 until 1996 he worked at the TNO Institute for Human Factors Research, doing research in the field of Man-Machine Interfaces and GIS systems. After two years with a software company, he now works since 1998 for the Dutch Cadastre as software architect, with special focus on GIS, Middleware and Software Development (architectures, tools, languages).

Prof.dr.ir. Peter van Oosterom obtained a MSc in Technical Computer Science in 1985 from Delft University of Technology. In 1990 he received a PhD from Leiden University for this thesis "Reactive Data Structures for GIS" (updated version published by Oxford University Press, 1993).
From 1985 until 1995 he worked at the TNO-FEL laboratory in The Hague, the Netherlands as a computer scientist. From 1995 until 2000 he was senior information manager at the Netherlands' Kadaster, were he was involved in the renewal of the cadastral (geographic) database.
Currently Van Oosterom is head of the section 'GIS Technology' at the Delft University of Technology, OTB. His main research themes are spatial database management systems, GIS architectures, generalization, spatial analysis, querying and presentation, Internet/interoperable GIS and cadastral applications.

CO-ORDINATES

Thijs Brentjens
Geodan B.V.
President Kennedylaan 1
1079 MB Amsterdam
The Netherlands
Tel. +31 20 5711 311
Fax +31 20 5711 333
E-mail brentjens1@zonnet.nl
Website www.geodan.nl

Marian de Vries, Wilko Quak and Peter van Oosterom
GIS Technology – Research Institute for Housing, Urban and Mobility Studies
Faculty of Technology, Policy and Management
Delft University of Technology
Jaffalaan 9
2628 BX Delft
The Netherlands

Tel.	+31-15-278 4268
	+31-15-278 3756
	+31-15-278 6950
Fax	+31-15-278 2745
E-mail	M.d.Vries@otb.tudelft.nl
	w.quak@otb.tudelft.nl
	oosterom@otb.tudelft.nl
Website	www.otb.tudelft.nl
	www.gdmc.nl

Tom Vijlbrief
Kadaster
P.O. Box 9046
7300 GH Apeldoorn
The Netherlands

Tel.	+31-55-528 5175
Fax	+31-55-528 5606
E-mail	tom.vijlbrief@kadaster.nl
Website	www.kadaster.nl