SVG OPEN 2010 WORKSHOP

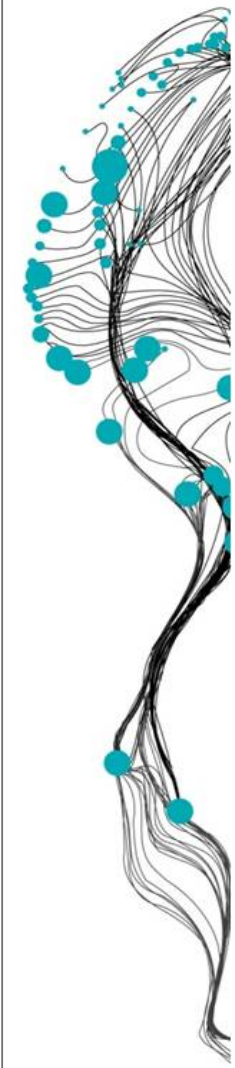
RIMapperWMS

An SVG
Web Mapping Service

Barend Köbben

kobben@itc.nl

ITC UNIVERSITY OF TWENTE.



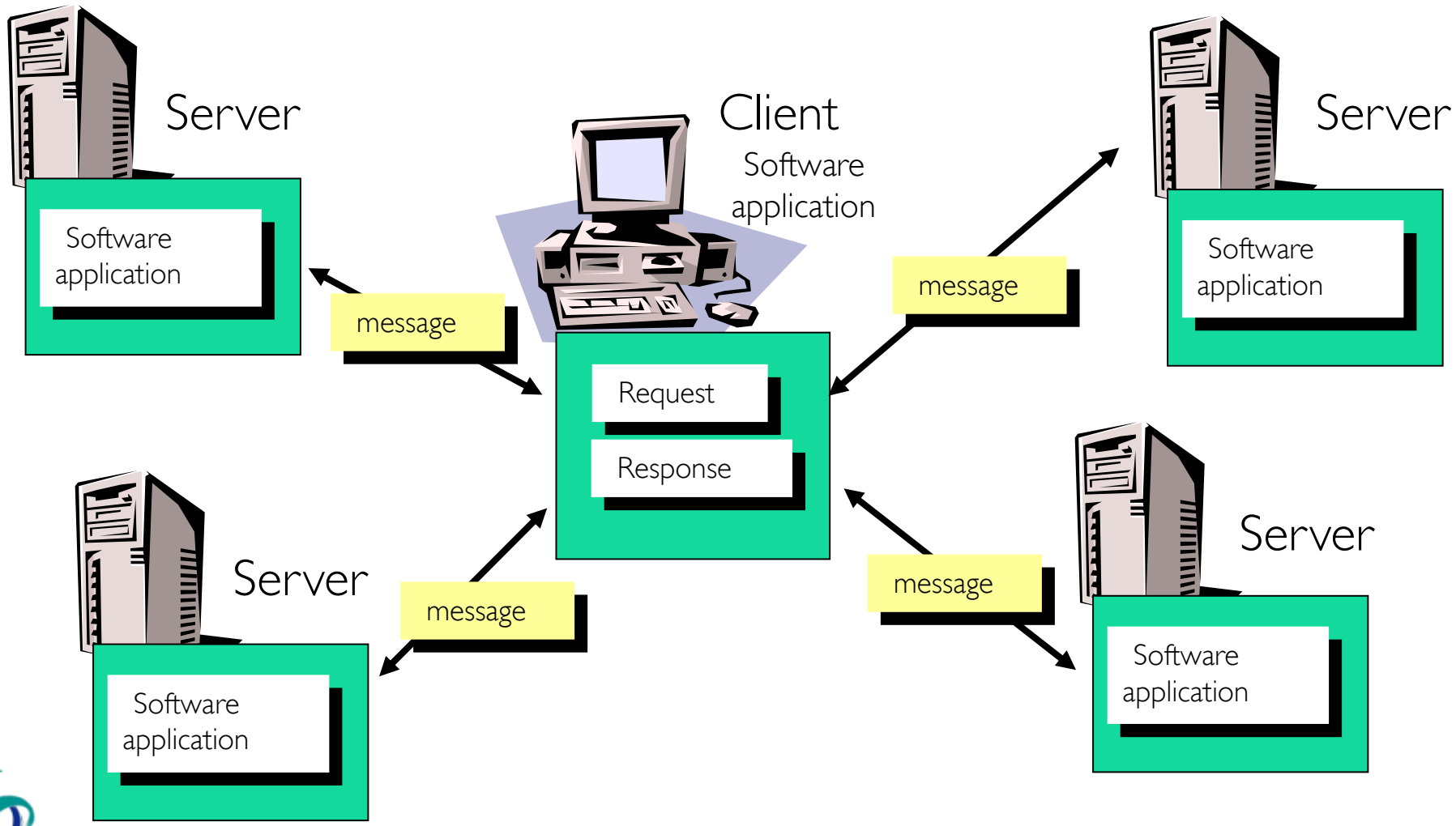
AGENDA

- Introducing RIMapperWMS
  - Geo-webservices
  - Open Geospatial Consortium standards
  - RIMapperWMS background
- Installing RIMapperWMS
  - PostgreSQL/Postgis spatial database
  - Tomcat Java Servlet container
  - The RIMapperWMS Java server application
- Setting up the RIMapperWMS services

interoperability
&
geowebservices

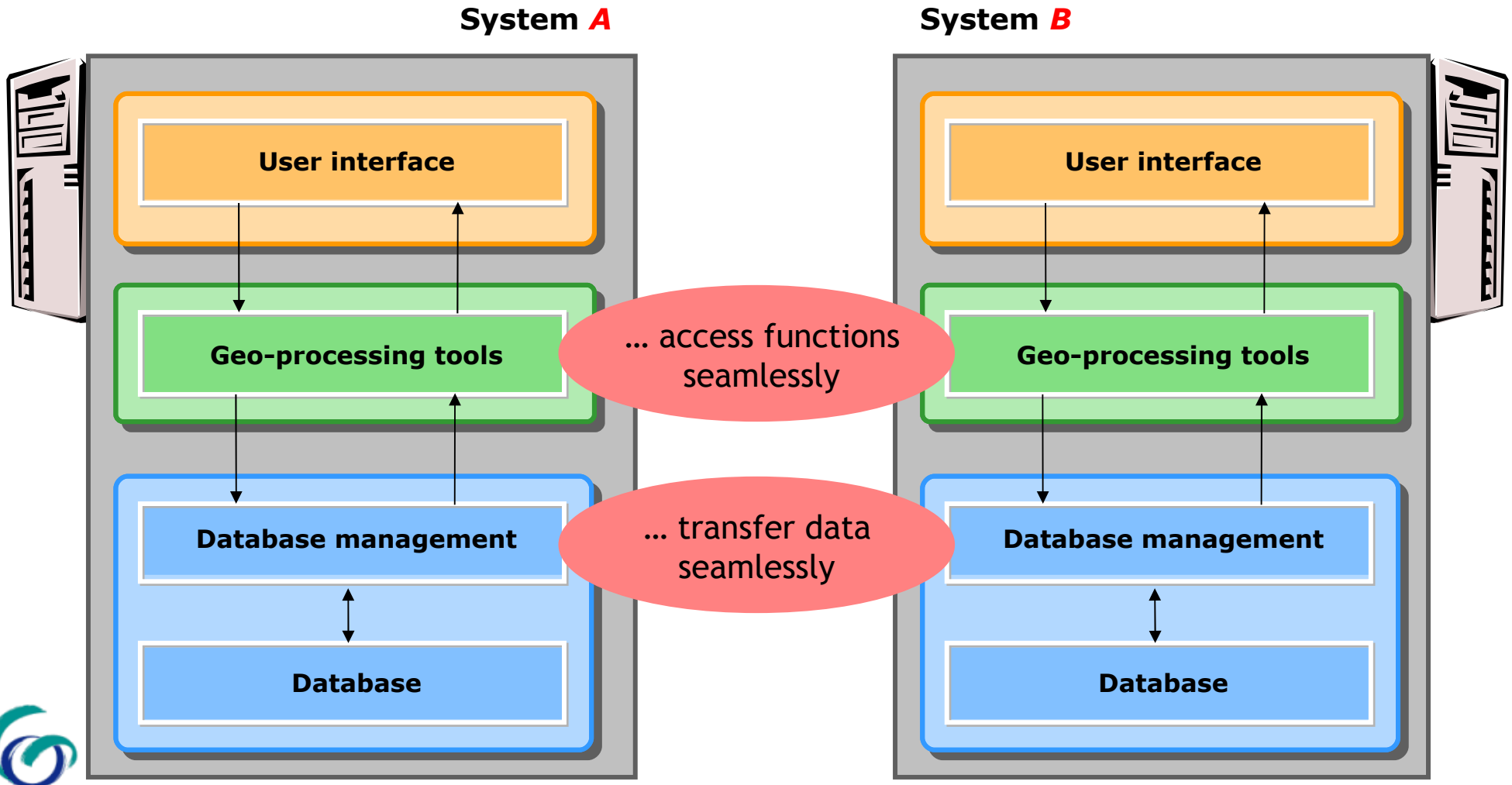
Interoperability

To communicate between systems we need to standardise the messages between them



Interoperability

Two information systems are interoperable, if they are able to …



How to achieve interoperability?

make data seamlessly transferable & accessible

- **Encode data** in a standardized, platform & application independent manner

XML

access distributed functionality seamlessly

- Specify and set up an infrastructure of interoperable (software) services, which encapsulate functionality and make it accessible via well **specified interfaces**

Web Services

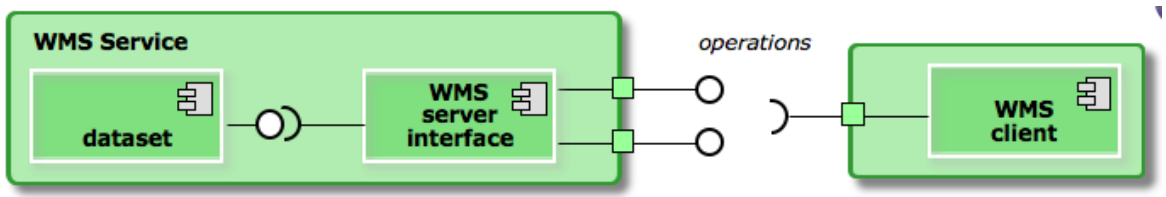
XML: eXtensible Markup Language

- Platform & application independent
  - Text format (like HTML)
  - Open Standard
- Separation of Content from Presentation
  - Provides a way to encode both structure and content of data (self descriptive)
  - Easy to execute structured queries
- Extensible
  - Tag-based, individual tags can be defined
  - Numerous XML languages for different purposes, e.g.
    - Geographical Markup Language (GML) for geographical data
    - Scalable Vector Graphics (SVG) for 2-dimensional graphics
    - etc...

WebServices

Web services are loosely coupled, contracted components that communicate via XML-based interfaces [Schmelzer 2002]

- loosely coupled:
  - can be changed independently
  - platform independent
- contracted
  - in– and output are publicly available
- components
  - interface encapsulates the code
- XML-based interfaces
  - human readable
  - self-describing (allows for discovery of their functionality)



Example

Currency converter with processable request and response: A flexible application using XML

- Request

```
<FromCurrency>USD</FromCurrency>
<ToCurrency >EUR</ToCurrency>
```

- Response

```
<double>0.92635</double>
```

many examples at http://www.webservicex.net/

GeoWebServices

- If webservices have *spatial* functionality, for example if they use geographic data, can output maps or find routes, we call them *geo*webservices
  - Google Maps, Bing maps, etc.: interfaces are publicly available, but defined, developed and owned by commercial companies
  - Open Standard GeoWebServices: Open Web Services (OWS) of the Open Geospatial Consortium (OGC).

Open Web Services Specifications

from the
Open Geospatial Consortium

OWS: Open Web Services & related Encodings

A set of Implementation Specifications for

(vector)data encoding:

- Geographic Markup Language (GML), Keyhole ML (KML)

Data Access:

- Web Feature Service (WFS)
- Web Coverage Service (WCS)

Portrayal:

- Web Mapping Service (WMS)
- Styled Layer Descriptor (SLD)
- Web Map Context Documents (WMCD)

Metadata:

- Catalog Service Web (CSW)

Geography Markup Language (GML)

XML encoding for the transport and storage of geographic information (vector)

Including both spatial and non-spatial properties of geographic features

GML does not contain information concerning the visual presentation of the data

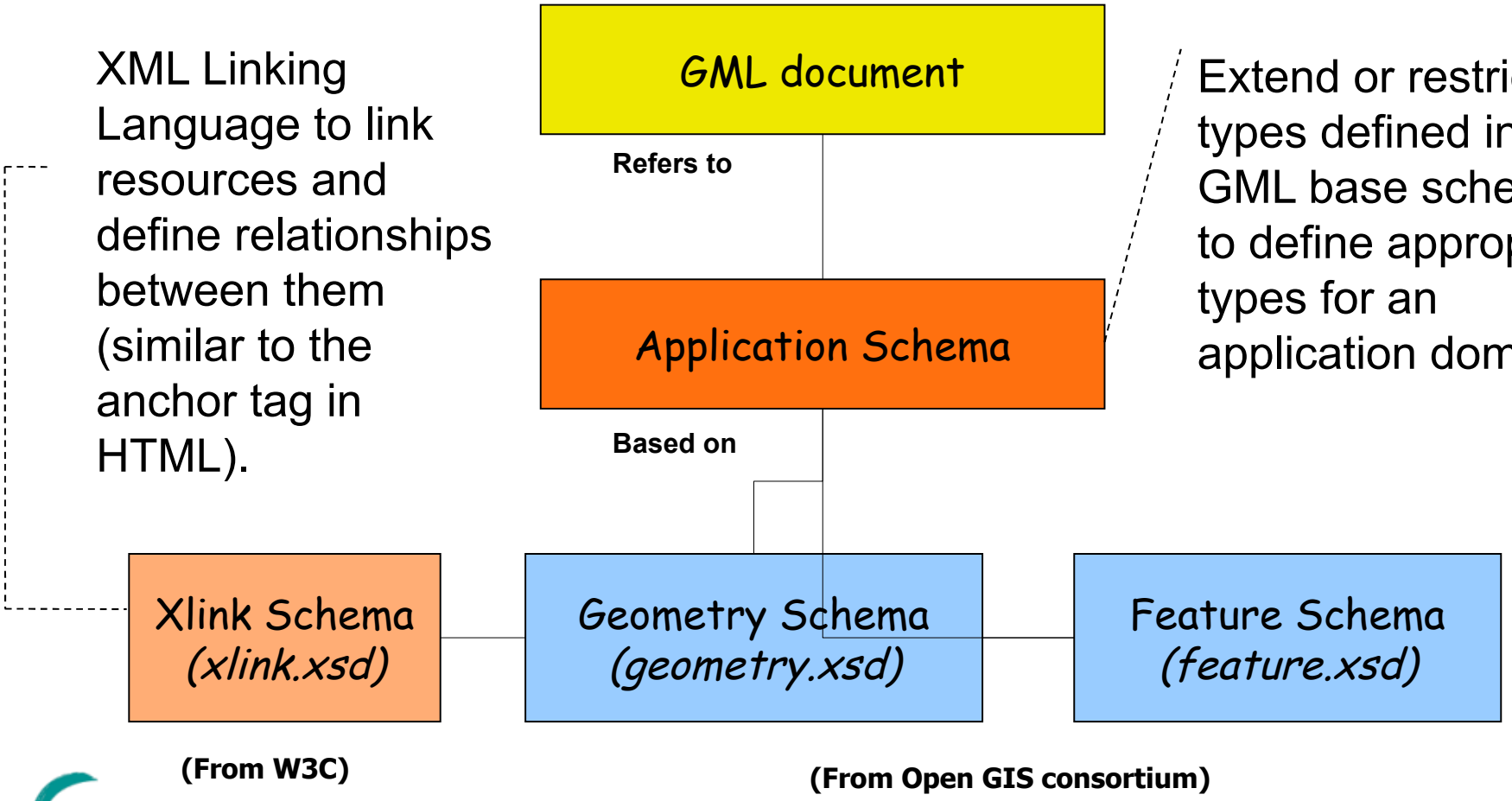
To draw a map is necessary to convert the GML into a graphic format, e.g. Scalable Vector Graphics

Extensible (application schemas)

Geography Markup Language

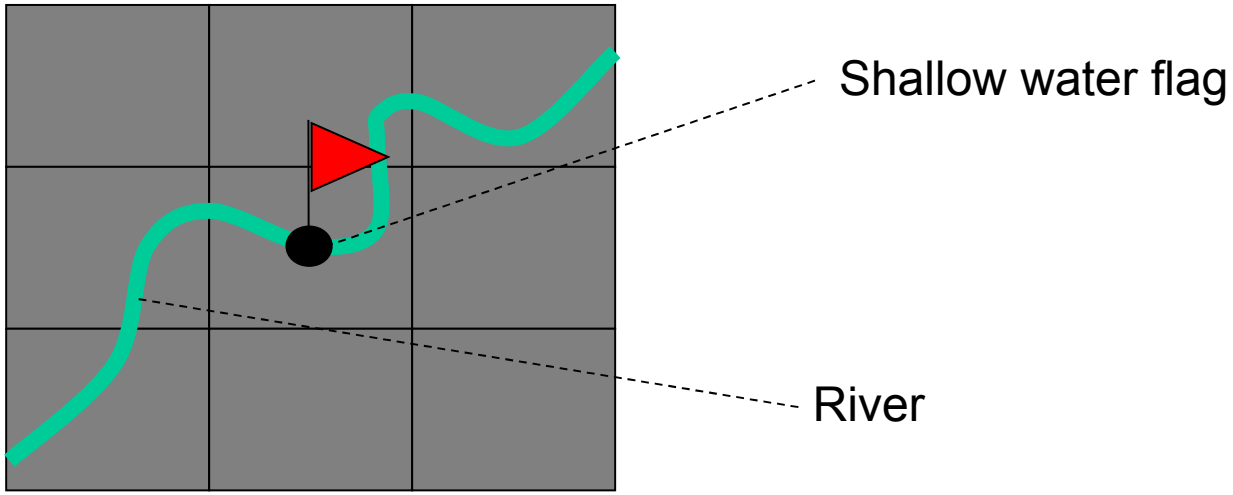
XML Schema's for GML data

XML Linking Language to link resources and define relationships between them (similar to the anchor tag in HTML).



Extend or restrict the types defined in the GML base schemas to define appropriate types for an application domain.

GML Feature Example




```
<Feature featureType="flag" fid="2" name="flag02">
    <Description>Shallow water flag</Description>
    <Geometry name="location" srsName="ESPG:4326">
        <Point>
            <CList>2.5,2.5</CList>
        </Point>
    </Geometry>
</Feature>
```

Web Feature Service (WFS)

- Standardized interface for accessing vector-data

  → The datastore used to store geographic features should be opaque to client applications and their only view of the data should be through the WFS interface

- WFS output is encoded in GML
- Optional web-interface for update, insert, and delete geo-data: WFS-T(ransactional)

WFS GetCapabilities

&SERVICE=WFS&VERSION=1.0.0&REQUEST=GetCapabilities

```
<WFS_Capabilities version="1.0.0" updateSequence="0" …name-spaces…>
<Service>
  <Name>MapServer WFS</Name>
  <Title>Nederland WFS test</Title>
  <OnlineResource>http://www.itc.nl/mapserv?/config_nl_wfs.map&</OnlineResource>
</Service>
<Capability>
 ...service parameters...
</Capability>
<FeatureTypeList>
       <FeatureType>
        <Name>gemeenten</Name>
        <Title>Gemeenten</Title>
        <SRS>EPSG:28992</SRS>
        <LatLongBoundingBox minx="3.25" miny="50.73" maxx="7.24" maxy="53.56"/>
    </FeatureType>
</FeatureTypeList>
<ogc:Filter_Capabilities>
  …filter parameters...
</ogc:Filter_Capabilities>
</WFS_Capabilities>
```

WFS DescribeFeatureType

`&REQUEST=DescribeFeatureType&TYPENAME=gemeenten`

generates a schema describing one or more feature types in detail

```
<schema version="0.1" >
<element name="gemeenten" type="ms:gemeentenType"/>
  <complexType name="gemeentenType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element name="multipolygon" type="gml:MultiPolygonPropertyType"
minOccurs="0" maxOccurs="1"/>
          <element name="GM_2003" type="string"/>
          <element name="GM2003" type="string"/>
          <element name="GM_NAAM" type="string"/>
          <element name="AANTINW" type="string"/>
          <element name="AANTMAN" type="string"/>
          <element name="AANTVROUW" type="string"/>
          <element name="ID" type="string"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</schema>
```

WFS GetFeature

&REQUEST=GetFeature&SRS=EPSG:28992&BBOX=136,306,278,619&
TYPENAME=gemeenten GML output in specific SRS & BBOX

```
<wfs:FeatureCollection>
      <gml:boundedBy>
        <gml:Box srsName="EPSG:28992">
          <gml:coordinates>136,306 278,619</gml:coordinates>
        </gml:Box>
      </gml:boundedBy>
    <gml:featureMember>
       <gml:MultiPolygon srsName="EPSG:28992">
        <gml:polygonMember>
          <gml:Polygon>
            <gml:outerBoundaryIs>
              <gml:LinearRing>
                <gml:coordinates> 226,615 226,615 225,617 225,617 …etc… </gml:coordinates>
              </gml:LinearRing>
            </gml:outerBoundaryIs>
          </gml:Polygon>
        </gml:polygonMember>
      </gml:MultiPolygon>
       <ms:GM_2003>1651</ms:GM_2003>
      <ms:GM2003>1651</ms:GM2003>
      <ms:GM_NAAM>Eemsmond</ms:GM_NAAM>
      <ms:AANTINW>17200</ms:AANTINW>
      <ms:AANTMAN>8770</ms:AANTMAN>
      <ms:ID>0</ms:ID>
  </gml:featureMember>
</wfs:FeatureCollection>
```

Web Coverage Service (WCS)

- Standardized interface for accessing gridded geo-data
- Primarily specified for the provision of imagery data
- Requests specification similar to WFS:
  - `GetCapabilities` for general capabilities of service
  - `DescribeCoverage` for full description of one or more coverages available
  - `GetCoverage` response is a raster file (eg. in geoTIFF, Imagine, DTED, etc…)

Web Map Service (WMS)

Standardized interface for the creation of superimposed map-like views of geographic information

Cascadable, meaning that one WMS can act as a ‘gateway’ to other ones

If configured in conjunction with WFS, flexible rendering is supported by user definable layer styles (→ Styled Layer Description specification)

WMS is actually the most mature and widest adopted OWS specification (numerous open source, as well as commercial solutions, e.g. WMS connector for ArcIMS)

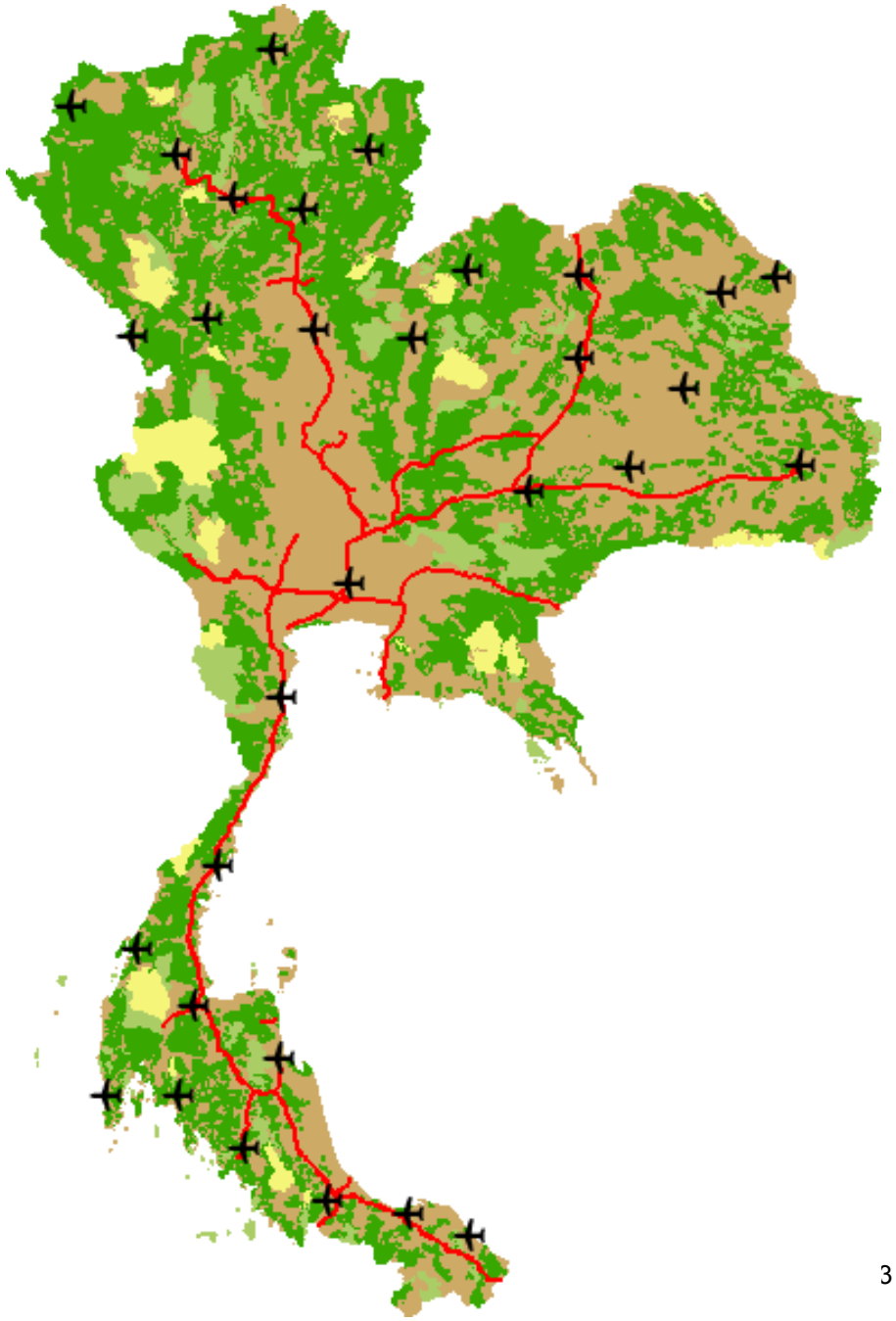
WMS GetCapabilities

&SERVICE=WMS&VERSION=1.1.1&REQUEST=GetCapabilities

```
<WMT_MS_Capabilities version="1.1.1">
  <Service>
    ...service metadata (name, title, keywords, etc)…
  </Service>
  <Capability>
      <GetMap>
                <Format>image/gif</Format>
           <Format>image/png</Format>
        </GetMap>
        <GetFeatureInfo>
           <Format>text/plain</Format>
           <Format>text/html</Format>
        </GetFeatureInfo>
     ... other service parameters...
  </Capability>
  <Layer queryable="1" opaque="0" cascaded="0">
    <Name>airports</Name>
    <Title>airports</Title>
    <SRS>EPSG:4326</SRS>
    <BoundingBox SRS="EPSG:4326" minx="97.9" miny="6.5" maxx="104.8" maxy="19.9" /
  </Layer>
    ...other layers...
</WFS_Capabilities>
```

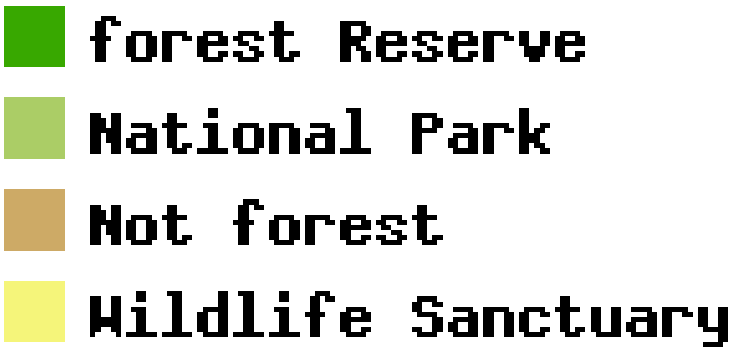
WMS GetMap

```
&SERVICE=WMS
&VERSION=1.1.1
&REQUEST=GetMap
&LAYERS=forest,railroad,
   airports
&SRS=EPSG:4326
&BBOX=97.3,5.6,105.6,20.4
&WIDTH=400&HEIGHT=600
&FORMAT=image/png
```



WMS GetLegendGraphic

```
&SERVICE=WMS
&REQUEST=GetLegendGraphic
&VERSION=1.1.1
&FORMAT=image/png
&LAYER=forest
```




```
&SERVICE=WMS
&REQUEST=GetLegendGraphic
&VERSION=1.1.1
&FORMAT=image/png
&LAYER=airports
```

WMS GetFeatureInfo

only for ‘queryable’ WMS (`LAYER queryable="1"`)

```
&SERVICE=WMS
&REQUEST=GetFeatureInfo
&EXCEPTIONS=text/html
&BBOX=97.3,5.6,105.6,20.4
&INFO_FORMAT=text/plain
&QUERY_LAYERS=forest
&WIDTH=600
&HEIGHT=800
&X=100
&Y=110
```

```
GetFeatureInfo results:

Layer 'forest'
  Feature 641:
    OBJECTID = '642'
    FOR_TYPE = 'NF'
    YEAR_GAZ = '0'
    Shape_area = '8.88578688327e+000'
    Shape_len = '1.95388446283e+002'
```

Portrayal specifications: SLD

Styled Layer Descriptors (SLD):

- XML format to allow user-defined symbolization
- Can be used to portray the output of WMS, WFS & WCS

RIMapperWMS: Overview

- Why SVG for a Web Mapping Service?
- Why a built-in GUI?
- Past: Predecessor projects
- Present
  - Principles
  - Technicalities
- Future: Outlook

What is a Web Mapping Service?

- A web service interface specification by the Open Geospatial Consortium (OGC)
- OGC delivers spatial interface specifications for Open Web Services (OWS) & related Encodings:
  - Geographic Markup Language (GML)
  - Web Catalog Service
  - Web Feature Service
  - Web Coverage Service
  - **Web Mapping Service**
  - Styled Layer Descriptor
  - Web Map Context Document

What is a Web Mapping Service?

"Standardized interface for the creation of super-imposed map-like views of geographic information"

- Delivers map graphics from standardised URL requests
- WMS is actually the most mature and widest adopted OWS specification (numerous open source, as well as commercial solutions)

Why Scalable Vector Graphics for a WMS?

SVG is XML-based vector graphics

- High quality (carto)graphics & attribute info
- low-bandwidth well suited for mobile applications

Many WMS exist, some with (limited) SVG

- All treat SVG as ‘static graphics format’ only
- SVG also can hold attribute data
- SVG also can provide animation
- SVG also can provide application logic

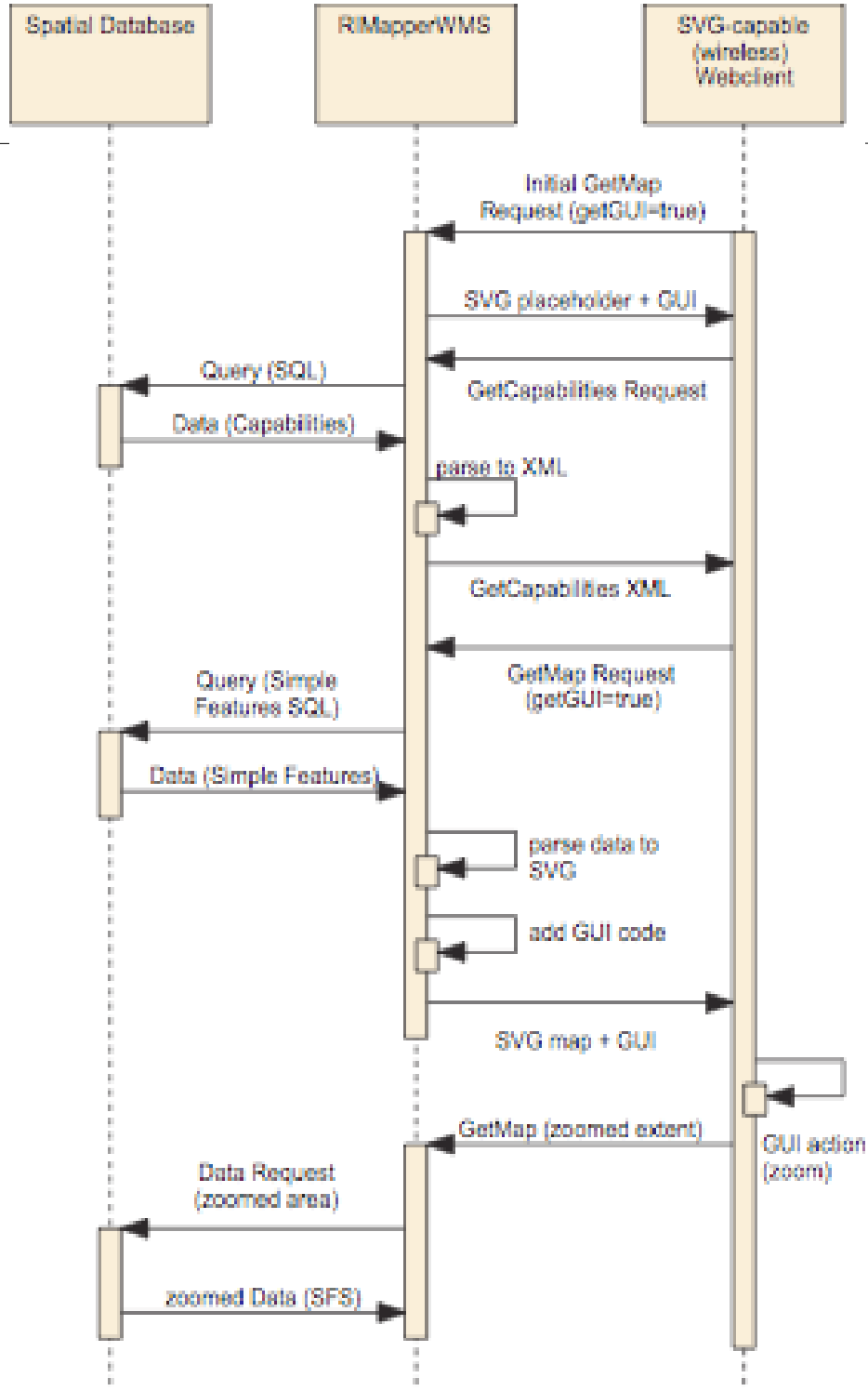
  ➔Can support built-in Graphical User Interface (GUI)

Why a built-in GUI?

No need for separate client application:

"output = application"

- simple WMS conformant interface to the data
- data includes built-in client-side GUI
- GUI handles the map interaction and generates further requests



Past: Predecessor projects

RIMapperWMS has “organically grown” out of a range of earlier project at ITC:

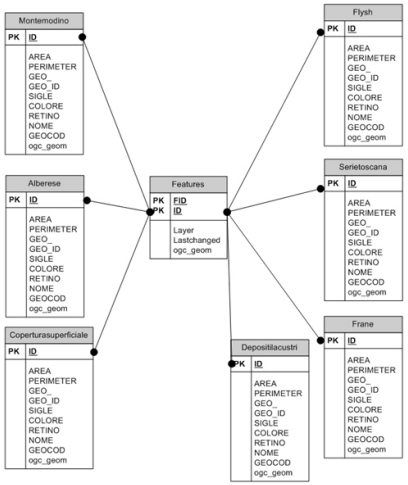
- RIMapper
- FLAVOUR (part of Wireless Campus LBS)
- Campusmapper

…all of these are under the umbrella of the SDI^{LIGHT} programme

SDI LIGHT

- Lightweight Spatial Data Infrastructure based on open standards/open source software
- testbed/playing ground at ITC
  - for research, PhD & MSc work
  - for projects & proof-of-concept applications
- server-side focus on PostgreSQL/PostGIS, Java, open source OWS services
- client-side focus on OpenLayers and SVG

RIMapper: Risk Inventory Mapper



Java servlets to deliver SVG output (=application)

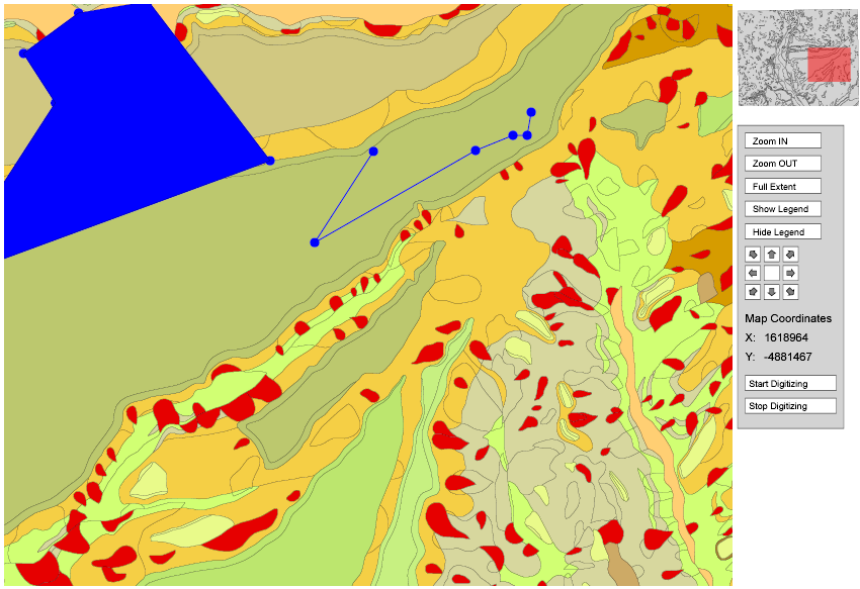
makeSVG

XML2SVG

parseXML

XML - configuration

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE RIM PUBLIC "" "/RIMapper/XML/RIM.dtd">
<RIM TYPE="SVG_STANDALONE" DB="rimapper" UN="un" PW="pw">
<HEADER>
    <FRAGMENT DBID="default" NAME="root" TYPE="SVG_ROOT"/>
    <STYLES>
    <STYLE DBID="default" NAME="defPoint" TYPE="CSS"/>
    <STYLE DBID="default" NAME="defLine" TYPE="CSS"/>
    <STYLE DBID="default" NAME="defArea" TYPE="CSS"/>
    </STYLES>
    <FRAGMENT DBID="default" NAME="init" TYPE="ECMASCRIPT"/>
    <FRAGMENT DBID="default" NAME="show" TYPE="ECMASCRIPT"/>
</HEADER>
<LAYERS>
  <LAYER DBID="default" NAME="ward" STYLETYPE="single"
        STYLE="defLine" />
  <LAYER DBID="default" NAME="river" STYLETYPE="single"
        STYLE="defArea" >
    <ACTION TYPE="simple" NAME="showRIM" SCOPE="feature"
        EVENT="onclick" PARAMS="evt, '', 'id'"/>
  </LAYER>
  <LAYER DBID="default" NAME="roads" STYLETYPE="single"
        STYLE="defArea" ATTRIBS="type" />
  <LAYER DBID="default" NAME="build" STYLETYPE="single"
        STYLE="defArea" />
</LAYERS>
<FOOTER/>
</RIM>
```



Wireless CampusLBS

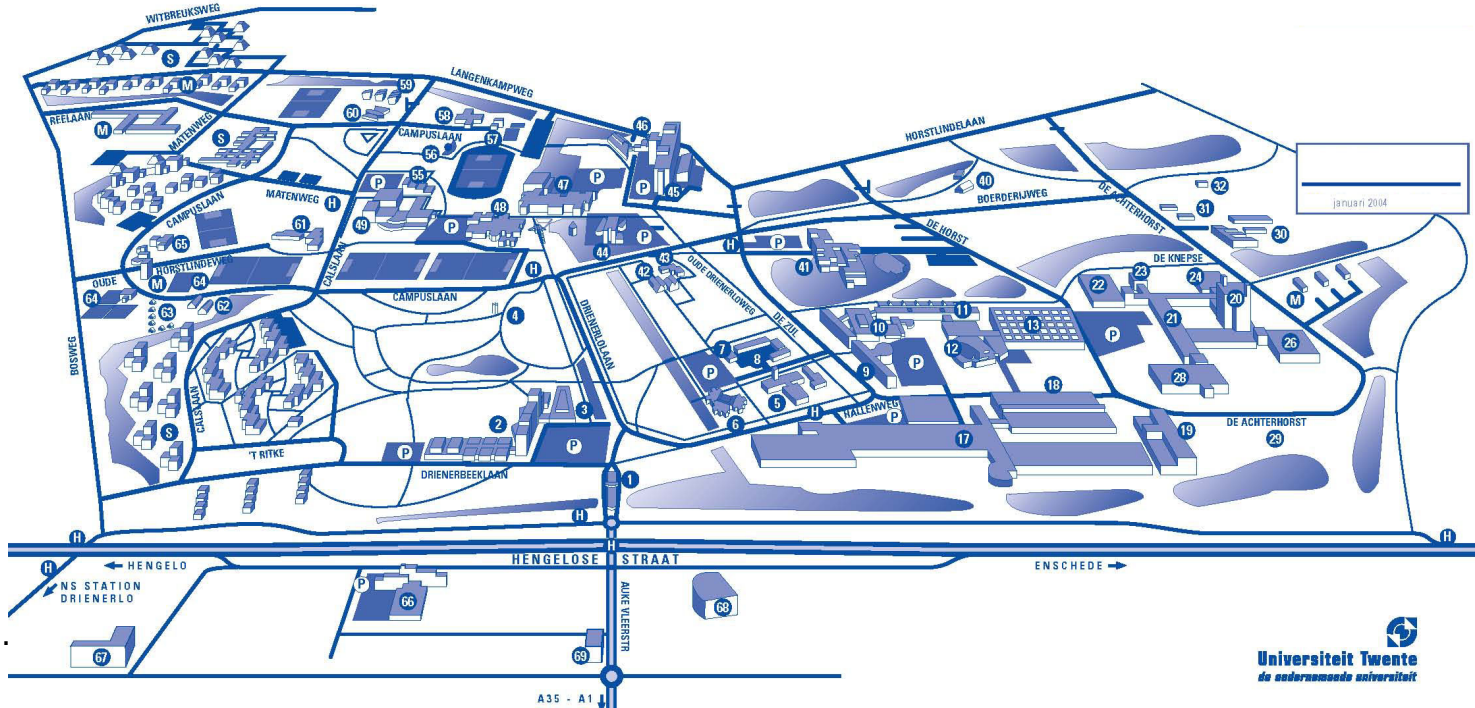
- co-operation between ITC & University of Twente
- to set up ***infrastructure*** necessary for Campus Location Based Services, pilot at ***SVGopen2005***

Europe's largest uniform hotspot

- 140 ha campus (covered in- and outdoors)
  + Enschede city centre (outdoors)
- 650+ individual access points

testbed for wireless

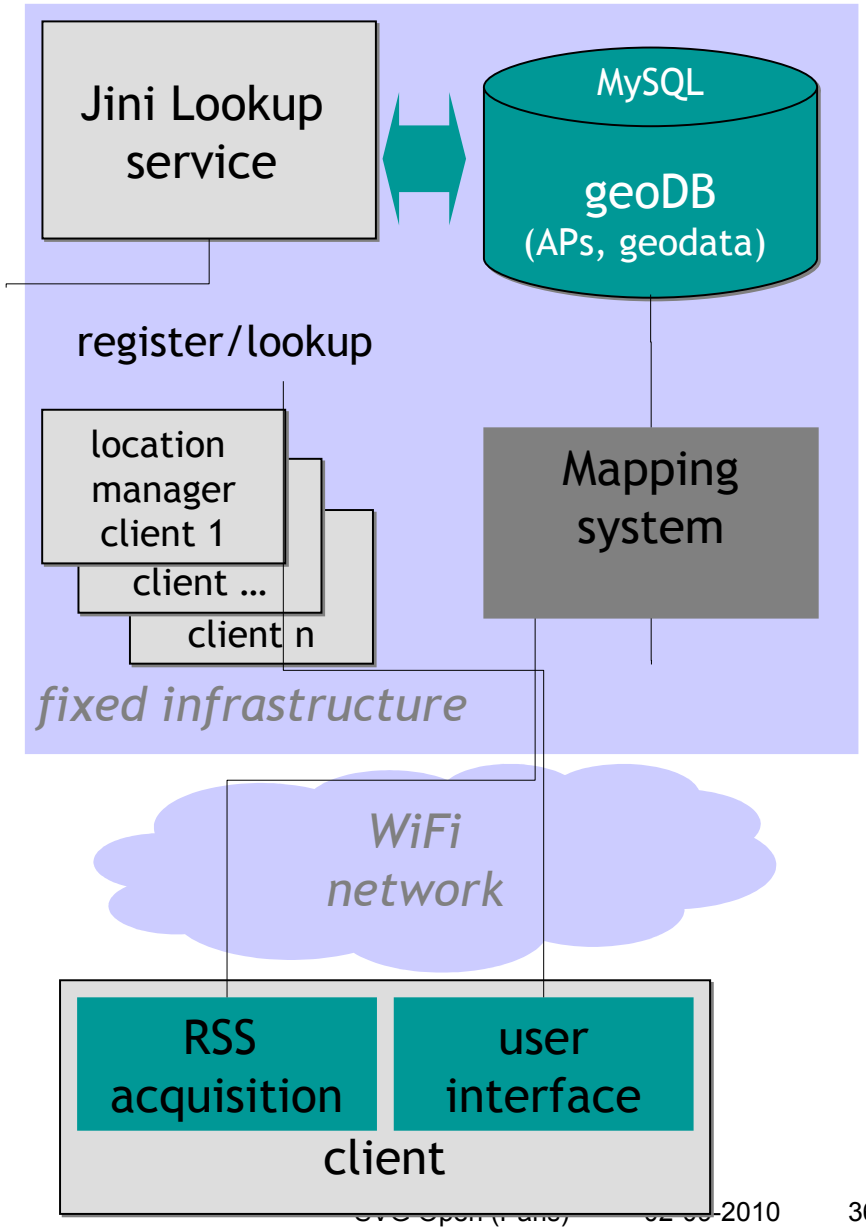
and mobile applications



FLAVOUR prototype: architecture

Friendly Location-aware conference Assistant with priVacy Observant architectURe

- Location Managers
  - provide client with location
  - register with:
- Jini Lookup Services:
  - ‘pull’ (find others, locate resources)
  - ‘push’ (communicate with others, conference messages)
- Client application
- Mapping System based on RIMapper



From Flavour to CampusMapper

- Flavour mapping system based on RIMapper with addition of extent-based feature extraction
- useful for more than Wifi localization:
  - basis for quickly and easily customised maps of the UT Campus

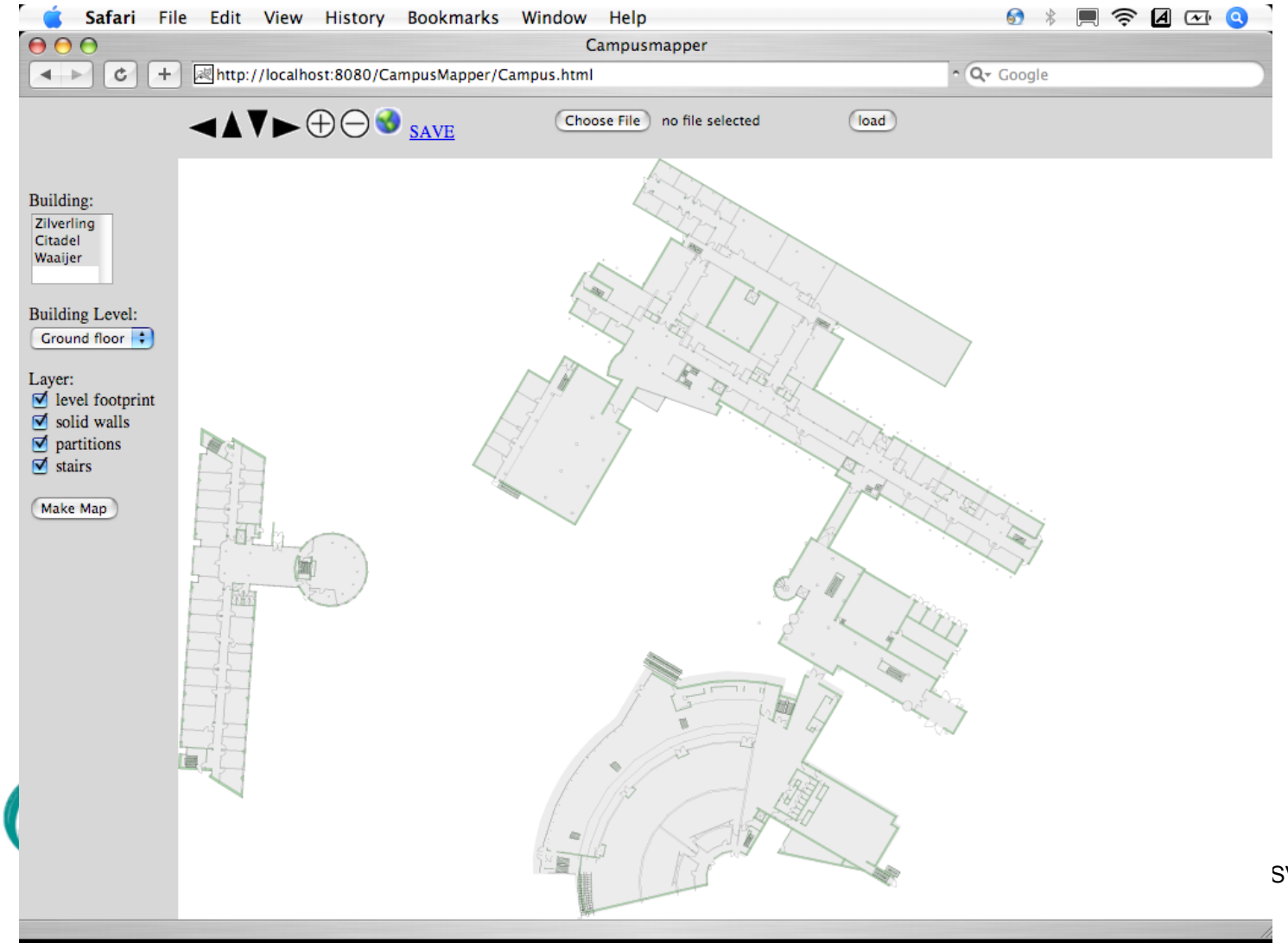
➔ CampusMapper pilot

  - DHTML interface generates GET/POST requests
  - JavaBeans store user/session settings

From CampusMapper to RIMapperWMS

CampusMapper already ‘almost’ an OGC WMS

- Only OGC compatible request/response missing



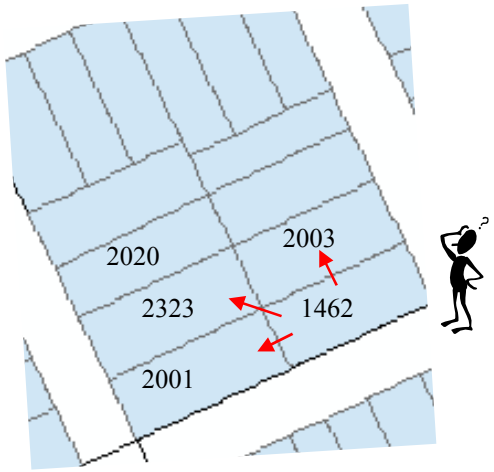
General setup of RIMapperWMS

- spatial database back-end (postGIS)
  - spatial and attribute data
  - Web Mapping Service configuration
- server application (Java)
  - responds to WMS compliant requests
  - provides output in SVG (with built-in GUI)
- mobile or desktop web client
  - renders interactive & dynamic SVG maps

Spatial databases:
Spatial object storage and querying

Rolf A. de By
<deby@itc.nl>

Spatial DBMS



Who are the owners younger than 40 year of the parcels adjacent to the parcel with location number 1462?

'Adjacent' is a spatial property and can not be determined from the thematic data!

A database that combines spatial data with thematic data is called a *spatial database*.

A *spatial DBMS* will also provide special functions for exploring spatial relationships such as 'area', 'buffer', 'distance', 'adjacency', etc.

Spatial DBMS

"Find all the Indian restaurants within 2 km of my hotel":

```
SELECT R.name
  FROM Restaurants AS R,
       Hotels AS H
 WHERE R.type = 'Indian'
   AND H.name = 'Hilton'
   AND Intersect(R.Geometry,
       Buffer(H.Geometry, 2))
```

- The red part creates a spatial join between restaurants and hotels.
- 'Geometry' carries the spatial data.
- 'Intersect' and 'Buffer' are spatial operators.

A *Spatial DBMS* will provide:

- Spatial data types
- Spatial indexing
- Spatial join
- Spatial operators

all these will be integrated in the relational model.

GIS and DBMS support

GIS software provides support for:

- spatial data
- thematic (or attribute) data

DBMSs are much better in table functionality.

Nowadays, GIS applications make use of external DBMSs for spatial and thematic data support.

One implementation: PostGIS extension of PostgreSQL

PostgreSQL: an object-relational database management system (ORDBMS)

- Open-source
- Supports large part of SQL:
  - complex queries
  - foreign keys
  - triggers
  - views
  - transactional integrity
  - multiversion concurrency control
- Can be extended in many ways, by adding new:
  - Datatypes
  - Functions
  - Operators
  - aggregate functions
  - index methods
  - procedural languages
- And because of the liberal license, PostgreSQL can be used, modified, and distributed by everyone free of charge for any purpose, be it private, commercial, or academic.

One implementation: PostGIS extension of PostgreSQL

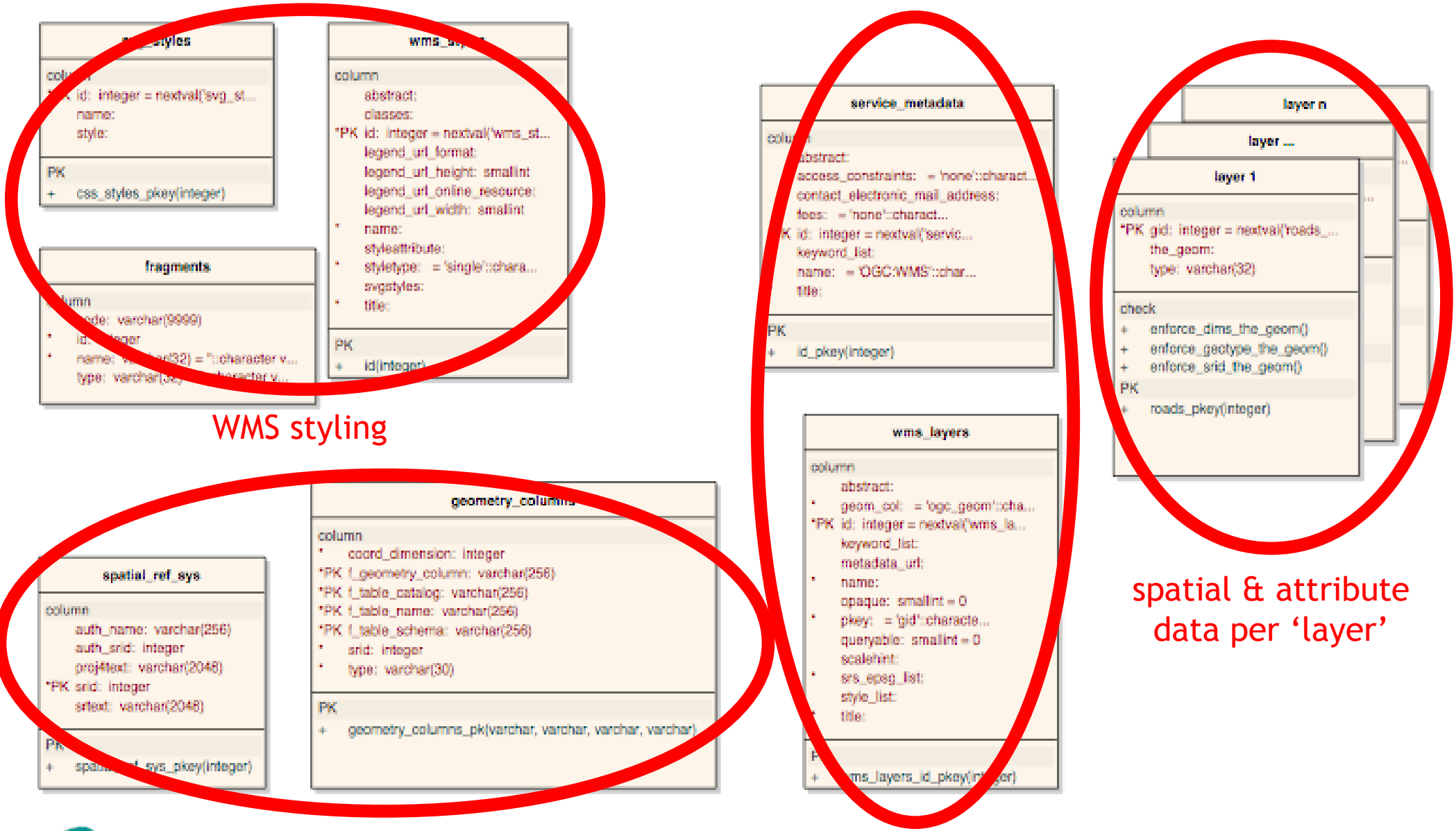
PostGIS is an extension to the PostgreSQL object-relational database system

Allows OGC Simple features objects to be stored in the database

PostGIS includes support for:

- GiST-based R-Tree spatial indexes
- functions for analysis of OGC geometries
- functions for processing of OGC geometries

spatial database back-end (PostGIS)



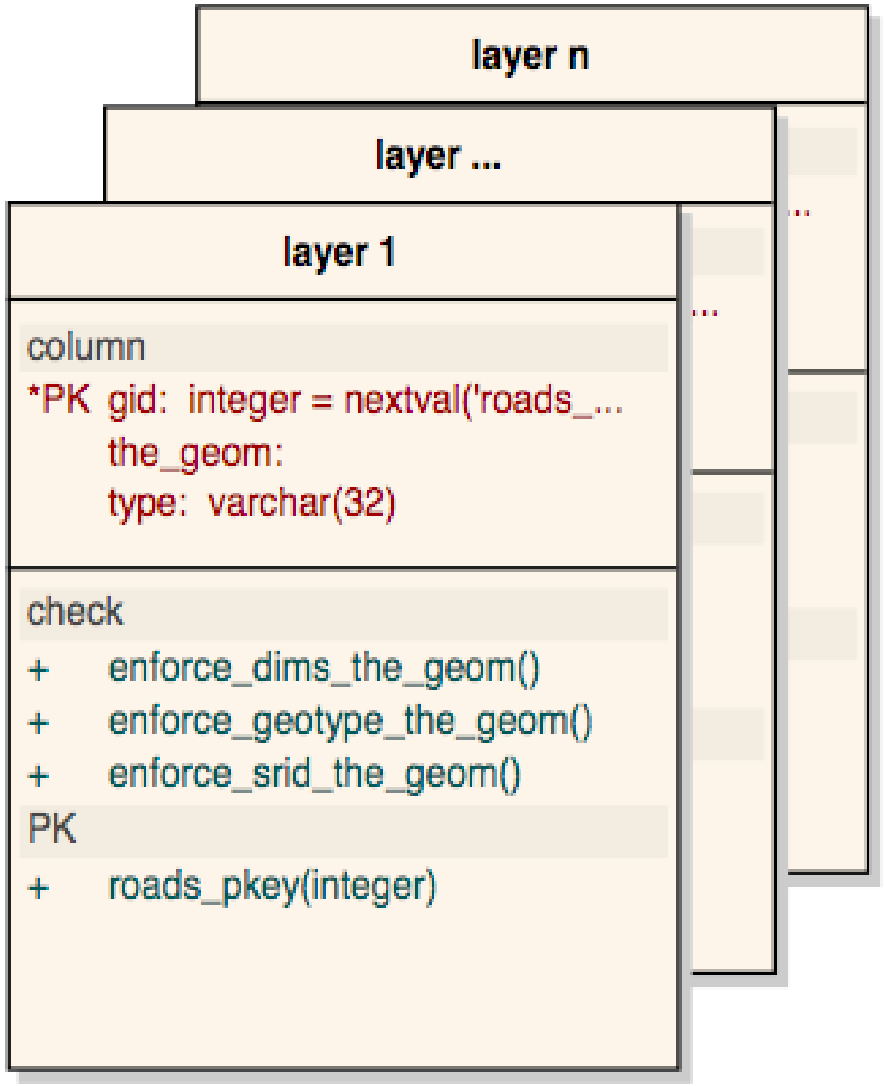
Spatial data layer tables

Object geometries in PostGIS GEOMETRY objects

- follows OGC Simple Features Specification
- spatially indexed
- (re-)projectable

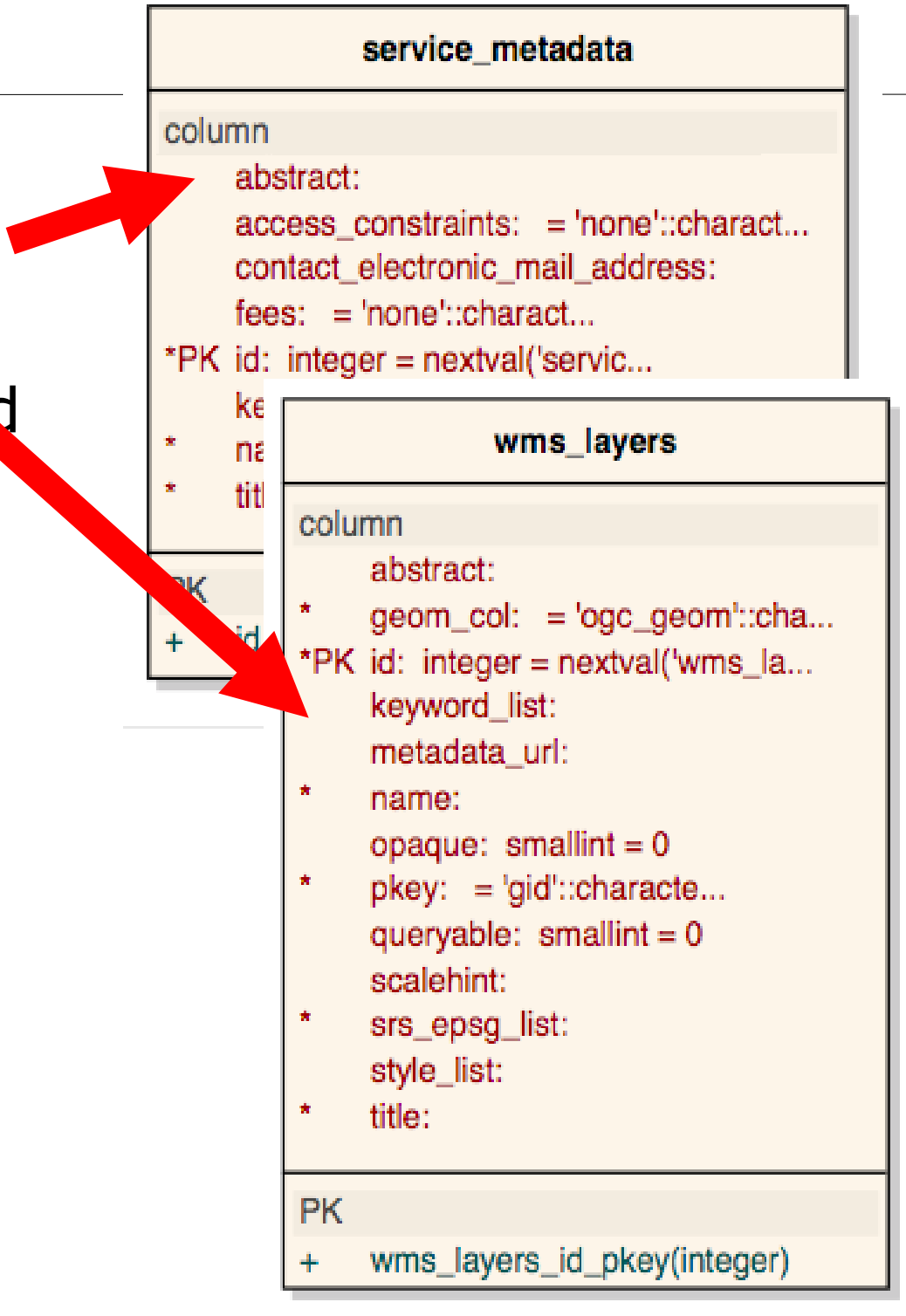
Object attributes

Can come from many data sources (eg. shp2pgsql)



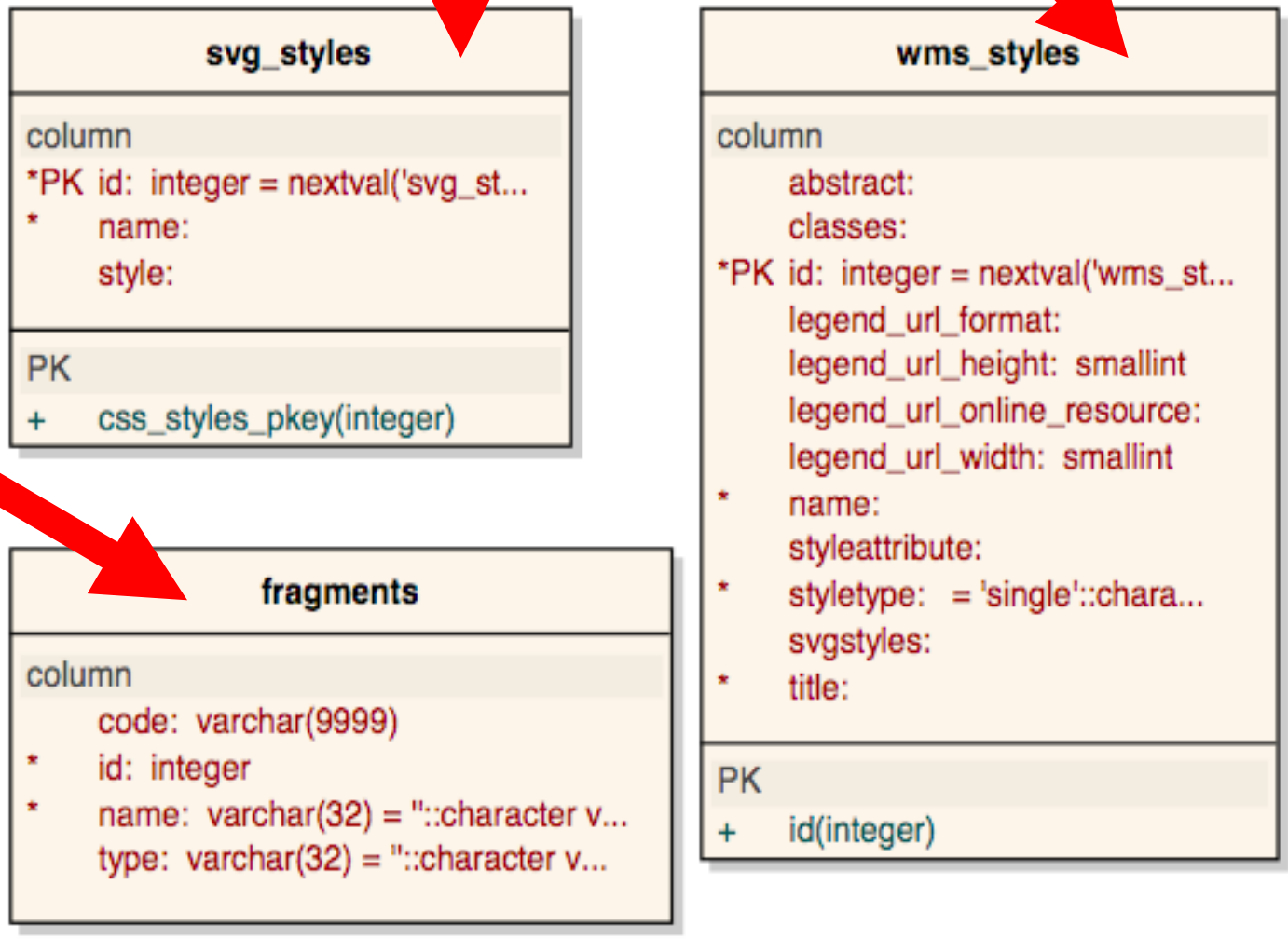
WMS metadata tables

- Defines the WMS instance metadata
- Lists available layers and their:
  - projection data
  - extent
  - styles
  - etc…



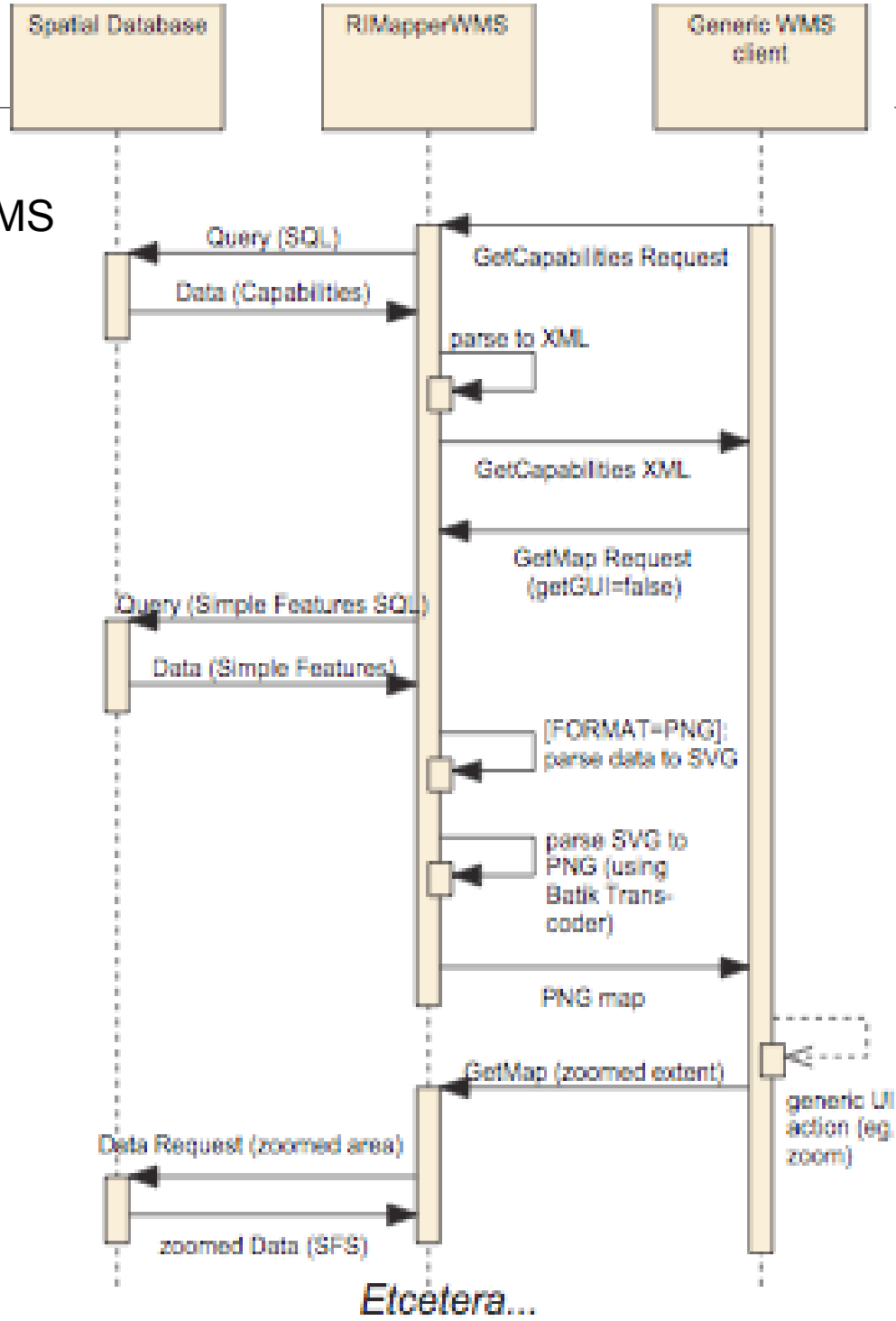
WMS styling tables

- Defines available styles from WMS perspective
- Defines underlying SVG graphic styles
- Multi-purpose table for SVG & script fragments (eg. GUI elements, interactivity event handlers, …)



Interoperability considerations

- GetGUI=true would break a cascading WMS
  - Default GetGUI=false
- Other output formats support needed
  - At least GIF & PNG
  - planned through Batik transcoding



Status: first public bèta released

- Adheres to OGC WMS *Basic* 1.1.1 specification
- Supports `GetCapabilities` & `GetMap` requests
- Additional vendor-specific `getGUI` capability
- Known limitations & issues:
  - GUI client very limited, need to make GUI more complete (layer switcher, attribute info, etc…) and more flexible (support more User Agents & SVG 1.2)
  - `getGUI=false` supported , but not yet output of formats other than SVG (PNG, GIF, etc…)
  - most OGC Compliance Tests pass, but no full compliance (ao. PNG or GIF output needed)
- Free, open source (*creative commons* license)

Outlook

Immediate plans:

- extending to *Queryable* WMS compliance
  - already possible to see attributes (client-side)
  - add server-side support: `GetFeatureInfo` interface
- WMS setup application for Database
- adding transcoding to other formats (PNG, GIF,…)
- performance & useability testing

and further…?

- WMS 1.3.0 support (depends on Proj4 library)
- Styled Layer Descriptor & Web Map Context
- …?

The WORKshop part:

- Copy INSTALL_{your OS} directory
- Install (described on the RIMapper website, in /background materials/):
  1. PostgreSQL
  2. PostGIS
  3. PgAdminIII (on Mac, other installers have it included)
  4. Apache Tomcat
- Deploy the RIMapper Java application (RIMapper.war)
- Set up the example database schema
- Try it on http://localhost:8080/RIMapper/testURL.html

- If time permist we'll do some editing:
  - changing a 'single' style
  - adding a 'chrorochromatic' style
  - ...