GeoSciML Cookbook

How to serve a GeoSciML version 2 Web Feature Service (WFS) using Open Source Software

Contents

1 INTRODUCTION

1.1 The purpose of this cookbook

The GeoSciML V2 XML schema is a standards-based data format that provides a framework for application-neutral encoding of geoscience thematic data and related spatial data. At present the scope is delimited by the information generally shown on geological maps, along with boreholes and field observations.
This document is written to assist organisations wishing to make use of the GeoSciML data exchange standard by serving a Web Feature Service (WFS) to the WWW. This will include Level 2 participants in OneGeology who will be serving GeoSciML as a WFS.
This cookbook is based on experience gained using free and Open Source software to serve GeoSciML WFS' and includes links to a OneGeology zip file containing a template kit containing extra Open Source software Cocoon (http://cocoon.apache.org/) and the deegree (http://www.deegree.org/) WFS and the GIN (http://www.gw-info.net) mediator required to implement the WFS' described in chapters 2 and 3 below.
There are many other software options available to serve a GeoSciML WFS (each with different functional capabilities), some based on variants of what is decribed here such as an XSLT approach to using deegree, some based on alternative Open Source software like Geoserver (which will be documented here from approx. June 2009) and some based on commercial software such as Snowflake software's Go-Publisher, ESRI ArcIMS and ArcServer and finally some geological surveys have developed their own direct WFS interfaces to their backend ORACLE databases using programming languages like Java.
This cookbook will be continually improved and expanded as experience and time permits but its aim today is to help those interested in serving a GeoSciML WFS using Open Source software that has been trialled during the development of GeoSciML V2.

1.2 Who should be using this cookbook?

This cookbook is highly technical and is targeted at those within geological surveys who are tasked with setting up web services from a web server, working closely with geoscience informatics experts who need to have read and understood the sister cookbook to this one first – the GeoSciML Cookbook 'How To Map Data to GeoSciML Version 2' (found at http://www.geosciml.org/geosciml/2.0/cookbook/GeoSciML_Data_CookBook_V2.pdf).
OneGeology participants who have used the OneGeology Cookbook 1 ('How To serve a OneGeology Level 1 Conformant WMS using MapServer', from http://www.onegeology.org/misc/downloads.html) based on a MapServer template kit supplied for it may want to take the simplest route to build the WFS on top of that already installed MapServer software, and they should then follow chapter 2 below and ignore chapter 3 for now. Those not involved with OneGeology or those who wish to install a WFS that may serve more than the very simple data generally available from a OneGeology WFS' should consider starting from scratch with the installation of Cocoon with deegree opensource software detailed in Chapter 3 (Chapter 2 should not be followed but it may be useful to read it for extra background).

2 ADDING A SIMPLE WFS SERVICE TO AN EXISTING ONEGEOLOGY LEVEL 1 MAPSERVER BASED WMS SERVICE

This chapter assumes that you have set up a OneGeology Level 1 conformant WMS using MapServer following the instructions in OneGeology Cookbook 1 ("How To serve a OneGeology Level 1 Conformant WMS using MapServer" from http://www.onegeology.org/misc/downloads.html) using a vector GIS shapefile, rather than image data source.

We will take advantage of the fact that setting up the WMS with MapServer also gives us a simple functional WFS at the same time. We will use some middleware developed by the Geological Survey of Canada (GSC) called the GIN Mediator to transform this simple WFS into one serving true GeoSciML. The mediator is part of a research project and is still being further developed by GSC. Here we will only discuss delivering very simple data into minimally valid GeoSciML. GeoSciML Cookbook 'How To Map Data to GeoSciML Version 2' (found at http://www.geosciml.org/geosciml/2.0/cookbook/GeoSciML_Data_CookBook_V2.pdf) discusses in more detail how to map more complex data to more complex GeoSciML nonetheless it is a requisite that anyone intending to serve any form of GeoSciML V2 as a WFS needs to have read and understood that cookbook first before trying to serve any geological data in a GeoSciML conforming WFS..

We will assume you are able to install a Java Development Kit and the Tomcat servlet container on your server yourself.

Whilst it is possible to run the MapServer service using the Tomcat web service, this cookbook assumes an initial start point of the base install of the WMS described in the OneGeology WMS Cookbook 1 (i.e. running in the absence of Tomcat). You should therefore set up your Tomcat service to run on a different port to the standard web service (80), unless you are running another web server or service; the standard port for Tomcat (8080) will probably suffice.

We have tested the system with Sun JDK 1.5 in combination with Apache Tomcat 5.5, and with Sun JDK 1.6.0 in combination with Apache Tomcat 6.0, later versions and other servlet containers may also work.

2.1 Modifications to your MapServer configuration

You may need to make the following modifications to your existing MapServer Level 1 WMS service. Your underlying shapefiles will need a data field that is suitable to be

used as the gml:id attribute of certain GeoSciML elements. The gml:id attribute is a unique identifier for any feature within your data, and is of type "ID" ~ a non case-sensitive string beginning with a letter and followed by any combination of letters, numbers, periods, or hyphens. The "fid" field in the shapefile is therefore not suitable, but it can be used to generate one.
In the template example shapefiles we have added a field called "ID" (using ArcToolbox > Data Management Tools > Fields > Add Field) and then calculated our unique values (using ArcToolbox > Data Management Tools > Fields > Calculate Field), with the expression

```
"id." & [FID]
```

The MapServer map file that you created for the Level 1 WMS will need two kinds of modification. First, the `WMS_TITLE` field needs changing to `OWS_TITLE,` so that its value can be used by the WFS as well as the WMS. Secondly you will need to add a line to identify the gml:id field in the shapefile using `GML_FEATUREID`, within each `LAYER` section such as:

```
GML_FEATUREID "ID"
# where ="ID"= is replaced by the name of the field described above which you
wish to use for gml:id attributes
```

You can test that the MapServer WFS is working by using the following URL request templates:

GML 2 (default)

```
http://[yourserver.org]/cgi-
bin/[your_map_service_name]/wms?service=WFS&version=1.0.0&request=GetCapabilitie&
```

GML 3 (by specifying it using the outputformat parameter)

```
http://[yourserver.org]/cgi-
bin/[your_map_service_name]/wms?service=WFS&outputformat=GML3&version=1.0.0&reque
st=GetCapabilities&
```

Example:
If you have the BGS_Bedrock_and_Superficial_Geology example service set up and were browsing from the server itself, the following URL would return a valid result.

http://127.0.0.1/cgi-bin/BGS_Bedrock_and_Superficial_Geology/wms?service=WFS&outputformat=GML3&version=1.0.0&request=GetCapabilities&

Note these GetCapabilities requests (above) are only used to test that the MapServer WFS is working. We will use a different application (and therefore different URL) for our live WFS.
We are now going to set up the GIN mediator to sit between WFS clients and the above MapServer simple WFS to provide a new GeoSciML V2.0 compliant WFS that can be registered with the OneGeology portal.

2.2 Installing the GIN mediator and creating a WFS profile

Download the template web application (52 Mbytes in size) from the OneGeology FTP site (ftp://ftp.bgs.ac.uk/pubload/OneGeology/GINMediator_v1.1.2.zip or later version if present).
If your Tomcat is installed in `$TOMCAT_HOME` then copy the template application to `$TOMCAT_HOME\webapps\cocoon`

Run Tomcat and make sure you can see the Apache Cocoon welcome page at http://yourserver.org:port/cocoon

Try browsing to http://yourserver.org(:port)/cocoon/geosciml/page/index.html

If you have Tomcat Apache and Tomcat HTTP all running on your local machine then this page should display, and you should be able to access the 'Underlying MapServer WFS' links. If not you will need to change the example query links to something with the correct server hostname in e.g.

```
http://yourserver.org/cocoon/geosciml/wfs/oneg_gbr_bls?service=WFS&version=1.1.0&request=GetFeature&typename=gsml:MappedFeature&
```

Note here that the string "oneg_gbr_bls" is the name of what we will call a "WFS profile". This identifies a particular WFS which can respond to certain defined queries with particular feature types with a defined level of detail in the property values returned. The template application comes with five WFS profiles (oneg_ gbr_bls, oneg_ gbr_blt, oneg_ gbr_ba, oneg_ gbr_sls and oneg_ gbr_slt) corresponding to 5 layers in the OneGeology WMS cookbook BGS example. (There is another WFS profile

MappedFeature used as an example in Chapter 3 of this cookbook.) For OneGeology services we suggest using a WFS profile name like "oneg_ [3 letter iso country code]_ [layer]" where layer is a string specifying which of your WMS layers this service is providing associated data for.

In $TOMCAT_HOME\webapps\cocoon\geosciml\sitemap.xmap

Change:

```
<map:pipelines>
   <map:component-configurations>=
      <global-variables>
         <!-- WFS url of the mediator-->
         <wfs-host>http://localhost:8080/cocoon/geosciml/wfs</wfs-host>
      </global-variables>
   </map:component-configurations>
...
```

To:

```
<map:pipelines>
   <map:component-configurations>
      <global-variables>
         <!-- WFS url of the mediator-->
         <wfs-host>http://[yourserver.org:port]/cocoon/geosciml/wfs</wfs-host>
      </global-variables>
   </map:component-configurations>
```

The above change just alters the host name reported in the mediated WFS GetCapabilities response so that WFS clients not on the server machine will be able to extract the correct address for further queries from it.
Once you have confirmed that you can access index.html, you will need to set up a new WFS profile for the mediated services to run on top of your own MapServer simple WFS. This is done in four places. For each WFS profile you will set up a GetCapabilities response file in `$TOMCAT_HOME\webapps\cocoon\geosciml\capabilities`, a configuration file in `$TOMCAT_HOME\webapps\cocoon\geosciml\config` that specifies the location of the underlying simple WFS, a mapping file in a subdirectory of `$TOMCAT_HOME\webapps\cocoon\geosciml\mapping` that will specify how incoming queries

specified in the GeoSciML model can be transformed to queries in the underlying simple WFS, and a handling transformation file (using XSLT) in a subdirectory of `$TOMCAT_HOME\webapps\cocoon\geosciml\handler` that transforms the output of the simple WFS into GeoSciML.

Capabilities file

To define the GetCapabilities response for your WFS profile you should create a file called [profile_name].xml in `$TOMCAT_HOME\webapps\cocoon\geosciml\capabilities`. If you know that the response should be the same, except for the service URLs, for several of your WFS profiles you can edit the single file `wfs-default.xml` to provide that GetCapabilities response, and only specify a profile named file for those profiles which need a different response.
This file is used by client applications to discover what features are available from your service and details (metadata) about those features. This document also contains information about the organisation serving the data and the contact persons responsible for the data and/or the service. You can change most of the text in the ServiceProvider section. You will note here and there are strings that contain parameters in square brackets.

EG:

```
<ows:OnlineResource xlink:href="[host]" xlink:type="simple"/>
```

Don't alter these entries. These parameters will be replaced by the current url of your service (the wfs-host value you just changed in the previous step).
The other part you might want to edit is further down in the file where you can find a list of Features. The template application example files contain a definition for gsml:MappedFeature which is what we expect most OneGeology registered services to provide. The human readable details and the bounding box should be altered to fit your specific data.

```
<wfs:FeatureType>
   <wfs:Name>gsml:MappedFeature</wfs:Name>
   <wfs:Title>Mapped Feature</wfs:Title>
   <wfs:Abstract>Mapped features specified by Geologic Units</wfs:Abstract>
   <ows:Keywords xmlns:ows="http://www.opengis.net/ows">
      <ows:Keyword>GeoSciML OneGeology</ows:Keyword>
   </ows:Keywords>
   <wfs:DefaultSRS>EPSG:4326</wfs:DefaultSRS>
```

```
        <wfs:OtherSRS>EPSG:4326</wfs:OtherSRS>
        <wfs:OutputFormats>
           <wfs:Format>text/xml; subtype=gml/3.1.1</wfs:Format>
        </wfs:OutputFormats>
        <ows:WGS84BoundingBox xmlns:ows="http://www.opengis.net/ows">
           <ows:LowerCorner>-180 180</ows:LowerCorner>
           <ows:UpperCorner>--90 90</ows:UpperCorner>
        </ows:WGS84BoundingBox>
     </wfs:FeatureType>
```

You will need to change the Title, Abstract, LowerCorner, UpperCorner, SRS and keywords elements as appropriate to describe your service, but don't* alter the Name tag value.

WFS service profile configuration files

$TOMCAT_HOME\webapps\cocoon\geosciml\config\oneg_*.xml

In the `$TOMCAT_HOME\webapps\cocoon\geosciml\config` directory you will find 5 example BGS configuration files named oneg_*.xml There is one for each of the 5 layers of the BGS OneGeology WMS service and they have been set up to provide 5 different WFS services, one per WMS layer. For your OneGeology services it may be appropriate to do something similar or it may be that a single WFS would provide appropriate data for all your WMS layers. The content of all five files is the same, however, and you can copy any one of them to a new configuration file for your WFS. You should then edit the service-url attribute of the <proc:datasource> element to point to your OneGeology WMS service.
If you have followed OneGeology Cookbook 1 this will mean changing the path element "BGS_Bedrock_and_Superficial_Geology" to your map service name. The host name in the URL will also need changing if your MapServer OneGeology service is on a different machine from Tomcat.

Change:

```
<proc:datasources>
  <proc:datasource name="oneg"
    service-url="http://127.0.0.1/cgi-
bin/BGS_Bedrock_and_Superficial_Geology/wms?outputformat=GML3&amp;">
    <proc:handledFeature name="gsml:MappedFeature" handler="MappedFeature"/>
    <!--<proc:processor name="debug"/>-->
```

```
    <proc:processor name="wfs"/>
  </proc:datasource>
</proc:datasources>
```

To:

```
<proc:datasources>
  <proc:datasource name="oneg"
    service-url="http://[yourserver.org]/cgi-
bin/[yourmapservice]wms?outputformat=GML3&amp;">
    <proc:handledFeature name="gsml:MappedFeature" handler="MappedFeature"/>
    <!--<proc:processor name="debug"/>-->
    <proc:processor name="wfs"/>
  </proc:datasource>
</proc:datasources>
```

Note
The service URL has "outputformat=GML3" as an extra parameter on the end. This is because the MapServer WFS created by following Cookbook 1 produces GML2 unless explicitly changed and GeoSciML uses GML3 for its geometry properties.

Handler and Mapping

These are obviously related as they are both based on how the fields in the underlying WFS map to fields in the GeoSciML model. For more information on how to map your data to GeoSciML see the GeoSciML cookbook 'How to map data to GeoSciML Version 2'. Here, we are going to show some very simple examples. All the BGS examples except Bedrock Lithostratigrapy (BLS) deal with only a single data field in the underlying simple WFS which we map to the name and description of the GeoSciML GeologicUnits in an output GeoSciML serialization. This data field is the field which determines the symbology of the corresponding WMS layer (CLASSITEM). We expect everyone who has set up a shapefile based OneGeology WMS should be able to achieve at least this level of mapping.

Uniform Resource Names (URNs)

GeoSciML uses URNs as a method to uniquely identify any data provided. URNs (by definition) are intended to serve as persistent, location-independent, resource identifiers; to satisfy the simplest implementation of a valid GeoSciML output, (as discussed in this chapter), URNs do not need to be resolvable. You may define your

own URNs to identify your data resource (see URN Syntax), or you may use the CGI pattern. Using the CGI pattern, all that is required is to change the {authority} and the name of the {vocabulary} as appropriate; so for example using the below codeSpace string, you would need to change **BGS** to be an identifier for your organization, and **Lithology** to be the name of (or type of) your vocabulary (or dictionary) for the data you are providing.

codeSpace="urn:cgi:classifierScheme:BGS:Lithology:description"

If the WFS was supplying structural geology data provided by the Afghanistan Geological Survey, the above URN could conceivably be modified to codeSpace="urn:cgi:classifierScheme:AGS:StructuralFeatures:description"

For a list of registered vocabularies see:
http://appgeosciml.brgm.fr/GeoSciMLWeb/middleware.jsp

For a list of registered authorities see:
https://www.seegrid.csiro.au/twiki/bin/view/CGIModel/CGIAuthorityRegister

OneGeology participants should register their organization details through the CGI Naming Authority
https://www.seegrid.csiro.au/twiki/bin/view/CGIModel/CGINamingAuthority

Mapping the incoming request to the underlying service

$TOMCAT_HOME\webapps\cocoon\geosciml\mapping\oneg_* folders and
$TOMCAT_HOME\webapps\cocoon\geosciml\mapping\oneg_*\oneg\MappedFeature.map

You can start by copying, for example, the `$TOMCAT_HOME\webapps\cocoon\geosciml\mapping\oneg_gbr_blt` folder to a folder with your WFS profile name; if you look inside this folder and then inside the oneg folder you will find MappedFeature.map, an XML document.

The parts you will need to edit for your case are the mappedType attribute value of the gsml MappedFeature element (which represents your WMS LAYER name), and the values in the gml:description and gml:name elements (which corresponds to the GML_INCLUDE_ITEMS field).

Change:

```
<?xml version="1.0" encoding="UTF-8"?>
<gsml:MappedFeature xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
 xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:ms="http://mapserver.gis.umn.edu/mapserver" xmlns:proc="urn:x-lcnp:proc"
 mappedType="GBR_BGS_625k_BLT">
 <proc:namespaces>
  <proc:namespace name="gsml">urn:cgi:xmlns:CGI:GeoSciML:2.0</proc:namespace>
  <proc:namespace name="gml">http://www.opengis.net/gml</proc:namespace>
  <proc:namespace name="xlink">http://www.w3.org/1999/xlink</proc:namespace>
  <proc:namespace
name="ms">http://mapserver.gis.umn.edu/mapserver</proc:namespace>
 </proc:namespaces>
 <proc:targetVersion>1.0.0</proc:targetVersion>
 <gsml:specification>
  <gsml:GeologicUnit>
   <gml:description>RCS_D</gml:description>
   <gml:name
codeSpace="urn:cgi:classifierScheme:BGS:Lithology:description">RCS_D</gml:name>
  </gsml:GeologicUnit>
 </gsml:specification>
 <gsml:shape>msGeometry</gsml:shape>
</gsml:MappedFeature>
```

To:

```
<?xml version="1.0" encoding="UTF-8"?>
<gsml:MappedFeature xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
 xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:ms="http://mapserver.gis.umn.edu/mapserver" xmlns:proc="urn:x-lcnp:proc"
 mappedType="[your WMS layer name]">
 <proc:namespaces>
  <proc:namespace name="gsml">urn:cgi:xmlns:CGI:GeoSciML:2.0</proc:namespace>
  <proc:namespace name="gml">http://www.opengis.net/gml</proc:namespace>
  <proc:namespace name="xlink">http://www.w3.org/1999/xlink</proc:namespace>
  <proc:namespace
name="ms">http://mapserver.gis.umn.edu/mapserver</proc:namespace>
```

```
 </proc:namespaces>
 <proc:targetVersion>1.0.0</proc:targetVersion>
 <gsml:specification>
  <gsml:GeologicUnit>
   <gml:description>[your GML_INCLUDE_ITEMS field]</gml:description>
   <gml:name codeSpace="urn:cgi:classifierScheme:[your-organization-
descriptor]:[your-vocabulary-descriptor]:description">
      [your GML_INCLUDE_ITEMS field]
   </gml:name>
  </gsml:GeologicUnit>
 </gsml:specification>
 <gsml:shape>msGeometry</gsml:shape>
</gsml:MappedFeature>
```

The mapping of the incoming queries may not be as complete as the output mapping below; the service will have some functionality but may not be able to handle queries on all properties.

Handler to transform output from the simple WFS into GeoSciML

$TOMCAT_HOME\webapps\cocoon\geosciml\handler\oneg_* folders and
$TOMCAT_HOME\webapps\cocoon\geosciml\handler\oneg_*\oneg\MappedFeature.xslt

We will create the handler which transforms the output of the MapServer WFS to valid GeoSciML. You can start by copying, for example, the $TOMCAT_HOME\webapps\cocoon\geosciml\handler\oneg_gbr_blt folder to a folder with your WFS profile name. If you look inside this folder and then inside the oneg folder you will find MappedFeature.xslt, an XML Stylesheet which converts the BGS simple WFS into a GeoSciML instance.
The parts you will need to edit for your case are the ms:GBR_BGS_625k_BLT value of the match attribute on the xsl:template element (which represents your WMS LAYER name), and the ms:RCS_D values in the gml:description and gml:name elements (which corresponds to the GML_INCLUDE_ITEMS field).

Change:

```
<xsl:template match="ms:GBR_BGS_625k_BLT">
  <gsml:MappedFeature gml:id="{@gml:id}">
    <xsl:apply-templates select="gml:boundedBy"/>
    <gsml:observationMethod>
```

```
      <gsml:CGI_TermValue>
        <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
      </gsml:CGI_TermValue>
    </gsml:observationMethod>
    <gsml:positionalAccuracy>
      <gsml:CGI_TermValue>
        <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
      </gsml:CGI_TermValue>
    </gsml:positionalAccuracy>
    <gsml:samplingFrame xlink:href="urn:cgi:feature:BGS:EarthBedrockSurface"/>
    <gsml:specification>
      <gsml:GeologicUnit gml:id="gu.{@gml:id}">
        <gml:description>
          <xsl:value-of select="ms:RCS_D"/>
        </gml:description>
        <gml:name codeSpace="urn:cgi:classifierScheme:BGS:Lithology:description">
          <xsl:value-of select="ms:RCS_D"/>
        </gml:name>
        <gsml:observationMethod>
          <gsml:CGI_TermValue>
            <gsml:value
codeSpace="urn:cgi:classifier:BGS:ObservationMethod">Summary of published
              description</gsml:value>
          </gsml:CGI_TermValue>
        </gsml:observationMethod>
        <gsml:purpose>instance</gsml:purpose>
      </gsml:GeologicUnit>
    </gsml:specification>
    <gsml:shape>
      <xsl:copy-of select="ms:msGeometry/*"/>
    </gsml:shape>
  </gsml:MappedFeature>
</xsl:template>
```

To:

```
<xsl:template match="ms:[your WMS LAYER name]">
  <gsml:MappedFeature gml:id="{@gml:id}">
    <xsl:apply-templates select="gml:boundedBy"/>
    <gsml:observationMethod>
      <gsml:CGI_TermValue>
        <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
      </gsml:CGI_TermValue>
    </gsml:observationMethod>
    <gsml:positionalAccuracy>
      <gsml:CGI_TermValue>
        <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
      </gsml:CGI_TermValue>
    </gsml:positionalAccuracy>
    <gsml:samplingFrame
      xlink:href="urn:cgi:feature:[your-organization-
descriptor]:EarthBedrockSurface"/> <!-- Or other value depending on what your
mapping surface is -->
    <gsml:specification>
      <gsml:GeologicUnit gml:id="gu.{@gml:id}">
        <gml:description>
          <xsl:value-of select="ms:[your GML_INCLUDE_ITEMS field]"/>
        </gml:description>
        <gml:name
          codeSpace="urn:cgi:classifierScheme:[your-organization-
descriptor]:[your-vocabulary-descriptor]:description">
          <xsl:value-of select="ms:[your GML_INCLUDE_ITEMS field]"/>
        </gml:name>
        <gsml:observationMethod>
          <gsml:CGI_TermValue>
            <gsml:value
              codeSpace="urn:cgi:classifier:[your-organization-
descriptor]:ObservationMethod"
              >Summary of published description</gsml:value>
          </gsml:CGI_TermValue>
        </gsml:observationMethod>
        <gsml:purpose>instance</gsml:purpose>
```

```
      </gsml:GeologicUnit>
    </gsml:specification>
    <gsml:shape>
      <xsl:copy-of select="ms:msGeometry/*"/>
    </gsml:shape>
  </gsml:MappedFeature>
</xsl:template>
```

2.3 Serving the mediated WFS on the same port as your WMS

In order to make OneGeology services as accessible as possible to people with possibly restrictive firewall policies (and also for aesthetic reasons) it is preferable to serve this from the standard web service (usually port 80). If you are already running your MapServer WMS through the Apache HTTP web server on the same machine you will not be able to make Tomcat run on port 80 as well. However, you can use the instructions below to configure Apache to proxy your Tomcat services so that you can make them available at a port 80 URL.
Edit the Apache HTTP server httpd.conf file (If your installation exactly follows that of Cookbook 1 this would be located at: c:\ms4w\Apache\conf\httpd.conf)

Change

```
#LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
#LoadModule proxy_http_module modules/mod_proxy_http.so
```

To:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule proxy_http_module modules/mod_proxy_http.so
```

and add the following directives:

```
TraceEnable off
# Configure Apache proxying to Tomcat
ProxyRequests Off
ProxyPreserveHost On
```

```
# We only want requests for the cocoon service to be sent to Tomcat
<ProxyMatch http://[^/]*/cocoon/*>
    Order deny,allow
    Allow from 127.0.0.1
</ProxyMatch>

# Include these two directives for each web application that you wish to forward
to Tomcat
ProxyPass /cocoon http://127.0.0.1:8080/cocoon/
ProxyPassReverse /cocoon http://127.0.0.1:8080/cocoon/

<Location /cocoon>
    Order allow,deny
    Allow from all
</Location>
```

Note

The regular expression used in <ProxyMatch>, is designed to work with most implementations based on the WMS configuration specified in OneGeology Cookbook 1 (How to serve a OneGeology Level 1 conformant WMS using MapServer), and may be too greedy for your purposes; you could replace it with a variant more specific to your servers, or you could substitute <Proxy> instead.

See your Apache HTTP documentation for more details

http://yourserver.org/manual/mod/mod_proxy.html#proxy

Now you need to edit the Tomcat server XML file (Eg. c:\Tomcat 6.0\conf\server.xml).

Change:

```
<Connector
  port="8080"
  protocol="HTTP/1.1"
  connectionTimeout="20000"
  redirectPort="8443" />
```

To:

```
<Connector
  port="8080"
```

```
  protocol="HTTP/1.1"
  connectionTimeout="20000"
  redirectPort="8443"
  proxyName="yourserver.org"
  proxyPort="80"
/>
```

ProxyName: is the domain name or IP of the standard (Apache HTTP Server) web service and can be omitted if you are running your Tomcat service on the same server as the http service.

Note:

- It is better to configure the proxy at this stage rather than after testing through the Tomcat service as it reduces the number of edits to the configuration files
- We have used mod_proxy here as it is included with the Apache HTTP Server binaries, but you could use other proxy modules such as mod_jk if desired.

2.4 Registering the new OneGeology Simple Web Feature Service

You should now have finished and after testing your service you should register it with the OneGeology portal by sending an email to `onegeologyportal@brgm.fr` with the URL of your WFS profile(s) (e.g. `http://[yourserver.org]/cocoon/geosciml/wfs/oneg_gbr_bls`) and noting which of your OneGeology WMS layers each WFS profile is to be associated with.

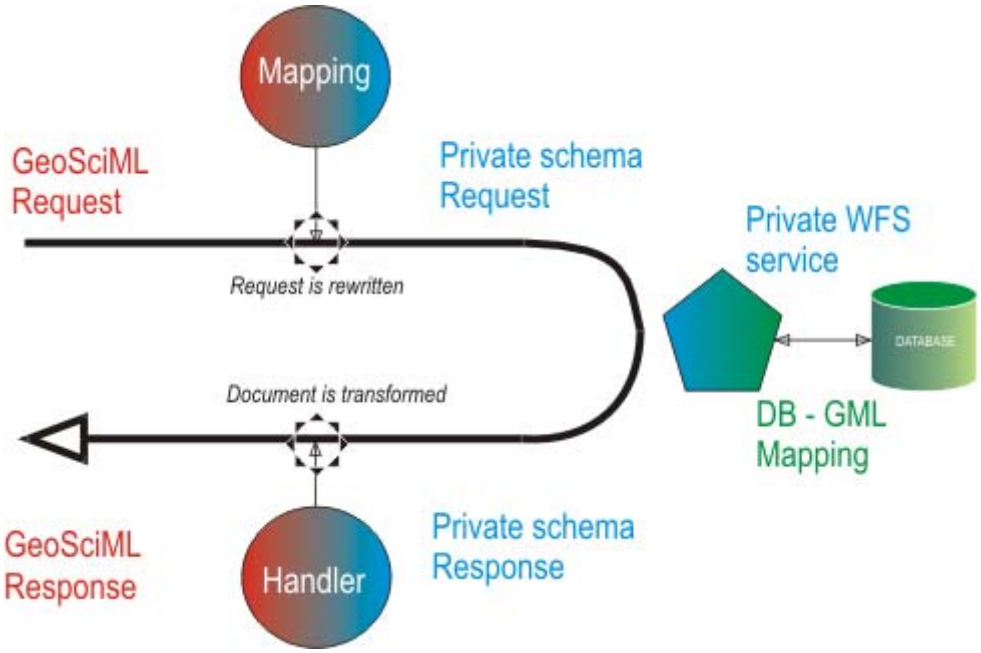
3 SERVING MAPPED GEOLOGICAL UNITS AS MAPPED FEATURES IN A WFS USING THE GIN MEDIATOR WITH DEEGREE WFS SOFTWARE

3.1 Introduction

Chapter 2 showed how to use the GIN mediator to transform a flat WFS with only one attribute into GeoSciML. This chapter will show a more complex WFS that has more data and can deliver some complexity of structure, such as having nested one-to-many properties for each top-level feature returned by the WFS. More specifically we will use the deegree WFS server to generate XML that is close, but not quite the same as the intended GeoSciML. It may be possible to use similar techniques for other WFS servers that can produce reasonably complex XML. Here we will use an embedded deegree component that is included in the Chapter 2 download. The embedded component is more tightly integrated into the Cocoon pipeline and has OGC Filter functions enabled. The steps are:

1. Create a Database to GML mapping using the internal deegree component
2. Define one or more WFS services (WFS service profiles)
3. Create a mapping for the incoming request to the intermediary layer
4. Create a handler

The architecture of the GIN mediator is illustrated in the diagram below:



The GIN mediator as configured with the template web application will further transform the final response to ensure that no duplicate features are returned. Duplicate features need to be handled to ensure the returned XML document is valid. The mediator converts any repeated features into internal (xlink:href) links to a single inline instance of the feature. For example in the BGS service described below, we have cases where the gsml:specification property of several 'gsml:MappedFeatures' is the same gsml:GeologicUnit.

3.2 Prerequisites

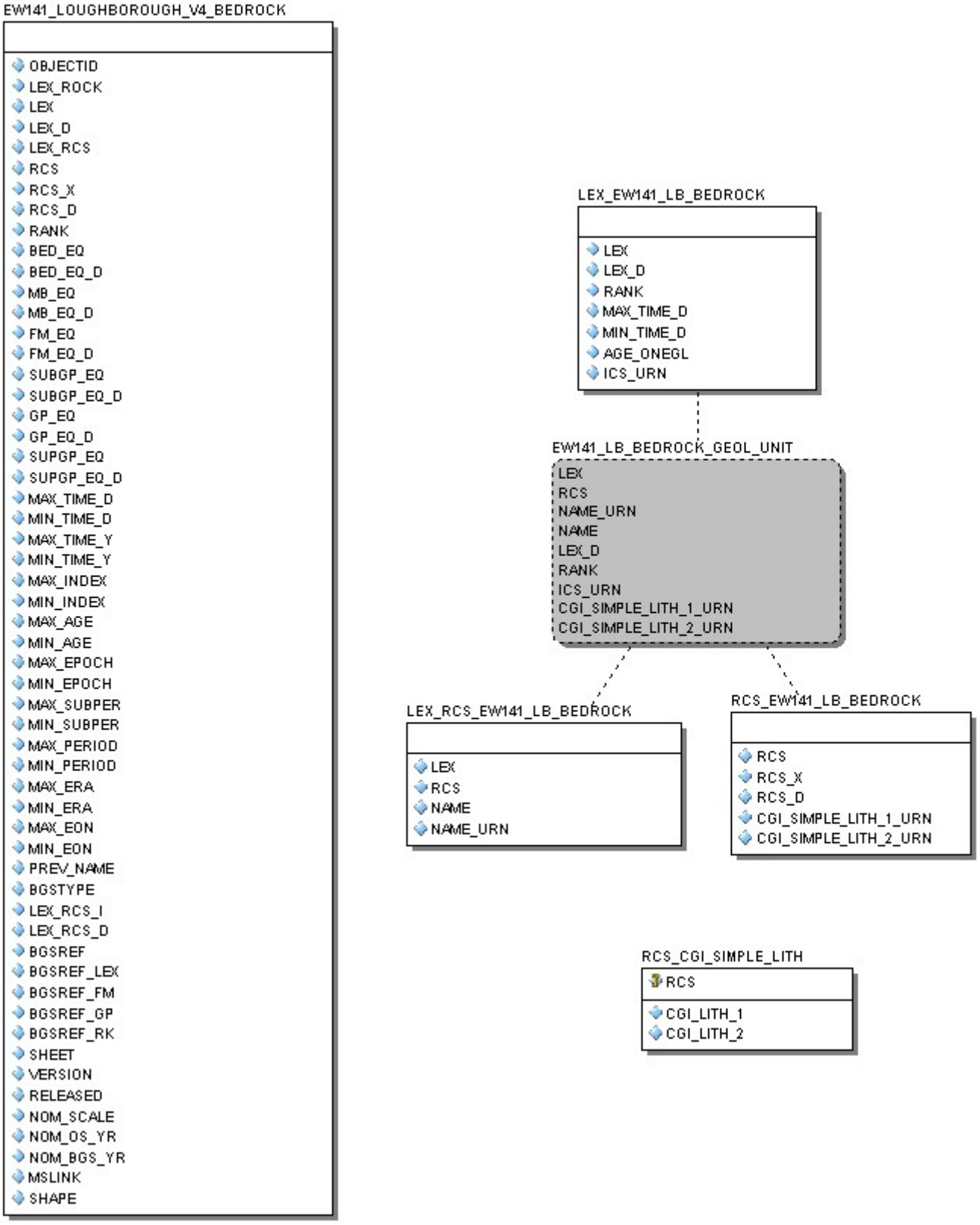
- You will need to have a good understanding of GeoSciML 2.0 and GML 3.1.1 to implement the mappings of your data structure to them.
- A multi-table database (data with some level of normalisation)
- The database backend must be supported by deegree, see the introduction section of http://download.deegree.org/deegree2.1/docs/htmldocu_wfs/deegree_wfs_documentation_en.html for the list of supported backends (note: deegree also supports non-spatial databases and provides a technique to handle geometries in those backends, so it's possible to use a non-geospatial aware backend, with some restrictions).
- The database must support database-side function execution if you want to implement queries with OGC Filter functions.
- The GIN mediator is installed (see Chapter 2)

3.3 deegree WFS data source configuration

```
$TOMCAT_HOME\webapps\cocoon\WEB-INF\conf\wfs
```

Data sources must be mapped to feature types in order to be served by the deegree WFS. There are 2 stages to this mapping, the first as described here, is a database to private GML schema mapping, and a second (as described in a later section) the GeoSciML to private schema mapping (which is a GML to GML mapping). The first (DB-Private) mapping is used by the WFS engine (the embedded deegree component in this case). The second mapping is used by the mediator trying to reconcile between the GML models (GeoSciML and your private GML structure). Obviously, the closer the private model is to GeoSciML, the less mediation is required. You should try to implement as much of this mapping as possible within the database backend, simply because database engines are optimised to manipulate data and deegree essentially generates a SQL statement from the mapping.

Database table structure for the BGS example WFS



The deegree component embedded in cocoon is essentially the same as the deegree you can find on the web. We have used an early 2.1.n version with two code changes; one, so it can serialize the results of the WFS query directly into a cocoon pipeline, and another so deegree can accept ogc:Function filters.
See http://download.deegree.org/deegree2.1/docs/htmldocu_wfs/deegree_wfs_documentation_en.html for full documentation, especially section 4. Your goal is not to create a full GeoSciML mapping, though it may be possible in simple cases, but something which is close, and preferably (to save transformation effort) with all nesting in the correct order.

3.4 Create the intermediary (private) mapping

The datasource mapping is an XSD (W3C schema file) with the deegree tags placed within application information tags (xsd:appinfo), within an annotation element. You can create faily complex mappings by joining multiple tables. The intermediary mapping uses a special namespace (http://www.iugs.org/cgi/tempInternal prefix:t) to distinguish it from the final GeoSciML output.

The complete intermediary mapping for the BGS example is shown below:

```
<xsd:schema xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:deegreewfs=http://www.deegree.org/wfs
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:t="http://www.iugs.org/cgi/temporary_internal"
  targetNamespace="http://www.iugs.org/cgi/tempInternal"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <xsd:import namespace="http://www.opengis.net/gml"
    schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/feature.xsd"/>
  <xsd:import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/geometryAggregates.xsd"
/>
<!-- ========================================================== -->
  <xsd:annotation>
    <xsd:appinfo>
      <deegreewfs:Prefix>t</deegreewfs:Prefix>
      <deegreewfs:Backend>ORACLE</deegreewfs:Backend>
```

```
    <deegreewfs:DefaultSRS>EPSG:4326</deegreewfs:DefaultSRS>
    <JDBCConnection xmlns="http://www.deegree.org/jdbc">
      <Driver>oracle.jdbc.OracleDriver</Driver>
      <Url>[jdbc connection string]</Url>
      <User>[database user]</User>
      <Password>[database password]</Password>
      <SecurityConstraints/>
      <Encoding>utf-8</Encoding>
    </JDBCConnection>
    <deegreewfs:SuppressXLinkOutput>true</deegreewfs:SuppressXLinkOutput>
  </xsd:appinfo>
</xsd:annotation>
<!-- ============================================================ -->
<xsd:element name="MappedFeature" type="t:MappedFeatureType"
  substitutionGroup="gml:_Feature">
  <xsd:annotation>
    <xsd:appinfo>
      <deegreewfs:table>OGC.EW141_LOUGHBOROUGH_V4_BEDROCK</deegreewfs:table>
      <deegreewfs:gmlId prefix="t.mf.">
        <deegreewfs:MappingField field="OBJECTID" type="INTEGER"/>
      </deegreewfs:gmlId>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>
<!-- ============================================================ -->
    <xsd:complexType name="MappedFeatureType">
      <xsd:complexContent>
        <xsd:extension base="gml:AbstractFeatureType">
          <xsd:sequence>
            <xsd:element name="gid" type="xsd:integer">
              <xsd:annotation>
                <xsd:appinfo>
                  <deegreewfs:Content>
                    <deegreewfs:MappingField field="MSLINK" type="INTEGER"/>
                  </deegreewfs:Content>
                </xsd:appinfo>
              </xsd:annotation>
            </xsd:element>
```

```
            <xsd:element name="specification" type="gml:FeaturePropertyType"
minOccurs="0" maxOccurs="unbounded">
              <xsd:annotation>
                <xsd:appinfo>
                  <deegreewfs:Content type="t:GeologicUnit">
                    <deegreewfs:Relation>
                      <deegreewfs:From>
                        <deegreewfs:MappingField field="LEX" type="VARCHAR"/>
                        <deegreewfs:MappingField field="RCS" type="VARCHAR"/>
                      </deegreewfs:From>
                      <deegreewfs:To>
                        <deegreewfs:MappingField field="LEX" type="VARCHAR"/>
                        <deegreewfs:MappingField field="RCS" type="VARCHAR"/>
                      </deegreewfs:To>
                    </deegreewfs:Relation>
                  </deegreewfs:Content>
                </xsd:appinfo>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="geom" type="gml:GeometryPropertyType">
              <xsd:annotation>
                <xsd:appinfo>
                  <deegreewfs:Content>
                    <deegreewfs:MappingField field="SHAPE" type="GEOMETRY"
srs="4326"/>
                  </deegreewfs:Content>
                </xsd:appinfo>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <!-- ========================================================== -->
    <xsd:element name="GeologicUnit" type="t:GeologicUnitType"
substitutionGroup="gml:_Feature">
      <xsd:annotation>
        <xsd:appinfo>
```

```
      <deegreewfs:table>OGC.EW141_LB_BEDROCK_GEOL_UNIT</deegreewfs:table>
      <deegreewfs:gmlId prefix="t.gu.">
        <deegreewfs:MappingField field="LEX" type="VARCHAR"/>
        <deegreewfs:MappingField field="RCS" type="VARCHAR"/>
      </deegreewfs:gmlId>
      <deegreewfs:visible>true</deegreewfs:visible>
    </xsd:appinfo>
  </xsd:annotation>
</xsd:element>
<!-- ============================================================ -->
<xsd:complexType name="GeologicUnitType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="name_urn" type="xsd:string">
          <xsd:annotation>
            <xsd:appinfo>
              <deegreewfs:Content>
                <deegreewfs:MappingField field="NAME_URN" type="VARCHAR"/>
              </deegreewfs:Content>
            </xsd:appinfo>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="descriptive_name" type="xsd:string">
          <xsd:annotation>
            <xsd:appinfo>
              <deegreewfs:Content>
                <deegreewfs:MappingField field="LEX_D" type="VARCHAR"/>
              </deegreewfs:Content>
            </xsd:appinfo>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="age" type="xsd:string">
          <xsd:annotation>
            <xsd:appinfo>
              <deegreewfs:Content>
                <deegreewfs:MappingField field="ICS_URN" type="VARCHAR"/>
              </deegreewfs:Content>
```

```
              </xsd:appinfo>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="rank" type="xsd:string">
            <xsd:annotation>
              <xsd:appinfo>
                <deegreewfs:Content>
                  <deegreewfs:MappingField field="RANK" type="VARCHAR"/>
                </deegreewfs:Content>
              </xsd:appinfo>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="lithology_1" type="xsd:string">
            <xsd:annotation>
              <xsd:appinfo>
                <deegreewfs:Content>
                  <deegreewfs:MappingField field="CGI_SIMPLE_LITH_1_URN"
type="VARCHAR"/>
                </deegreewfs:Content>
              </xsd:appinfo>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="lithology_2" type="xsd:string">
            <xsd:annotation>
              <xsd:appinfo>
                <deegreewfs:Content>
                  <deegreewfs:MappingField field="CGI_SIMPLE_LITH_2_URN"
type="VARCHAR"/>
                </deegreewfs:Content>
              </xsd:appinfo>
            </xsd:annotation>
          </xsd:element>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <!-- ========================================================== -->
</xsd:schema>
```

Let’s look at this is a bit more detail:

First we define the (persistent) connection to the back end data source, in this case ORACLE.

```
    <xsd:annotation>
      <xsd:appinfo>
        <deegreewfs:Prefix>t</deegreewfs:Prefix>
        <deegreewfs:Backend>ORACLE</deegreewfs:Backend>
        <deegreewfs:DefaultSRS>EPSG:4326</deegreewfs:DefaultSRS>
        <JDBCConnection xmlns="http://www.deegree.org/jdbc">
          <Driver>oracle.jdbc.OracleDriver</Driver>
          <Url>[jdbc connection string]</Url>
          <User>[database user]</User>
          <Password>[database password]</Password>
          <SecurityConstraints/>
          <Encoding>utf-8</Encoding>
        </JDBCConnection>
        <deegreewfs:SuppressXLinkOutput>true</deegreewfs:SuppressXLinkOutput>
      </xsd:appinfo>
    </xsd:annotation>
```

Next we define the first element **t:MappedFeature** that we wish to create from our database and link it to the appropriate table **OGC.EW141_LOUGHBOROUGH_V4_BEDROCK** :

```
    <xsd:element name="MappedFeature" type="t:MappedFeatureType"
substitutionGroup="gml:_Feature">
      <xsd:annotation>
        <xsd:appinfo>
          <deegreewfs:table>OGC.EW141_LOUGHBOROUGH_V4_BEDROCK</deegreewfs:table>
          <deegreewfs:gmlId prefix="t.mf.">
            <deegreewfs:MappingField field="OBJECTID" type="INTEGER"/>
          </deegreewfs:gmlId>
        </xsd:appinfo>
      </xsd:annotation>
    </xsd:element>
```

Our element **t:MappedFeature** is of type **t:MappedFeatureType** and we now describe how the fields in our database map to this FeatureType, starting with our first sub element **t:gid** which is a straight mapping with **OGC.EW141_LOUGHBOROUGH_V4_BEDROCK.MSLINK** .

```
<xsd:complexType name="MappedFeatureType">
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element name="gid" type="xsd:integer">
          <xsd:annotation>
            <xsd:appinfo>
              <deegreewfs:Content>
                <deegreewfs:MappingField field="MSLINK" type="INTEGER"/>
              </deegreewfs:Content>
            </xsd:appinfo>
          </xsd:annotation>
        </xsd:element>
```

Our next element **t:specification** has a more complex relation to the database, defined by the compound foreign key relationship between **OGC.EW141_LOUGHBOROUGH_V4_BEDROCK.LEX** and **OGC.EW141_LOUGHBOROUGH_V4_BEDROCK.RCS** with **OGC.EW141_LB_BEDROCK_GEOL_UNIT.LEX** and **OGC.EW141_LB_BEDROCK_GEOL_UNIT.RCS:**

```
        <xsd:element name="specification" type="gml:FeaturePropertyType"
minOccurs="0" maxOccurs="unbounded">
          <xsd:annotation>
            <xsd:appinfo>
              <deegreewfs:Content type="t:GeologicUnit">
                <deegreewfs:Relation>
                  <deegreewfs:From>
                    <deegreewfs:MappingField field="LEX" type="VARCHAR"/>
                    <deegreewfs:MappingField field="RCS" type="VARCHAR"/>
                  </deegreewfs:From>
                  <deegreewfs:To>
                    <deegreewfs:MappingField field="LEX" type="VARCHAR"/>
```

```
                    <deegreewfs:MappingField field="RCS" type="VARCHAR"/>
                  </deegreewfs:To>
                </deegreewfs:Relation>
              </deegreewfs:Content>
            </xsd:appinfo>
          </xsd:annotation>
        </xsd:element>
```

Note, you need to look further down the schema to the definition of element **t:GeologicUnit** to understand the relationship with the **OGC.EW141_LB_BEDROCK_GEOL_UNIT** view.

```
    <xsd:element name="GeologicUnit" type="t:GeologicUnitType"
substitutionGroup="gml:_Feature">
      <xsd:annotation>
        <xsd:appinfo>
          <deegreewfs:table>OGC.EW141_LB_BEDROCK_GEOL_UNIT</deegreewfs:table>
          <deegreewfs:gmlId prefix="t.gu.">
            <deegreewfs:MappingField field="LEX" type="VARCHAR"/>
            <deegreewfs:MappingField field="RCS" type="VARCHAR"/>
          </deegreewfs:gmlId>
          <deegreewfs:visible>true</deegreewfs:visible>
        </xsd:appinfo>
      </xsd:annotation>
    </xsd:element>
```

Finally we deal with the geometry of the feature:

```
      <xsd:element name="geom" type="gml:GeometryPropertyType">
        <xsd:annotation>
          <xsd:appinfo>
            <deegreewfs:Content>
              <deegreewfs:MappingField field="SHAPE" type="GEOMETRY" srs="4326"/>
            </deegreewfs:Content>
          </xsd:appinfo>
        </xsd:annotation>
      </xsd:element>
```

Remember that when this mapping is processed by deegree it will produce an incomplete version of the desired GeoSciML output; it must however return all the required data and attributes required by the final GeoSciML output.
Once you have created the mapping, double-check everything - the way deegree errors are reported by GIN mediator is not very helpful (most of the time, you'll get an empty file). Most of the time, a single typo (typing error) will prevent deegree from working.
Once the mapping is completed, you must restart Tomcat.

3.5 Using the GIN mediator to debug the intermediary (private) mapping

```
$TOMCAT_HOME\webapps\cocoon\geosciml\document\
```

The GIN mediator has a special pipeline to test deegree, using any arbitrarily named wfs:Query xml instance.

For example:

Prepare a document like this one and save it as bgs.xml in the geosciml document folder.

```
<?xml version="1.0" encoding="utf-8"?>
    <wfs:GetFeature
        xmlns:wfs="http://www.opengis.net/wfs"
        xmlns:ogc="http://www.opengis.net/ogc"
        xmlns:t="http://www.iugs.org/cgi/tempInternal"
        maxFeatures="5" service="WFS"
        version="1.1.0" xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0">
        <wfs:Query typeName="t:MappedFeature">
        </wfs:Query>
    </wfs:GetFeature>
```

To send this document to deegree, invoke this pipeline

```
http://yourserver.org(:port)/cocoon/geosciml/debug/deegree/bgs
```

The last part of the url must match the name of the file you just created in the document directory (less the file suffix), so here /bgs matches /document/bgs.xml)

If everything works fine, you should see a document similar to this one:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:FeatureCollection xmlns:t="http://www.iugs.org/cgi/tempInternal"
    xmlns:wfs="http://www.opengis.net/wfs"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    numberOfFeatures="5"
    xsi:schemaLocation="http://www.iugs.org/cgi/tempInternal
http://localhost:8080/deegree-
wfs/services?SERVICE=WFS&amp;VERSION=1.1.0&amp;REQUEST=DescribeFeatureType&amp;TY
PENAME=t:MappedFeature&amp;NAMESPACE=xmlns(t=http://www.iugs.org/cgi/tempInternal
) http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.1.0/wfs.xsd">
    <gml:boundedBy>
        <gml:Envelope srsName="EPSG:4326">
            <gml:pos srsDimension="2">-1.53358714293239
52.7447745036496</gml:pos>
            <gml:pos srsDimension="2">-1.25631067876913
52.8204090327274</gml:pos>
        </gml:Envelope>
    </gml:boundedBy>
    <gml:featureMember>
        <t:MappedFeature gml:id="t.mf.53">
            <gml:boundedBy>
                <gml:Envelope srsName="EPSG:4326">
                    <gml:pos srsDimension="2">-1.25709663323346
52.7447745036496</gml:pos>
                    <gml:pos srsDimension="2">-1.25631067876913
52.7454577448837</gml:pos>
                </gml:Envelope>
            </gml:boundedBy>
            <t:gid>595</t:gid>
            <t:specification>
                <t:GeologicUnit gml:id="t.gu.BCTDATUF">

<t:name_urn>urn:cgi:feature:BGS:StratigraphicLexicon:BCT</t:name_urn>
<t:descriptive_name>BEACON TUFF MEMBER</t:descriptive_name>
```

```
<t:age>urn:cgi:classifier:ICS:StratChart:2008:Ediacaran</t:age>
                    <t:rank>MEMBER</t:rank>
                </t:GeologicUnit>
            </t:specification>
            <t:geom>
                <gml:Polygon srsName="EPSG:4326">
                    <gml:outerBoundaryIs>
                        <gml:LinearRing>
                            <gml:coordinates cs="," decimal="." ts=""
                                >-1.25670372306689,52.7454577448837
                                -1.25689780981031,52.7453690631345
                                -1.25703356437367,52.7452260771792
                                -1.25709663323346,52.7450017282326
                                -1.25705403117502,52.7448935842841
                                -1.25692209903121,52.7448118503595
                                -1.25670053155272,52.7447745036496
                                -1.25650721154648,52.7448182412529
                                -1.25634229163705,52.7449340744674
                                 -1.25631067876913,52.745050743079
                                -1.2563380081549,52.7451857606649
                                -1.25640962123418,52.7453300456306
                                -1.25655575427669,52.7454478281049
                                -1.25670372306689,52.7454577448837
                            </gml:coordinates>
                        </gml:LinearRing>
                    </gml:outerBoundaryIs>
                </gml:Polygon>
            </t:geom>
        </t:MappedFeature>
    </gml:featureMember>
 ...
</wfs:FeatureCollection>
```

Note

`$TOMCAT_HOME\logs`

`$TOMCAT_HOME\webapps\cocoon\WEB-INF\logs`

If it doesn't work, check your Tomcat logs, especially the stdout_[date].log which will record errors with the database connection.
Be aware that deegree will autoload all xsd files in the `$TOMCAT_HOME\webapps\cocoon\WEB-INF\conf\wfs\featuretypes\` directory, even if they are not referenced in the capability document. If any data source referenced in any xsd file is not available, deegree will fail.
Be aware too that deegree needs to write to your system temporary files folder and expects a sub-folder with the same name as your Tomcat version. If this folder doesn't exist degree will fail.
You may wish to set up more than one WFS service to serve different variants (WFS service profiles) of your data depending on purpose or user. To distinguish between different versions of the same features within these profiles you could use a different namespace on your intermediary (private) mapping, such as t1:MappedFeature and t2:MappedFeature, or you could use the same namespace and with a different name, such as bgs:MappedFeature1, bgs:MappedFeature2. There is no semantic difference between these two methodologies, it's purely a matter of personal preference, and the mediator will convert either into proper GeoSciML.

3.6 Create a WFS service profile

`$TOMCAT_HOME\webapps\cocoon\geosciml\capabilities\`
`$TOMCAT_HOME\webapps\cocoon\geosciml\config\`

As mentioned described in Chapter 2 we create a WFS profile by editing files in four places. The example template application contains an example profile following the Chapter 3 procedure called "MappedFeature". The GetCapabilities response should be configured just as in Chapter 2. The WFS profile configuration file in the config directory will look like the MappedFeature.xml example below with "deegree" specified as the processor rather than "wfs" which was used in Chapter 2. We then rename the **proc:datasource** attribute name to one appropriate to BGS, and add appropriate FeatureTypes that we wish to be handled.

Our resultant configuration file looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<proc:config xmlns:proc="urn:x-lcnp:proc">
  <!-- LCNP GeoSciML service wrapper configuration -->
  <proc:defaultPrefixMapping>
    <proc:ns prefix="gsml">urn:cgi:xmlns:CGI:GeoSciML:2.0</proc:ns>
```

```
  </proc:defaultPrefixMapping>
  <proc:maxFeatures>1000</proc:maxFeatures>
  <proc:defaultGeometries>
    <!-- for BBOX queries in HTTP/GET mode, there is no way to tell the service
which geometry is to be filtered.  If not defined here, defaultGeometry is used.
The geometry property must be provided with the target Feature as the context
node -->
    <proc:featureGeometry
typeName="gsml:MappedFeature">gsml:shape</proc:featureGeometry>
    <proc:defaultGeometry>gsml:shape</proc:defaultGeometry>
  </proc:defaultGeometries>
  <proc:datasources>
    <proc:datasource name="bgs">
      <proc:handledFeature name="gsml:MappedFeature" handler="MappedFeature"/>
      <!--<proc:processor name="debug"/>-->
      <proc:processor name="deegree"/>
    </proc:datasource>
  </proc:datasources>
</proc:config>
```

The datasources/datasource part is the interesting part.

```
  <proc:datasources>
    <proc:datasource name="bgs">
      <proc:handledFeature name="gsml:MappedFeature" handler="MappedFeature"/>
      <!--<proc:processor name="debug"/>-->
      <proc:processor name="deegree"/>
    </proc:datasource>
  </proc:datasources>
```

This tells the GIN mediator that when it receives a request for gsml:MappedFeature in the MappedFeature profile, it can be handled by the **bgs** datasource and the name of the handler is MappedFeature.

Three elements are important here:

- bgs (the datasource name)
- MappedFeature (the profile name)

- MappedFeature (the handler)

These three components are important for the next step.

3.7 Mapping the incoming request via the intermediary layer

```
$TOMCAT_HOME\webapps\cocoon\geosciml\mapping\
```

We now need to create a mapping between our intermediary gml output and GeoSciML for the GIN mediator to do the transformation to our desired output.
The mapping file **looks** like an instance document, but it is not. The goals of the mapping file are to

- provide some key information such as target namespaces, etc..
- provide a way to resolve GeoSciML property names (expressed as XPath)

We need to create a directory structure under our geosciml/mapping that looks like this:

- geosciml <- already exists
    - mapping <- already exists
        - MappedFeature <- must create.. (the name of the WFS service profile)
            - bgs <- must create.. (the name of the proc:datasource)

In the datasource directory (bgs), you must create a mapping file named **MappedFeature.map:**

```
<?xml version="1.0" encoding="UTF-8"?>
<gsml:MappedFeature xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:proc="urn:x-lcnp:proc"
xmlns:t="http://www.iugs.org/cgi/tempInternal" mappedType="t:MappedFeature">
 <proc:namespaces>
  <proc:namespace name="gsml">urn:cgi:xmlns:CGI:GeoSciML:2.0</proc:namespace>
  <proc:namespace name="gml">http://www.opengis.net/gml</proc:namespace>
  <proc:namespace name="xlink">http://www.w3.org/1999/xlink</proc:namespace>
  <proc:namespace name="t">http://www.iugs.org/cgi/tempInternal</proc:namespace>
 </proc:namespaces>
 <proc:targetVersion>1.1.0</proc:targetVersion>
 <gml:name>name</gml:name>
```

```
 <gsml:specification>
  <gsml:GeologicUnit>
   <gml:name codeSpace="http://www.cgi-
iugs.org/uri">t:specification/t:GeologicUnit/t:name_urn</gml:name>
   <gml:name
codeSpace="urn:cgi:classifierScheme:BGS:StratigraphicLexicon:description">t:speci
fication/t:GeologicUnit/t:descriptive_name</gml:name>
   <gsml:preferredAge>
    <gsml:GeologicEvent>
     <gsml:eventAge>
      <gsml:CGI_TermValue>
       <gsml:value
codeSpace="urn:cgi:classifierScheme:ICS:StratChart:2008">t:specification/t:Geolog
icUnit/t:age</gsml:value>
      </gsml:CGI_TermValue>
     </gsml:eventAge>
    </gsml:GeologicEvent>
   </gsml:preferredAge>
   <gsml:rank
codeSpace="urn:cgi:classifierScheme:BGS:Rank">t:specification/t:GeologicUnit/t:ra
nk</gsml:rank>
   <gsml:composition>
    <gsml:CompositionPart>
     <gsml:lithology xlink:href="t:specification/t:GeologicUnit/t:lithology_1"/>
     <gsml:lithology xlink:href="t:specification/t:GeologicUnit/t:lithology_2"/>
     <gsml:lithology>
      <gsml:ControlledConcept>
       <gsml:identifier codeSpace="http://www.cgi-
iugs.org/uri">t:specification/t:GeologicUnit/t:lithology_1</gsml:identifier>
       <gsml:identifier codeSpace="http://www.cgi-
iugs.org/uri">t:specification/t:GeologicUnit/t:lithology_2</gsml:identifier>
      </gsml:ControlledConcept>
     </gsml:lithology>
    </gsml:CompositionPart>
   </gsml:composition>
  </gsml:GeologicUnit>
 </gsml:specification>
 <gsml:shape>t:geom</gsml:shape>
```

```
</gsml:MappedFeature>
```

This file will be used by the GIN mediator to map the GeoSciML to the private schema properties.

Important things to check:

- be sure you have mappedType="t:MappedFeature" property filled in the root tag.
- be sure that all relevant namespaces are listed in the proc:namespaces section
- be sure that all the relevant properties are mapped for this profile.

The order of the alternate properties here is not important, as this is just a mapping file.

3.8 Using the debug processor to check your mapping is correct

A special processor in the GIN mediator allows you to check your mapping, by sending an arbitrary file in the `$TOMCAT_HOME\webapps\cocoon\geosciml\document\` back to itself. To enable this, you must alter your WFS service profile configuration file to tell the mediator to use the 'debug' processor instead of the deegree processor. The debug processor will just perform the mapping and return the rewritten query.

First we open `MappedFeature`.xml and replace the processor in the datasource from 'deegree' to 'debug'.

From this:

```
<proc:datasources>
  <proc:datasource name="bgs">
    <proc:handledFeature name="gsml:MappedFeature" handler="MappedFeature"/>
    <!--<proc:processor name="debug"/>-->
    <proc:processor name="deegree"/>
  </proc:datasource>
</proc:datasources>
```

To this:

```
<proc:datasources>
  <proc:datasource name="bgs">
```

```
      <proc:handledFeature name="gsml:MappedFeature" handler="MappedFeature"/>
      <proc:processor name="debug"/>
      <!--<proc:processor name="deegree"/>-->
    </proc:datasource>
  </proc:datasources>
```

Now we will use a couple of queries (cb-example1.xml and cb-example2.xml) as the files to send to the debug processor.

example cb-example1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<wfs:GetFeature
 xmlns:wfs="http://www.opengis.net/wfs"
 xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:gml="http://www.opengis.net/gml"
 xmlns:proc="urn:x-lcnp:proc"
 maxFeatures="10" service="WFS"
 traverseXlinkDepth = "20"
 version="1.1.0" xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0">
   <wfs:Query typeName="gsml:MappedFeature">
   </wfs:Query>
</wfs:GetFeature>
```

To send this document to deegree, invoke this pipeline:
`http://yourserver.org:8080/cocoon/geosciml/wfs/MappedFeature?DEBUG=true&SRC=cb-example1`

Where 'MappedFeature' is the name of our WFS service profile and SRC=cb-example1 matches the path to our test query document (document/cb-example1.xml).
Returns:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
    xmlns:src="http://apache.org/cocoon/source/1.0"
    xmlns:include="http://apache.org/cocoon/include/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```
    xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:h="http://apache.org/cocoon/request/2.0"
    xmlns:proc="urn:x-lcnp:proc" xmlns:fn="http://www.w3.org/2005/xpath-
functions"
    xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
maxFeatures="10"
    service="WFS" traverseXlinkDepth="20" version="1.1.0">
    <wfs:Query xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
        xmlns:t="http://www.iugs.org/cgi/tempInternal"
typeName="t:MappedFeature"/>
</wfs:GetFeature>
```

example cb-example2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
xmlns:ogc="http://www.opengis.net/ogc"
  maxFeatures="5" service="WFS" version="1.1.0"
xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0">
  <wfs:Query typeName="gsml:MappedFeature">
    <ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
      <ogc:PropertyIsEqualTo>

<ogc:PropertyName>gsml:specification/gsml:GeologicUnit/gsml:preferredAge/gsml:Geo
logicEvent/gsml:eventAge/gsml:CGI_TermValue/gsml:value</ogc:PropertyName>
        <ogc:Literal>urn:cgi:classifier:ICS:StratChart:2008:Visean</ogc:Literal>
      </ogc:PropertyIsEqualTo>
    </ogc:Filter>
  </wfs:Query>
</wfs:GetFeature>
```

Returns:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
    xmlns:src="http://apache.org/cocoon/source/1.0"
    xmlns:include="http://apache.org/cocoon/include/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```
    xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:h="http://apache.org/cocoon/request/2.0"
    xmlns:proc="urn:x-lcnp:proc" xmlns:fn="http://www.w3.org/2005/xpath-
functions"
    xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
maxFeatures="5"
    service="WFS" version="1.1.0">
    <wfs:Query xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
        xmlns:t="http://www.iugs.org/cgi/tempInternal"
typeName="t:MappedFeature">
        <ogc:Filter>
            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>t:specification/t:GeologicUnit/t:age</ogc:PropertyName>

<ogc:Literal>urn:cgi:classifier:ICS:StratChart:2008:Visean</ogc:Literal>
            </ogc:PropertyIsEqualTo>
        </ogc:Filter>
    </wfs:Query>
</wfs:GetFeature>
```

Note the change in XPaths in the <ogc:PropertyName> element from **gsml:specification/gsml:GeologicUnit/gsml:preferredAge/gsml:GeologicEvent/gsml:eventAge/gsml:CGI_TermValue/gsml:value** to **t:specification/t:GeologicUnit/t:age** - the translation of the GeoSciML query to the location of that property in your private schema.

To test with the deegree processor revert your setting in the WFS service profile (MappedFeature.xml) configuration file.

example cb-example1.xml

Invoked using:
`http://yourserver.org:8080/cocoon/geosciml/wfs/MappedFeature?DEBUG=true&SRC=cb-example1`

Now returns:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:FeatureCollection xmlns:wfs="http://www.opengis.net/wfs"
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:h="http://apache.org/cocoon/request/2.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:proc="urn:x-
lcnp:proc"
  xmlns:fn=http://www.w3.org/2005/xpath-functions
xmlns:gml="http://www.opengis.net/gml"
xmlns:t="http://www.iugs.org/cgi/tempInternal"
xmlns:ogc="http://www.opengis.net/ogc">
  <gml:featureMember>
    <t:MappedFeature gml:id="t.mf.53">
       <gml:boundedBy>
          <gml:Envelope srsName="EPSG:4326">
             <gml:pos srsDimension="2">-1.25709663323346
52.7447745036496</gml:pos>
             <gml:pos srsDimension="2">-1.25631067876913
52.7454577448837</gml:pos>
          </gml:Envelope>
       </gml:boundedBy>
       <t:gid>595</t:gid>
       <t:specification>
           <t:GeologicUnit gml:id="t.gu.BCTDATUF">
<t:name_urn>urn:cgi:feature:BGS:StratigraphicLexicon:BCT</t:name_urn>
                  <t:descriptive_name>BEACON TUFF MEMBER</t:descriptive_name>
                <t:age>urn:cgi:classifier:ICS:StratChart:2008:Ediacaran</t:age>
                  <t:rank>MEMBER</t:rank>
              </t:GeologicUnit>
          </t:specification>
          <t:geom>
              <gml:Polygon srsName="EPSG:4326">
                  <gml:outerBoundaryIs>
                      <gml:LinearRing>
                          <gml:coordinates cs="," decimal="." ts=""
                              >-1.25670372306689,52.7454577448837
                              -1.25689780981031,52.7453690631345
                              -1.25703356437367,52.7452260771792
```

```
                                -1.25709663323346,52.7450017282326
                                -1.25705403117502,52.7448935842841
                                -1.25692209903121,52.7448118503595
                                -1.25670053155272,52.7447745036496
                                -1.25650721154648,52.7448182412529
                                -1.25634229163705,52.7449340744674
                                 -1.25631067876913,52.745050743079
                                -1.2563380081549,52.7451857606649
                                 -1.25640962123418,52.7453300456306
                                -1.25655575427669,52.7454478281049
                                -1.25670372306689,52.7454577448837
                            </gml:coordinates>
                        </gml:LinearRing>
                    </gml:outerBoundaryIs>
                </gml:Polygon>
            </t:geom>
        </t:MappedFeature>
    </gml:featureMember>
...
</wfs:FeatureCollection>
```

example cb-example2.xml

Invoked using:

http://yourserver.org:8080/cocoon/geosciml/wfs/MappedFeature?DEBUG=true&SRC=cb-example2

Now returns:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:FeatureCollection xmlns:wfs="http://www.opengis.net/wfs"
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:h="http://apache.org/cocoon/request/2.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:proc="urn:x-lcnp:proc"
  xmlns:fn=http://www.w3.org/2005/xpath-functions
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:t=http://www.iugs.org/cgi/tempInternal
```

```
  xmlns:ogc="http://www.opengis.net/ogc">
    <gml:featureMember>
        <t:MappedFeature gml:id="t.mf.133">
            <gml:boundedBy>
                <gml:Envelope srsName="EPSG:4326">
                    <gml:pos srsDimension="2">-1.36881827459842
52.7705204628598</gml:pos>
                    <gml:pos srsDimension="2">-1.36757043721036
52.7712380023487</gml:pos>
                </gml:Envelope>
            </gml:boundedBy>
            <t:gid>712</t:gid>
            <t:specification>
                <t:GeologicUnit gml:id="t.gu.TILLMST">
            <t:name_urn>urn:cgi:feature:BGS:StratigraphicLexicon:TIL</t:name_urn>
                    <t:descriptive_name>TICKNALL LIMESTONE
FORMATION</t:descriptive_name>
                    <t:age>urn:cgi:classifier:ICS:StratChart:2008:Visean</t:age>
                    <t:rank>FORMATION</t:rank>
<t:lithology_1>urn:cgi:classifier:CGI:SimpleLithology:2008:calcareous_carbonate_s
edimentary_rock</t:lithology_1>
                </t:GeologicUnit>
            </t:specification>
            <t:geom>
                <gml:Polygon srsName="EPSG:4326">
                    <gml:outerBoundaryIs>
                        <gml:LinearRing>
                            <gml:coordinates cs="," decimal="." ts=""
                                >-1.36881827459842,52.7707837449203
                                -1.36864315115429,52.770594030513
                                -1.36833294954209,52.7705204628598
                                -1.36797646438526,52.7705725032038
                                -1.36767783057701,52.7707237368446
                                -1.36757043721036,52.7709748750386
                                -1.36773021570826,52.7712004677276
                                -1.36802612012787,52.7712380023487
                                -1.3684424156711,52.7711503215879
                                -1.36875639102448,52.7709632088151
```

```
                                    -1.36881827459842,52.7707837449203
                                </gml:coordinates>
                            </gml:LinearRing>
                        </gml:outerBoundaryIs>
                    </gml:Polygon>
                </t:geom>
            </t:MappedFeature>
        </gml:featureMember>
...
</wfs:FeatureCollection>
```

Note:
The above results are using our private schema because we haven't yet provided a GeoSciML mapping for the returned content.

3.9 Handler

The final step is to transform the private schema output to GeoSciML.
We need to create a directory structure under our geosciml/handler using the same logic as the mapping folder:

- geosciml <- already exists
  - handler <- already exists
    - MappedFeature<- must create... (the name of the WFS service profile)
      - bgs <- must create... (the name of the proc:datasource)

In the bgs directory, we then create an XSLT 2.0 file called MappedFeature.xslt; it is this stylesheet which will convert the private schema output to GeoSciML.
Our MappedFeature.xslt file in this case looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   xmlns:t="http://www.iugs.org/cgi/tempInternal"
   xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:gml="http://www.opengis.net/gml"
```

```
   xmlns:wfs="http://www.opengis.net/wfs"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wfs
http://schemas.opengis.net/wfs/1.1.0/wfs.xsd urn:cgi:xmlns:CGI:GeoSciML:2.0
http://www.geosciml.org/schemas/geosciml/2.0_rc2/geosciml.xsd"
   >
   <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
   <xsl:template match="t:MappedFeature">
      <gsml:MappedFeature gml:id="{@gml:id}">
         <xsl:apply-templates select="gml:boundedBy"/>
         <gsml:observationMethod>
            <gsml:CGI_TermValue>
               <gsml:value
codeSpace="urn:cgi:classifierScheme:BGS:ObservationMethod">fieldObservation</gsml
:value>
            </gsml:CGI_TermValue>
         </gsml:observationMethod>
         <gsml:observationMethod>
            <gsml:CGI_TermValue>
               <gsml:value
codeSpace="urn:cgi:classifierScheme:BGS:ObservationMethod">Boreholes</gsml:value>
            </gsml:CGI_TermValue>
         </gsml:observationMethod>
      <gsml:positionalAccuracy>
            <gsml:CGI_TermValue>
               <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
            </gsml:CGI_TermValue>
         </gsml:positionalAccuracy>
         <gsml:samplingFrame
xlink:href="urn:cgi:feature:BGS:EarthBedrockSurface"/>
         <gsml:specification>
            <xsl:apply-templates select="t:specification/t:GeologicUnit"/>
         </gsml:specification>
         <gsml:shape>
            <xsl:apply-templates select="t:geom/*"/>
         </gsml:shape>
      </gsml:MappedFeature>
```

```
   </xsl:template>
   <xsl:template match="t:GeologicUnit">
      <gsml:GeologicUnit gml:id="{@gml:id}">
         <gml:name codeSpace="http://www.cgi-iugs.org/uri">
            <xsl:value-of select="t:name_urn"/>
         </gml:name>
         <gml:name
codeSpace="urn:cgi:classifierScheme:BGS:StratigraphicLexicon:description">
            <xsl:value-of select="t:descriptive_name"/>
         </gml:name>
         <gsml:observationMethod>
            <gsml:CGI_TermValue>
               <gsml:value
codeSpace="urn:cgi:classifier:BGS:ObservationMethod">Summary of published
description</gsml:value>
            </gsml:CGI_TermValue>
         </gsml:observationMethod>
         <gsml:purpose>instance</gsml:purpose>
         <gsml:preferredAge>
            <gsml:GeologicEvent>
               <gsml:eventAge>
                  <gsml:CGI_TermValue>
                     <gsml:value
codeSpace="urn:cgi:classifierScheme:ICS:StratChart:2008">
                        <xsl:value-of select="t:age"/>
                     </gsml:value>
                  </gsml:CGI_TermValue>
               </gsml:eventAge>
               <gsml:eventEnvironment>
                  <gsml:CGI_TermValue>
                     <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
                  </gsml:CGI_TermValue>
               </gsml:eventEnvironment>
               <gsml:eventProcess>
                  <gsml:CGI_TermValue>
                     <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
```

```
                  </gsml:CGI_TermValue>
               </gsml:eventProcess>
            </gsml:GeologicEvent>
         </gsml:preferredAge>
         <gsml:geologicUnitType

xlink:href="urn:cgi:classifier:CGI:GeologicUnitType:200811:lithostratigraphic_uni
t"/>
         <gsml:rank codeSpace="urn:cgi:classifierScheme:BGS:Rank">
            <xsl:value-of select="t:rank"/>
         </gsml:rank>
         <gsml:classifier xlink:href="{t:name_urn}"/>
         <xsl:if test="t:lithology_1">
         <gsml:composition>
            <gsml:CompositionPart>
               <gsml:role
codeSpace="urn:cgi:classifierScheme:BGS:RoleVocab">dominantConstituent</gsml:role
>
               <gsml:lithology xlink:href="{t:lithology_1}"/>
               <xsl:if test="t:lithology_2">
                  <gsml:lithology xlink:href="{t:lithology_2}"/>
               </xsl:if>
               <gsml:proportion>
                  <gsml:CGI_NumericValue>
                     <gsml:principalValue
uom="urn:ogc:def:uom:UCUM::%25">100</gsml:principalValue>
                  </gsml:CGI_NumericValue>
               </gsml:proportion>
            </gsml:CompositionPart>
         </gsml:composition>
         </xsl:if>
      </gsml:GeologicUnit>
   </xsl:template>
   <xsl:template match="@*|node()">
      <xsl:copy>
         <xsl:apply-templates select="@*|node()"/>
      </xsl:copy>
   </xsl:template>
```

```
</xsl:stylesheet>
```

Now when we repeat our previous step using our example (cb-example2.xml) we get a full GeoSciML result ‘

example cb-example2.xml

Invoked using:

http://yourserver.org:8080/cocoon/geosciml/wfs/MappedFeature?DEBUG=true&SRC=cb-example2

Now returns:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<wfs:FeatureCollection xmlns:wfs="http://www.opengis.net/wfs"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:h="http://apache.org/cocoon/request/2.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:proc="urn:x-lcnp:proc"
  xmlns:fn="http://www.w3.org/2005/xpath-functions"
xmlns:gml="http://www.opengis.net/gml"
  xmlns:t="http://www.iugs.org/cgi/tempInternal"
xmlns:ogc="http://www.opengis.net/ogc">
  <gml:featureMember>
    <gsml:MappedFeature xmlns:gsml="urn:cgi:xmlns:CGI:GeoSciML:2.0"
gml:id="t.mf.133">
      <gml:boundedBy>
        <gml:Envelope srsName="EPSG:4326">
          <gml:pos srsDimension="2">-1.36881827459842 52.7705204628598</gml:pos>
          <gml:pos srsDimension="2">-1.36757043721036 52.7712380023487</gml:pos>
        </gml:Envelope>
      </gml:boundedBy>
      <gsml:observationMethod>
        <gsml:CGI_TermValue>
          <gsml:value codeSpace="urn:cgi:classifierScheme:BGS:ObservationMethod"
            >fieldObservation</gsml:value>
        </gsml:CGI_TermValue>
      </gsml:observationMethod>
```

```
      <gsml:observationMethod>
        <gsml:CGI_TermValue>
          <gsml:value codeSpace="urn:cgi:classifierScheme:BGS:ObservationMethod"
            >Boreholes</gsml:value>
        </gsml:CGI_TermValue>
      </gsml:observationMethod>
      <gsml:positionalAccuracy>
        <gsml:CGI_TermValue>
          <gsml:value codeSpace="http://urn.opengis.net"
            >urn:ogc:def:nil:OGC::unknown</gsml:value>
        </gsml:CGI_TermValue>
      </gsml:positionalAccuracy>
      <gsml:samplingFrame xlink:href="urn:cgi:feature:BGS:EarthBedrockSurface"/>
      <gsml:specification>
        <gsml:GeologicUnit gml:id="t.gu.TILLMST">
          <gml:name codeSpace="http://www.cgi-iugs.org/uri"
            >urn:cgi:feature:BGS:StratigraphicLexicon:TIL</gml:name>
          <gml:name
codeSpace="urn:cgi:classifierScheme:BGS:StratigraphicLexicon:description"
            >TICKNALL LIMESTONE FORMATION</gml:name>
          <gsml:observationMethod>
            <gsml:CGI_TermValue>
              <gsml:value
codeSpace="urn:cgi:classifier:BGS:ObservationMethod">Summary
                of published description</gsml:value>
            </gsml:CGI_TermValue>
          </gsml:observationMethod>
          <gsml:purpose>instance</gsml:purpose>
          <gsml:preferredAge>
            <gsml:GeologicEvent>
              <gsml:eventAge>
                <gsml:CGI_TermValue>
                  <gsml:value
codeSpace="urn:cgi:classifierScheme:ICS:StratChart:2008">urn:cgi:classifier:ICS:S
tratChart:2008:Visean</gsml:value>
                </gsml:CGI_TermValue>
              </gsml:eventAge>
              <gsml:eventEnvironment>
```

```
                <gsml:CGI_TermValue>
                  <gsml:value codeSpace="http://urn.opengis.net"
>urn:ogc:def:nil:OGC::unknown</gsml:value>
                </gsml:CGI_TermValue>
              </gsml:eventEnvironment>
              <gsml:eventProcess>
                <gsml:CGI_TermValue>
                  <gsml:value
codeSpace="http://urn.opengis.net">urn:ogc:def:nil:OGC::unknown</gsml:value>
                </gsml:CGI_TermValue>
              </gsml:eventProcess>
            </gsml:GeologicEvent>
          </gsml:preferredAge>
          <gsml:geologicUnitType
xlink:href="urn:cgi:classifier:CGI:GeologicUnitType:200811:lithostratigraphic_uni
t"/>
          <gsml:rank
codeSpace="urn:cgi:classifierScheme:BGS:Rank">FORMATION</gsml:rank>
          <gsml:classifier
xlink:href="urn:cgi:feature:BGS:StratigraphicLexicon:TIL"/>
          <gsml:composition>
            <gsml:CompositionPart>
              <gsml:role
codeSpace="urn:cgi:classifierScheme:BGS:RoleVocab">dominantConstituent</gsml:role
><gsml:lithology
xlink:href="urn:cgi:classifier:CGI:SimpleLithology:2008:calcareous_carbonate_sedi
mentary_rock"/>
              <gsml:proportion>
                <gsml:CGI_NumericValue>
                  <gsml:principalValue uom="urn:ogc:def:uom:UCUM:%25"
                    >100</gsml:principalValue>
                </gsml:CGI_NumericValue>
              </gsml:proportion>
            </gsml:CompositionPart>
          </gsml:composition>
        </gsml:GeologicUnit>
      </gsml:specification>
      <gsml:shape>
```

```
        <gml:Polygon srsName="EPSG:4326">
          <gml:outerBoundaryIs>
            <gml:LinearRing>
              <gml:coordinates cs="," decimal="." ts=""
                >-1.36881827459842,52.7707837449203
                -1.36864315115429,52.770594030513
                -1.36833294954209,52.7705204628598
                -1.36797646438526,52.7705725032038
                -1.36767783057701,52.7707237368446
                -1.36757043721036,52.7709748750386
                -1.36773021570826,52.7712004677276
                -1.36802612012787,52.7712380023487
                -1.3684424156711,52.7711503215879
                -1.36875639102448,52.7709632088151
                -1.36881827459842,52.7707837449203
              </gml:coordinates>
            </gml:LinearRing>
          </gml:outerBoundaryIs>
        </gml:Polygon>
      </gsml:shape>
    </gsml:MappedFeature>
  </gml:featureMember>
  ...
</wfs:FeatureCollection>
```

and that's it! You can now make standard WFS requests or use a WFS client with the URL `http://yourserver.org:8080/cocoon/geosciml/wfs/MappedFeature?` For example get some features with:
`http://yourserver.org:8080/cocoon/geosciml/wfs/MappedFeature?service=WFS&version=1.1.0&request=GetFeature&typename=gsml:MappedFeature&maxfeatures=20&`

Note
If you create your MappedFeature.xslt but get no result (or an error), double check that you are using the same namespace in your XSLT as that returned by your private schema mapping, both in the root element namespace declaration and all the xsl element rules.