GeoServer WPS

An integrated Web Processing Service

Andrea Aime

aaime@opengeo.org

OpenGeo

http://opengeo.org

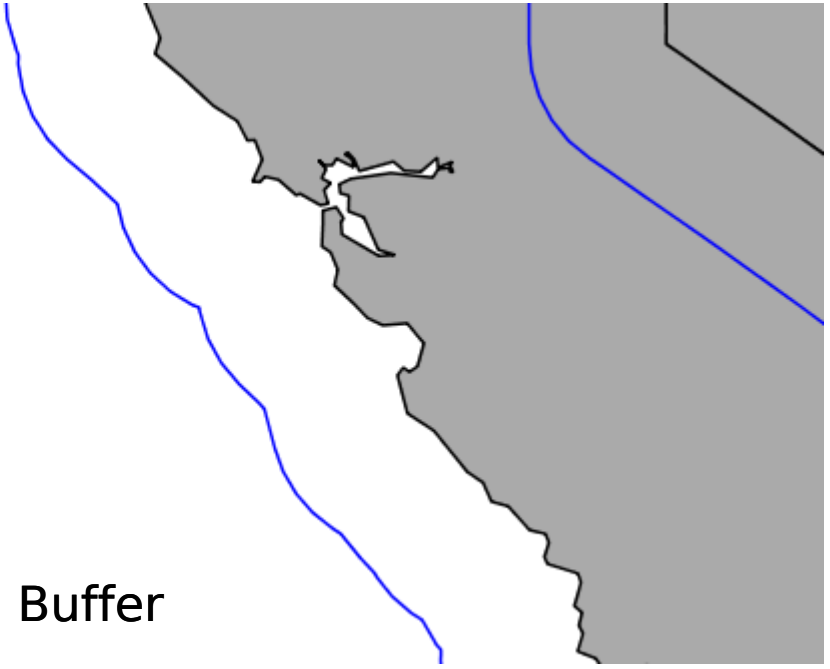
OpenGeo
http://opengeo.org

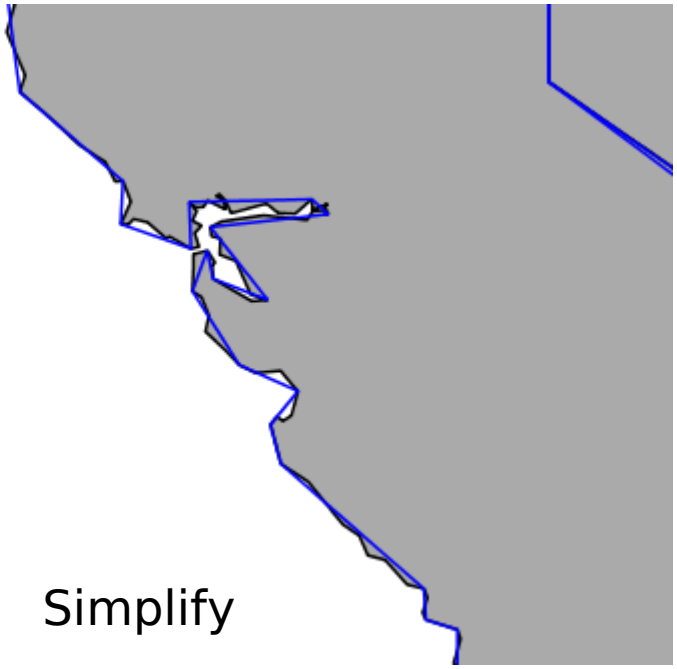
Quick introduction to WPS

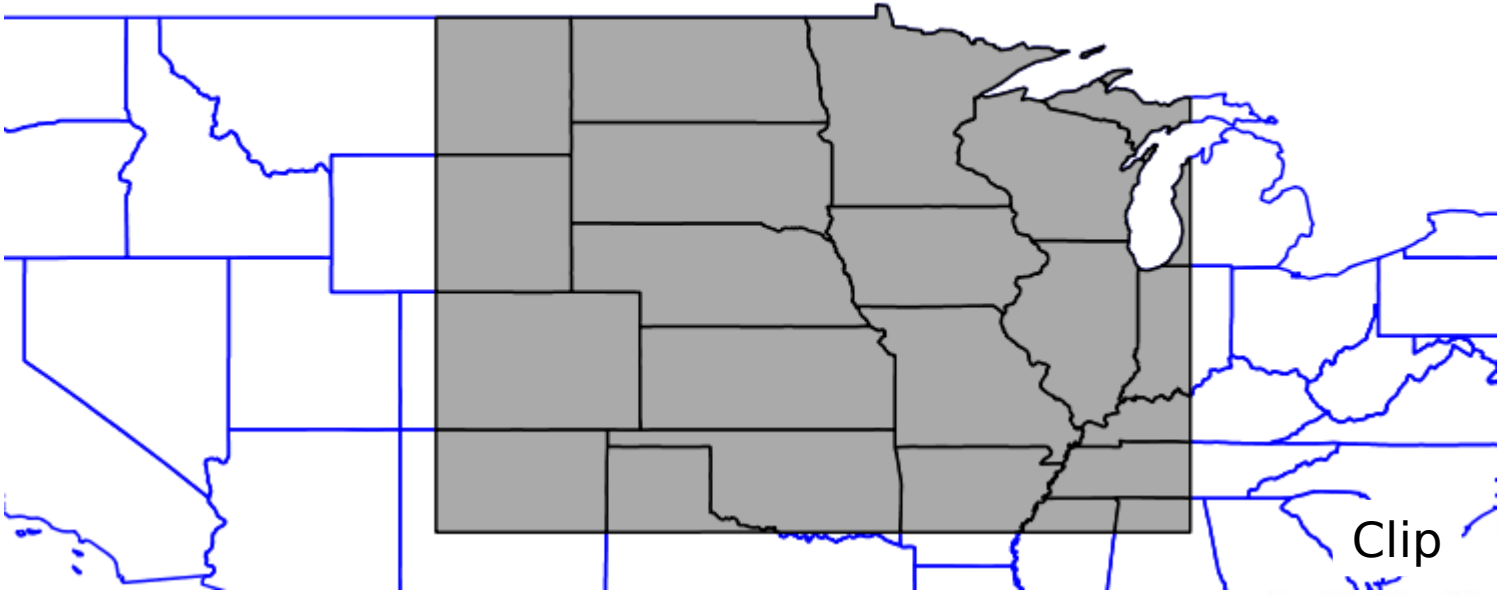
Web Processing Service

- Wikipedia introduces OGC WPS as:
  - [A service] designed to standardize the way that *GIS* calculations are made available to the Internet.
  - WPS can describe any calculation including all of its inputs and outputs, and trigger its execution
  - The specific processes served up by a WPS implementation are defined by the owner of that implementation.
  - Although WPS was designed to work with spatially referenced data, it can be used with any kind of data.

OpenGeo
http://opengeo.org


Buffer


Simplify


Clip

Operations

- *GetCapabilities*
  - Server metadata
  - List of processes
- *DescribeProcess*
  - Human process description
  - Machine input/output description
- *Execute*
  - Provide inputs, invoke the process, gather the outputs

OpenGeo
http://opengeo.org

The simplest example

gt:DoubleAddition(input_a, input_b)
= input_a + input_b

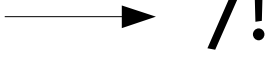
OpenGeo
http://opengeo.org

DescribeProcess: sum

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:ProcessDescriptions service="WPS" version="1.0.0"
  xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:ows="http://www.opengis.net/ows/1.1"
  xmlns:xlink="http://www.w3.org/1999/xlink">
  <wps:ProcessDescription wps:processVersion="1.0.0"
    statusSupported="false" storeSupported="false">
    <ows:Identifier>gt:DoubleAddition</ows:Identifier>
    <ows:Title>DoubleAddition</ows:Title>
    <ows:Abstract>Adds two floating point numbers</ows:Abstract>
    <wps:DataInputs>
      <wps:Input maxOccurs="1" minOccurs="1">
        <ows:Identifier>input_a</ows:Identifier>
        <ows:Title>First value</ows:Title>
        <ows:Abstract>First value to add</ows:Abstract>
        <wps:LiteralData><ows:DataType>xs:double</ows:DataType>
                         <ows:AnyValue /></wps:LiteralData>
      </wps:Input>
      <wps:Input maxOccurs="1" minOccurs="1">
        <ows:Identifier>input_b</ows:Identifier>
        <ows:Title>Second value</ows:Title>
        <ows:Abstract>Second value to add</ows:Abstract>
        <wps:LiteralData><ows:DataType>xs:double</ows:DataType>
                         <ows:AnyValue /></wps:LiteralData>
      </wps:Input>
    </wps:DataInputs>
    <wps:ProcessOutputs>
      <wps:Output>
        <ows:Identifier>result</ows:Identifier>
        <ows:Title>Result value</ows:Title>
        <wps:LiteralOutput><ows:DataType>xs:double</ows:DataType>
        </wps:LiteralOutput></wps:Output>
    </wps:ProcessOutputs>
  </wps:ProcessDescription>
</wps:ProcessDescriptions>
```

Execute example: sum

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute version="1.0.0" service="WPS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opengis.net/wps/1.0.0" xmlns:ows="http://www.opengis.net/ows/1.1"
  xmlns:gml="http://www.opengis.net/gml"
  xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
                      http://schemas.opengis.net/wps/1.0.0/wpsAll.xsd">
  <ows:Identifier>gt:DoubleAddition</ows:Identifier>
  <wps:DataInputs>
    <wps:Input>
      <ows:Identifier>input_a</ows:Identifier>
      <wps:Data>
        <wps:LiteralData>2</wps:LiteralData>
      </wps:Data>
    </wps:Input>
    <wps:Input>
      <ows:Identifier>input_b</ows:Identifier>
      <wps:Data>
        <wps:LiteralData>5</wps:LiteralData>
      </wps:Data>
    </wps:Input>
  </wps:DataInputs>
  <wps:ResponseForm>
    <wps:RawDataOutput>
      <ows:Identifier>result</ows:Identifier>
    </wps:RawDataOutput>
  </wps:ResponseForm>
</wps:Execute>
```



OpenGeo
http://opengeo.org

GeoServer WPS at a glance

GeoServer WPS history

- Started by Refractions in 2008, with limited capabilities (only single geometry and single feature support)
- First overhaul attempt end of 2008 by the community, added testing, support for vector collections
- In heavy development since mid 2010 with Sextante integration, JTS processes, raster data support

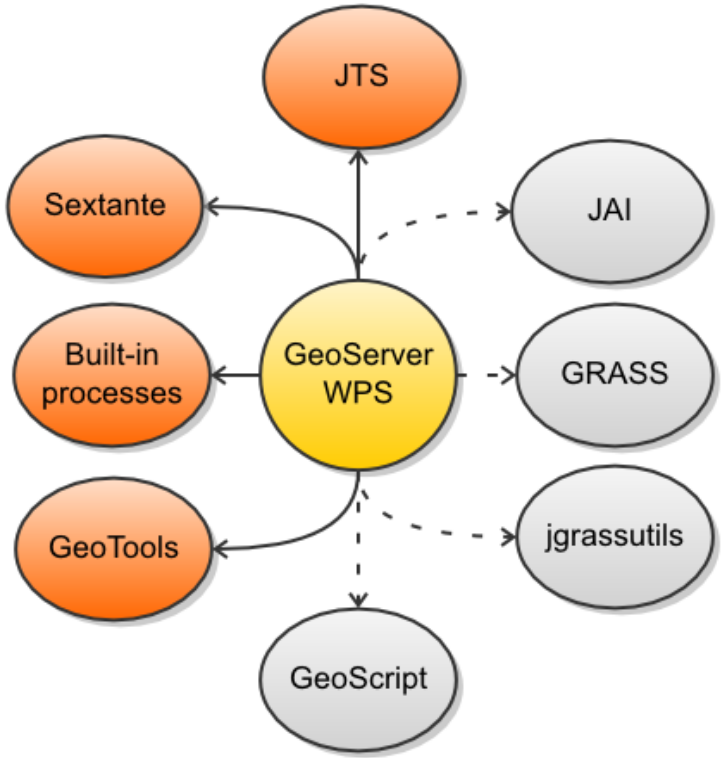
Inputs and outputs

- GeoServer supports
  - Primitives: strings, numbers, dates, bounding boxes
  - Plain geometries: in WKT and GML 2/3 format
  - Feature collections: GML 2/3, GeoJSON, zipped shapefile
  - Rasters: GeoTiff and ArcGrid
- The WPS spec does not really say what input and outputs one has to support, and in which format!

OpenGeo
http://opengeo.org

Process sources

- **JTS**: 45 simple geometry manipulation processes
- **Sextante**: over 200 raster processes
- **Built-in**: 10 processes to improve over WFS and interact with the catalog
- **GeoTools**: the pluggable process API



OpenGeo
http://opengeo.org

Built-in demo client

WPS request builder

Step by step WPS request builder.

Choose process

gs:RectangularClip

Clips the features to the specified bounding box (WPS D

Process inputs

features* - SimpleFeatureCollection

The feature collection to be simplified

VECTOR_LAYER | shp:building

clip* - ReferencedEnvelope

The clipping area

Min X	Min Y	Max X
0	40.6	1

EPSG:4326 Find... EPSG:WGS 84

Process outputs

result* - SimpleFeatureCollection

The feature collection bounds

Generate application/zip

Execute process | Generate XML from process inputs/outputs

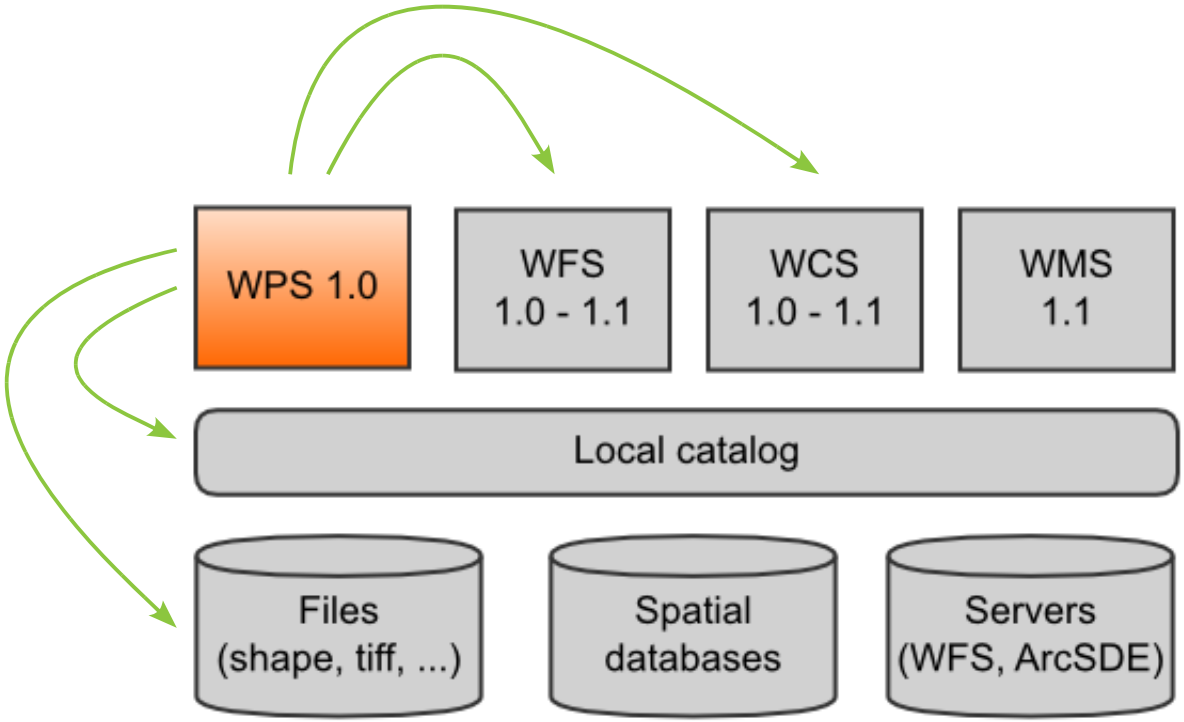
```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute version="1.0.0" service="WPS" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
  <ows:Identifier>gs:RectangularClip</ows:Identifier>
  <wps:DataInputs>
    <wps:Input>
      <ows:Identifier>features</ows:Identifier>
      <wps:Reference mimeType="text/xml; subtype=wfs-collection/1.0" xlink:href="http://geoserv
        <wps:Body>
          <wfs:GetFeature service="WFS" version="1.0.0" outputFormat="GML2">
            <wfs:Query typeName="shp:building"/>
          </wfs:GetFeature>
        </wps:Body>
      </wps:Reference>
    </wps:Input>
    <wps:Input>
      <ows:Identifier>clip</ows:Identifier>
      <wps:Data>
        <wps:BoundingBoxData crs="EPSG:4326" dimensions="2">
          <ows:LowerCorner>0.0 41.0</ows:LowerCorner>
          <ows:UpperCorner>1.0 406.0</ows:UpperCorner>
        </wps:BoundingBoxData>
      </wps:Data>
    </wps:Input>
  </wps:DataInputs>
  <wps:ResponseForm>
    <wps:RawDataOutput mimeType="application/zip">
      <ows:Identifier>result</ows:Identifier>
    </wps:RawDataOutput>
  </wps:ResponseForm>
</wps:Execute>
```

OpenGeo
http://opengeo.org

What makes GeoServer WPS different?

Integration!

- Direct communication with the other services, the catalog, the data sources



OpenGeo
http://opengeo.org

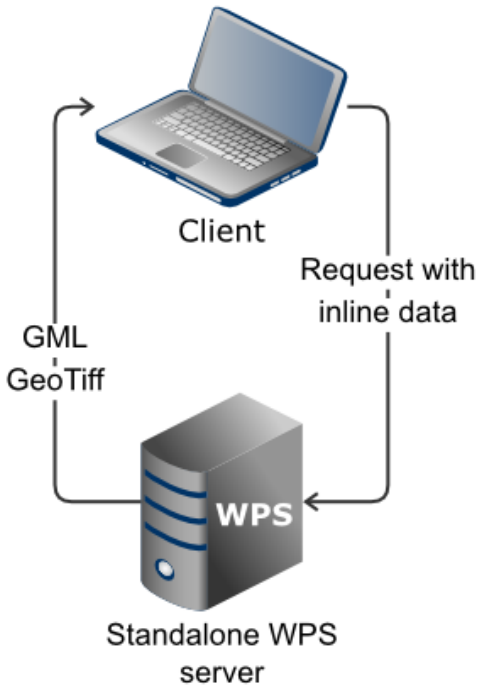
Communication patterns

- The data exchange with a WPS can be carried on in various ways
- Each has different assumptions on:
  - The client capabilities
  - The network speed
  - The availability of other services and data local to the WPS

OpenGeo
http://opengeo.org

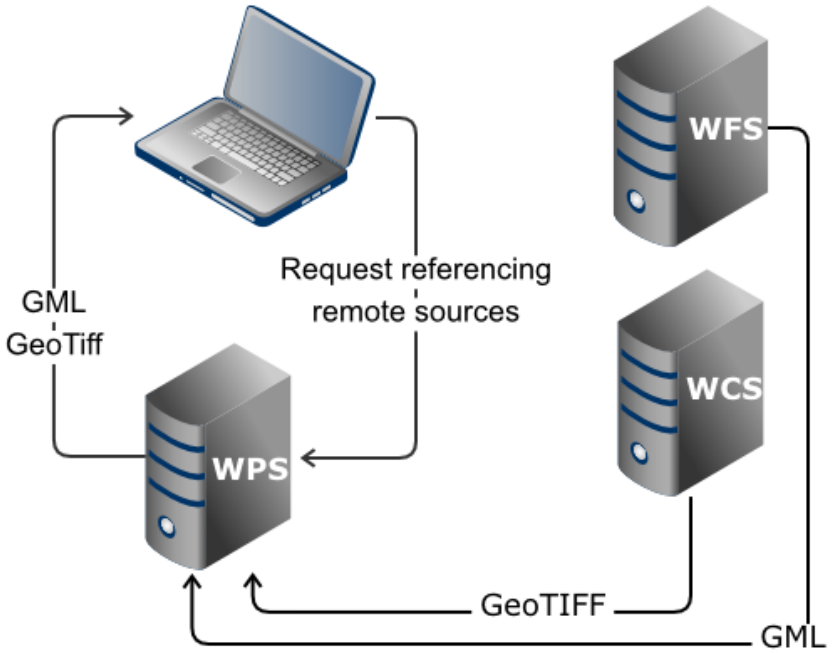
In-line data exchange

- The client sends over the data to be processed
- The server returns the result fully
- Assumptions
  - Fast network
  - GML overhead acceptable
  - Either desktop client or small amounts of data



Referring to remote servers

- Client sends a request referring to remote resources, the server responds fully
- Might work for thin clients if the result is small (for example, a summary)

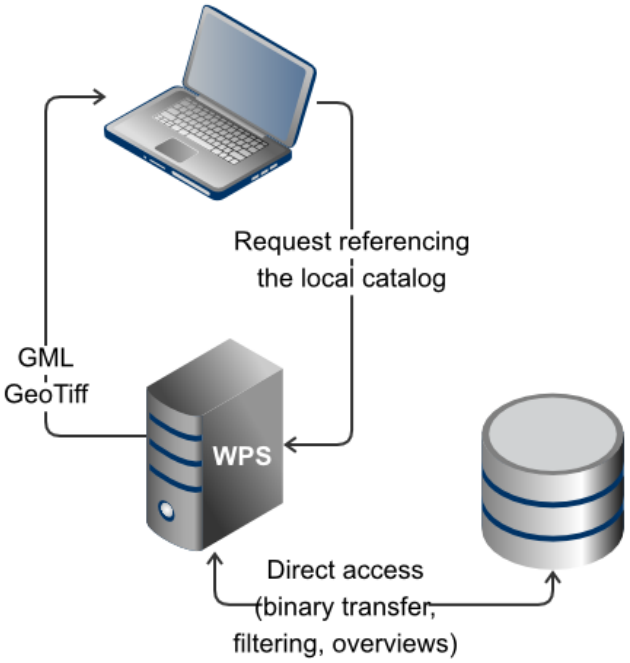


GML and network overhead still present

OpenGeo
http://opengeo.org

Referring to local catalog

- The client refers to some data that is local to the WPS server
- Still assumes few data or a summary type result

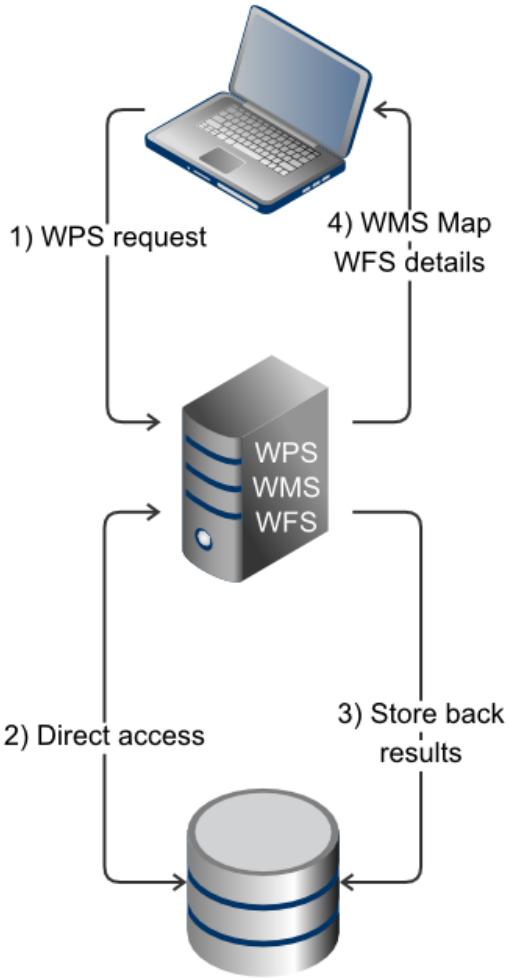


- Fast and native communication with the data source (native indexing, no ordinate value rounding, possible to offload part of the computation the data source itself)

OpenGeo
http://opengeo.org

Full service integration

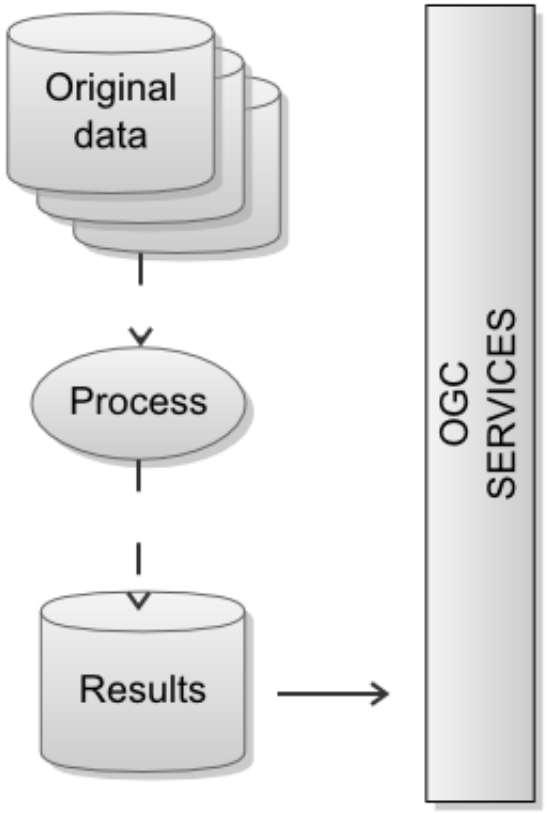
- The client refers to data local to the server
- The result is stored back in the integrated server
- The client accesses the results with WMS and WFS
- The best case scenario for a thin client (browser based)



OpenGeo
http://opengeo.org

Direct storage

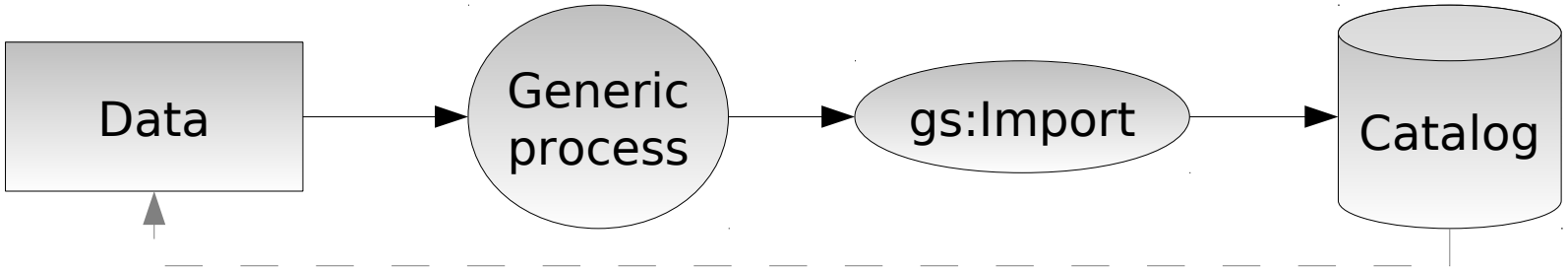
- The process is executed, the results stored and registered in the catalog
- Simple to implement
- Access to the results benefits from spatial indexing, etc.



OpenGeo
http://opengeo.org

Comm. patterns support

- The fourth style is supported via the **gs:Import** process + chaining



OpenGeo
http://opengeo.org

Services learn from each other

WFS cross pollination

- WFS common questions:
  - Can I get the **bounds** of the features that satisfy a certain filter? (without getting the features along?)
  - Can I get the min/max/avg/var/sum of a certain attribute? (**aggregate**)
  - Can I **simplify** the geometries so that my thin client won't choke on so many ordinates?
- The answer is: no, no and no!
- But WPS can!

OpenGeo
http://opengeo.org

Processes helping WFS

- *gs:Import*: imports vector data into the catalog
- *gs:Aggregate*: compute min/max/avg/sum/... over a certain attribute
- *gs:Unique*: return the unique values for a certain attribute
- *gs:Nearest*: find the N nearest features to a given point
- *gs:RectangularClip*: clips and returns features inside a certain rectangle
- *gs:Simplify:* generalize geometries
- *gs:Snap*: snaps the given point to the closest geometry vertexes

OpenGeo
http://opengeo.org

WPS learning from WFS

- Doing what WFS is capable of, but on a remotely provided data set:
  - gs:Count: count how many features
  - gs:Reproject: reproject a feature collection
  - Query like filtering is one the way

Sneak peek into the future

Scripting

- A server can have hundreds of processes
- But it often happens that it does not have the **one** you need!
- We want to *effortlessly* add new processes with a *light coding environment* and *without restarting* the server → ***scripting***!

GeoScript

- http://geoscript.org
- “GeoScript adds spatial capabilities to dynamic scripting languages”
- GeoScript is based on GeoTools just like GeoServer
- Work is underway to integrate it into GeoServer as a source of WPS processes
- Imagine...

Scripting processes

- Imagine:
  - Writing a process in Python, Javascript, Groovy or Scala
  - Deploying/updating it to the WPS by simply copying a file
  - Cover your specific needs in a short time and get on with your work
- If high performance or better integration is needed, Java is still available for a native process implementation!

Advanced raster processes

- jgrasstools provides a variety of scalable, pure Java processes based on JAI and GeoTools
- Various of them are tile based, and leverage the extra performance of Java Advanced Images libraries
- Stay tuned for advanced raster processes in GeoServer!
- http://code.google.com/p/jgrasstools/

OpenGeo
http://opengeo.org

Missing bits

- Adding support for asynchronous execution
  - Submit
  - Periodically check
  - Eventually get the results
- Better support for external schemas
- Support for unit of measure
- Go beyond process chaining into process orchestration (workflow engines)

OpenGeo
http://opengeo.org

Closing up

- WPS brings much needed new capabilities to GeoServer
- The module is brand new, it requires the whole community to test it (yes, I'm looking at you)
- Has a staggering potential for growth: join the team and help us make it great

OpenGeo
http://opengeo.org

Questions?

OpenGeo
http://opengeo.org

Extras

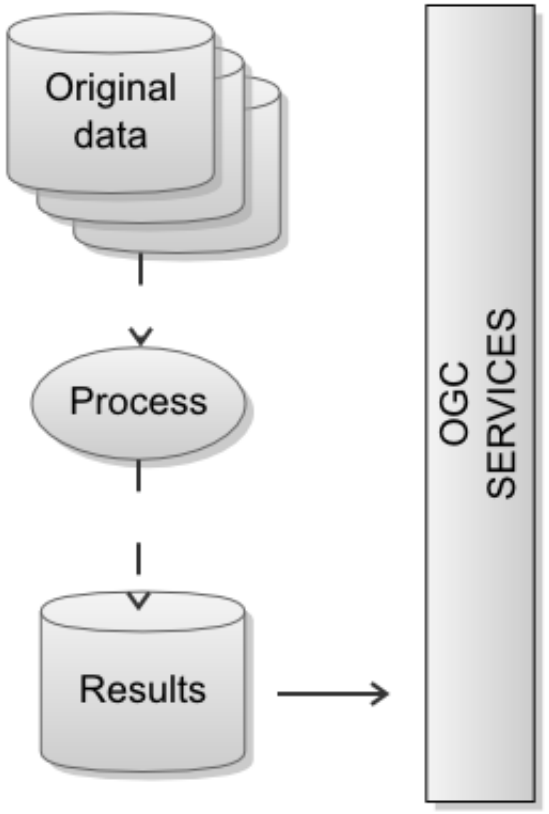
OpenGeo
http://opengeo.org

Result integration strategies

- Storing back in the catalog... how?
- Statically?
- Dynamically?
- Various approaches are possible

Direct storage

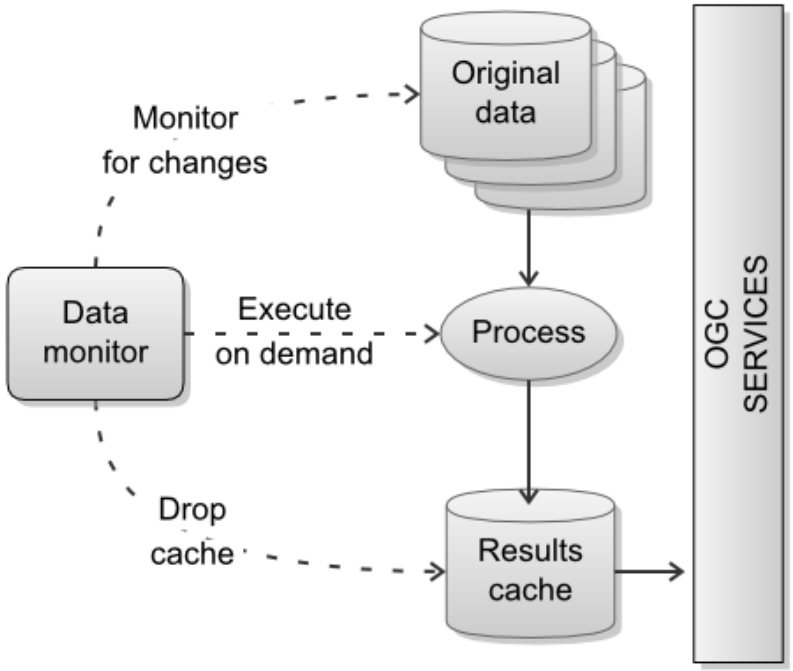
- The process is executed, the results stored and registered in the catalog
- Simple to implement,
- Loses all links to the process that generated the data
- Access to the results benefits from spatial indexing, etc.



OpenGeo
http://opengeo.org

Monitor and cache approach

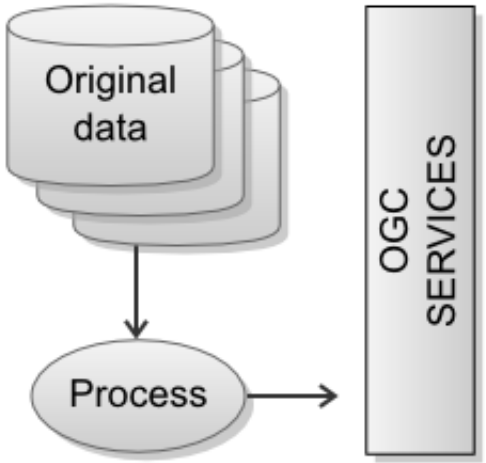
- The results are stored
- Yet, a data monitor maintains memory of the process and listens for changes in the original data
- On data change the process is run again and the results updated



OpenGeo
http://opengeo.org

On the fly approach

- Results are not stored
- The definition of the process and the original data are instead
- The process is run on the fly as there is demand for result data
- Works for very light processes



- Hard to implement efficiently over vector data: the process should consider the filters over the results to avoid needless computation (push vs pull)

OpenGeo
http://opengeo.org

Strategies support

- GeoServer implements the direct storage strategy via the gs:Import process
- Implementing a cache and monitor approach is not too difficult (monitor listens to WFS-T events)
- The third strategy is somewhat harder to implement, requires changes at the catalog level (a new type of layer)