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
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
The document contains diagrams illustrating the spatial database back-end (PostGIS). The diagrams show the relationships and data structures for WMS styling, PostGIS spatial metadata, and WMS metadata. The focus is on how spatial and attribute data is managed per 'layer' in the database.
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, interaction event handlers, ...)

```
svg_styles

<table>
<thead>
<tr>
<th>column</th>
</tr>
</thead>
<tbody>
<tr>
<td>*PK id: integer = nextval('svg_styles')</td>
</tr>
<tr>
<td>name: style:</td>
</tr>
<tr>
<td>PK</td>
</tr>
<tr>
<td>+ css_styles_pkey(integer)</td>
</tr>
</tbody>
</table>

wms_styles

<table>
<thead>
<tr>
<th>column</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstract:</td>
</tr>
<tr>
<td>classes:</td>
</tr>
<tr>
<td>*PK id: integer = nextval('wms_styles')</td>
</tr>
<tr>
<td>legend_url_format:</td>
</tr>
<tr>
<td>legend_url_height: smallint</td>
</tr>
<tr>
<td>legend_url_online_resource:</td>
</tr>
<tr>
<td>legend_url_width: smallint</td>
</tr>
<tr>
<td>name: styleattribute:</td>
</tr>
<tr>
<td>* styletype: = 'single':chara...</td>
</tr>
<tr>
<td>svgstyles:</td>
</tr>
<tr>
<td>* title:</td>
</tr>
<tr>
<td>PK</td>
</tr>
<tr>
<td>+ id(integer)</td>
</tr>
</tbody>
</table>
```

fragments

```
column
    code: varchar(9999)
* id: integer
* name: varchar(32) = "::character v... type: varchar(32) = "::character v...
```
Status: 1.1

• Adheres to OGC WMS Basic 1.1.1 specification
• Supports GetCapabilities & GetMap requests
• **TIME** parameter supported: data can be selected based on time extent, as well as spatial extent
• Additional vendor-specific `getGUI` capability
• TimeMapper code included to offer SMIL animated SVG for point type data
• Known limitations & issues:
  ▪ Output formats other than SVG (png, jpeg) not possible in combination with `GetGUI=true`.
  ▪ External (cascaded) WMS layers are loaded in the background, without checking for success or progress report
  ▪ Extension to ISO 8601:1988(E) not fully implemented: Only contiguous TIME periods (datetime_begin/datetime_end) for now.
  ▪ Animation GUI timeslider only works in Opera and FireFox (basic animation works also in Safari and other WebKit based browsers).
• Free, open source (**creative commons** license)
Outlook

Near future plans:
- WMS setup application for Database
- WMS 1.3.0 support (depends on Proj4 library)

and further…?
- More animation types
- Text labelling
- Styled Layer Descriptor & Web Map Context
- …?
The WORKshop part:

- Install (described on the RIMapper website):
  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](http://localhost:8080/RIMapper/testURL.html)

- If time permist we’ll do some editing:
  - changing the animation style settings
  - changing a ‘single’ style
  - adding a ‘chorochromatic’ style
  - ...

UNIVERSITY OF TWENTE.