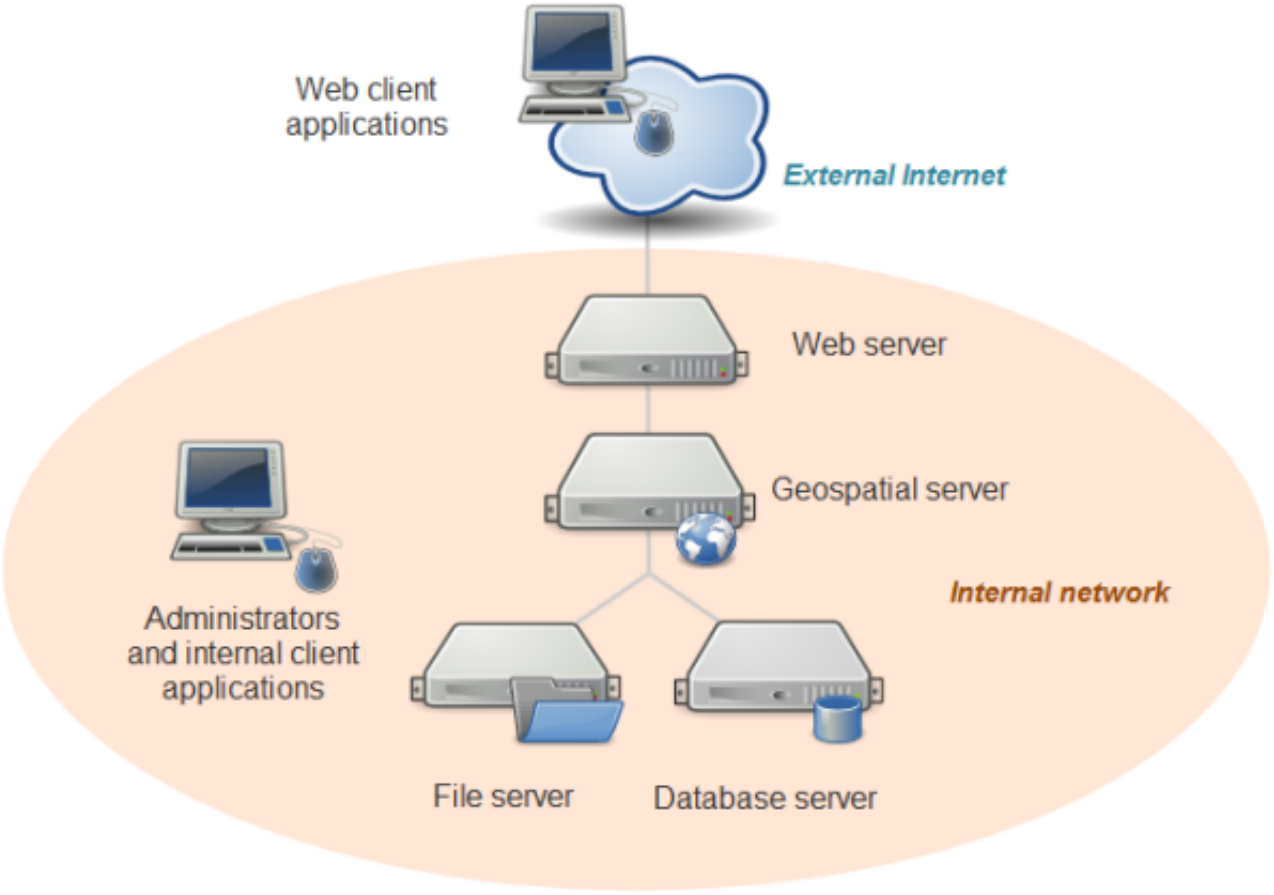
Les 8: Een backend opzetten of zelf programmeren

Om je eigen dat te serven heb je een backend nodig, die je data zal filteren en omzetten naar een webservice die leesbaar is voor web-applicatie's. Zoals WMS en Tiles voor raster data en gerenderde vectorkaarten of text-formaten zoals geojson voor niet-gerenderde vectordata.

Een type GIS-server-infrastructuur bestaat meestal uit 3 lagen:

- Een datalaag: meestal een ruimtelijke database zoals postgis, maar kan in principe ook een bestandssysteem zijn, zoals een folder met shapefiles.
- Een GIS-(applicatie)server: de backend voor webserver, die de GIS-data vertaald naar informatie die leesbaar is in webpagina, Geoserver en Mapserver.
- Een webserver waarop je html en eigen logica staat.



Een backend in de cloud

Er zijn tegenwoordig heel bedrijven die je data opslaan op hun server-infrastructuur en je toelaten om dit via een gebruiksvriendelijke webinterface te beheren. Daarnaast zijn ook al de data in deze webtoepassingen ook toegankelijk via een web-api om eigen toepassingen mee te ontwikkelen.

- **Cartodb (http://cartodb.com/)** is een platform dat erg gericht is op de visualisatie van vectordata. Via Spatial SQL kan je allerlei analyses en bewerkingen op je data doen.

- **MapBox (https://www.mapbox.com/)** is ook erg gericht op visualisatie van gegevens, maar werkt meer op basis van tiles. Ideaal om een basiskaart op maat van je bedrijf te maken.
- **Arcgis (http://www.arcgis.com/)** gebruikers kunnen tegenwoordig ook via Arcgis-online hun data beschikbaar maken in de cloud, zonder dat nog een eigen Arcgis server nodig hebt. Zonder Arcgis desktop is deze toepassing echter maar beperkt bruikbaar.

Een backend op eigen infrastructuur

Als een eigen server infrastructuur hebt kan je ook je services zelf hosten. Daarvoor zijn heel wat *out of the box* server applicaties beschikbaar. Rendering van ruwe data en gestandaardiseerde OGC-services zijn erg complex, daarom implementeer je dit best niet zelf. Dit soort services komen soms met een beheer en raadpleeg toepassing of worden geconfigureerd met text-bestanden. Vermits ze meestal OGC-services ondersteunen, zijn ze bruikbaar voor het aanbieden van data voor zowel Desktop als Web-toepassingen.

- **geoserver (http://geoserver.org/)** is een gebruiksvriendelijke Java server applicatie die alle belangrijke OGC-standaarden (WMS, WMTS, WFS, WPS, ...) ondersteund. Alle configuratie van kan verlopen via een web interface, Rest-api of via een QGIS plugin. Scripting is mogelijk via plugin. Bijvoorbeeld DOV (https://www.dov.vlaanderen.be/geoserver/web/) en MercatorNet (https://mercator.vlaanderen.be/raadpleegdienstenmercatorpubliek/web/) zijn hierop gebaseerd.
- **mapserver (http://mapserver.org/)** mapserver is een geografische data server applicatie geprogrammeerd in C. Configuratie doe je via text-bestanden (Map-files) of via scripting-talen, meestal is dit python hoewel ook andere talen ondersteund zijn. Output is WMS, Tiles of WFS of afbeeldingen in vele formaten. AGIV gebruikt deze erg performante service voor zijn WMS-diensten.
- **arcgis server (http://www.esri.com/software/arcgis/arcgisserver)** ESRI heeft voor arcgis-gebruikers arcgis-server. Deze ondersteunt vele OGC-standaarden naast de eigen Rest-api.

Een eigen service programmeren

Vectoriële data is relatief eenvoudig te serven in een json-formaat zoals geojson, het is dus niet perse nodig om hiervoor een zeer uitgebreide *out the box*, met hoge systeem vereisten, applicatie server te voorzien zoals de bovenstaande toepassingen.

Als je data uit een databank komt zoals postgis kan je deze meestal via de standaard database-api aan te spreken. Voor bestandsformaten zoals shapefile heb je aan library nodig. Voor geometrische operaties zoals buffers, het dichtste punt bij en lijn bepalen, alle gegevens op een bepaalde afstand tot een andere geometrie bepalen en zo voort, gebruik je ook best een library. Als je deze gegevens wilt renderen naar een afbeelding zal wellicht ook een rendering-API nodig hebben.

Open source API's

C/C++

- **gdal (http://www.gdal.org/)** is een data-toegang library voor zowel vector als rasterdata, met bindingen voor vele talen, waaronder python (http://pcjericks.github.io/py-gdalogr-cookbook/index.html) en nodeJs (http://naturalatlas.github.io/node-gdal/classes/gdal.html).

- **geos (http://trac.osgeo.org/geos/)** geometrische operaties, ingebouwd in gdal en database API's zoals postgis.
- **proj4 (http://trac.osgeo.org/proj/)** projecties, met bindingen naar verschillden talen, zoals python (http://jswhit.github.io/pyproj).
- **mapnik (https://github.com/mapnik/mapnik)** kaart rendering framework, met bindingen naar verschillden talen nodeJs (https://github.com/mapnik/node-mapnik) en python (https://github.com/mapnik/pymapnik2).

python

- **geodjango (https://docs.djangoproject.com/en/1.7/ref/contrib/gis/)** webapplicatie framework, inclusief geometrische operaties en data-toegang. Alles wat je nodig hebt om je eigen services te maken.
- **shapely (http://toblerity.org/shapely/)** geometrische operaties.
- **fiona (http://toblerity.org/fiona/manual.html)** vector data-toegang.
- **rasterIO (https://github.com/mapbox/rasterio)** raster data-toegang en berekeningen.

nodeJs (javascript)

- **turfJs (http://turfjs.org/)** geometrische operaties in node of rechtstreeks in de browser.
- **proj4js (https://github.com/proj4js/proj4js)** projecties in node of rechtstreeks in de browser.

C-sharp

- **Net Topology Suite (https://github.com/NetTopologySuite/NetTopologySuite)** geometrische operaties
- **sharpmap (http://sharpmap.codeplex.com/)** rendering + data-toegang.

Java

- **Geotools (http://geotools.org/)** een all-inclusive toolkit voor het bouwen van desktop en server Java toepassingen.
- **Geomayas (http://www.geomajas.org/)** een client/server API in Java, die toelaat een webapplicatie volledig in Java te ontwikkelen, zonder dat nog iets Html/Javascript moet doen.
