The Migmap project: technical aspects

New Trends in e-Humanities, 29 November 2012

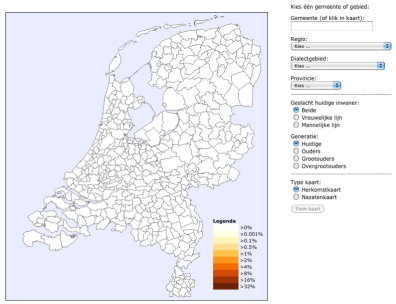

```
{"gemeentepercentages" : {
  "1" : {
    "data" : {
      "g_0003" : "0.0004",
      "g_0005" : "0.0008",
      "g_0007" : "0.0001",
      "g_0009" : "0.0003",
      "g_0010" : "0.0018",
      "g_0014" : "0.0207",
      "g_0015" : "0.0010",
      "g_0017" : "0.0004",
      "g_0018" : "0.0020",
      "g_0022" : "0.0028",
      "g_0024" : "0.0004",
      "g_0025" : "0.0021",
      "g_0034" : "0.0014",
      "g_0037" : "0.0015",
      "g_0039" : "0.0003",
      "g_0040" : "0.0005",
```


Jan Pieter Kunst, Meertens Institute

CLARIN NL

General architecture of the application

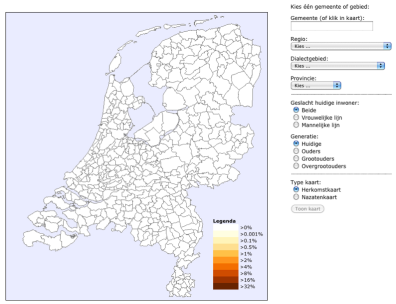
General architecture of the application



Migmap application

General architecture of the application

1. Request for data for location

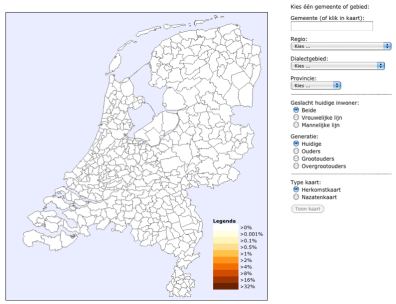


Migmap application

Migration data web service

General architecture of the application

1. Request for data for location



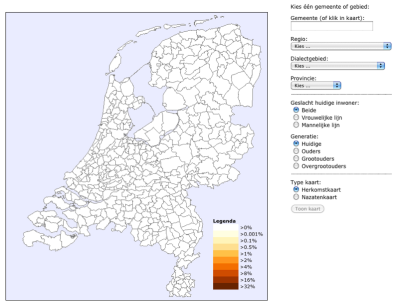
Migmap application

2. JSON document with data for location

Migration data web service

General architecture of the application

1. Request for data for location



Migmap application

2. JSON document with data for location

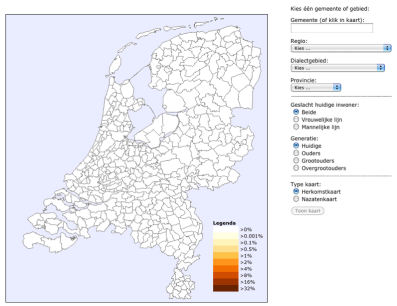
3. Request for map of data from 2.

Migration data web service

Mapping web service
(Meertens "Kaart" module)

General architecture of the application

1. Request for data for location



Migmap application

4. Map images

2. JSON document with data for location

3. Request for map of data from 2.

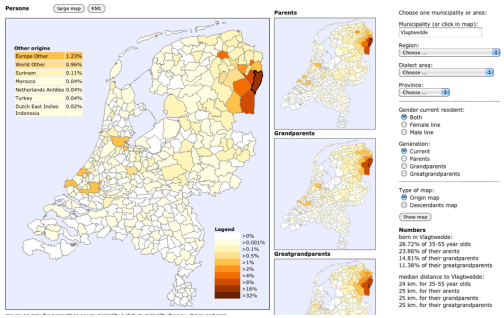
Migration data web service

Mapping web service
(Meertens "Kaart" module)

CLARIN NL

General architecture of the application

1. Request for data for location



Migmap application

4. Map images

2. JSON document with data for location

3. Request for map of data from 2.

Migration data web service

Mapping web service
(Meertens "Kaart" module)

All data here is open

All data here is open

- The migration data is directly accessible at http://www.meertens.knaw.nl/migmap/migrationdata/ and available under a Creative Commons license.

All data here is open

- The migration data is directly accessible at http://www.meertens.knaw.nl/migmap/migrationdata/ and available under a Creative Commons license.

- The mapping module is open source (GPL v2) and its web service at http://www.meertens.knaw.nl/kaart/beta/rest/ can be used without restrictions.

CLARIN NL

Aside: performance considerations

Aside: performance considerations

Applications from the Meertens are often very popular with the general public. We need to prepare for that so that our server isn't brought down by too many visitors. In Migmap, we try to accomplish this in two ways:

Aside: performance considerations

Applications from the Meertens are often very popular with the general public. We need to prepare for that so that our server isn't brought down by too many visitors. In Migmap, we try to accomplish this in two ways:

1) Caching as much as possible on the server, to minimize database access (migration data) and computation (map generation)

CLARIN NL

Aside: performance considerations

Applications from the Meertens are often very popular with the general public. We need to prepare for that so that our server isn't brought down by too many visitors. In Migmap, we try to accomplish this in two ways:

1) Caching as much as possible on the server, to minimize database access (migration data) and computation (map generation)
2) Doing computation client-side instead of server-side (using Javascript)

CLARIN NL

Meertens *Kaart* module

One of the main goals of the Migmap project is enriching and extending the existing Meertens mapping module ("Kaart") to make it more usable for developers, also from outside the Meertens Institute.

All planned features are now working, what follows is an overview of the possibilities.

Meertens *Kaart* module

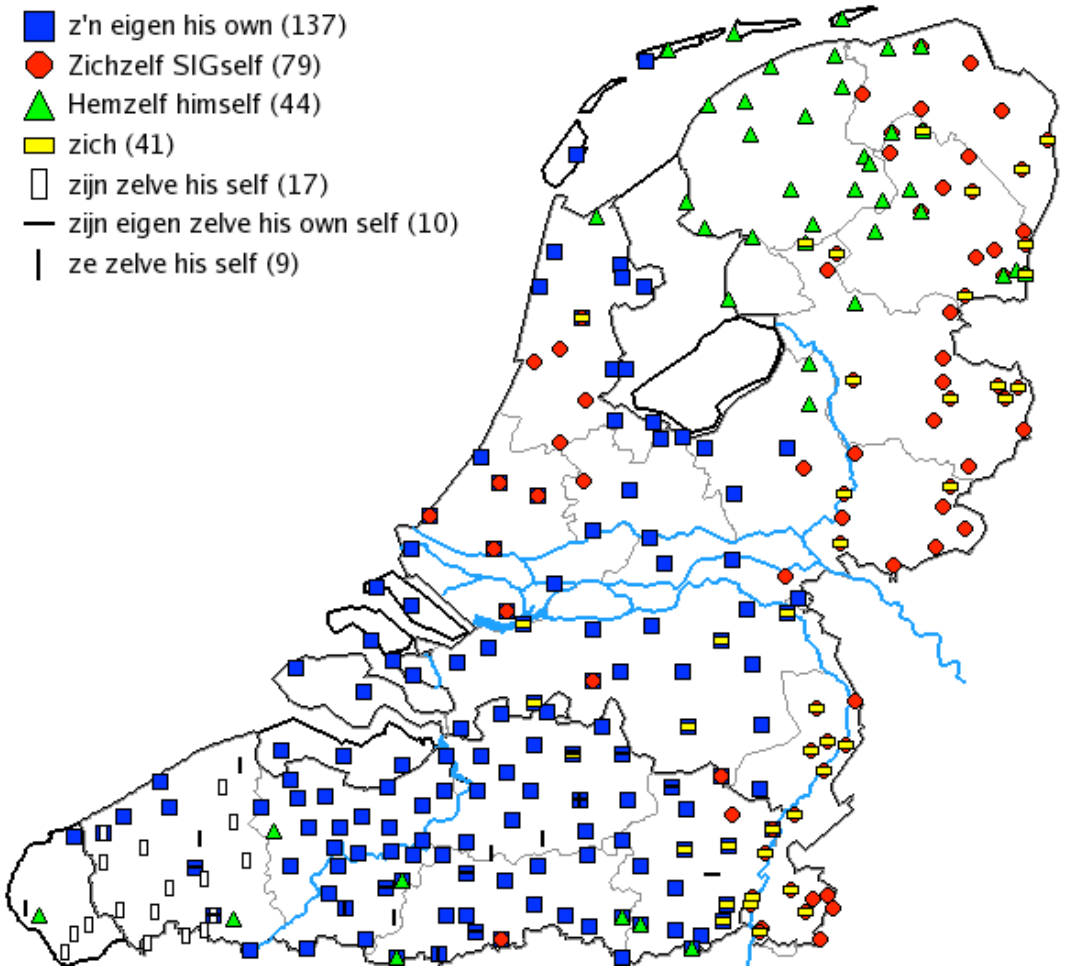
Two kinds of maps can be generated.

1) Maps in which places (cities/towns/villages) in the Dutch language area are shown with user-configurable symbols. Typical use case is visualizing the distribution of different phenomena in a single map. An example:

Meertens *Kaart* module

Two kinds of maps can be generated.

1) Maps in which places (cities/towns/villages) in the Dutch language area are shown with user-configurable symbols. Typical use case is visualizing the distribution of different phenomena in a single map. An example:

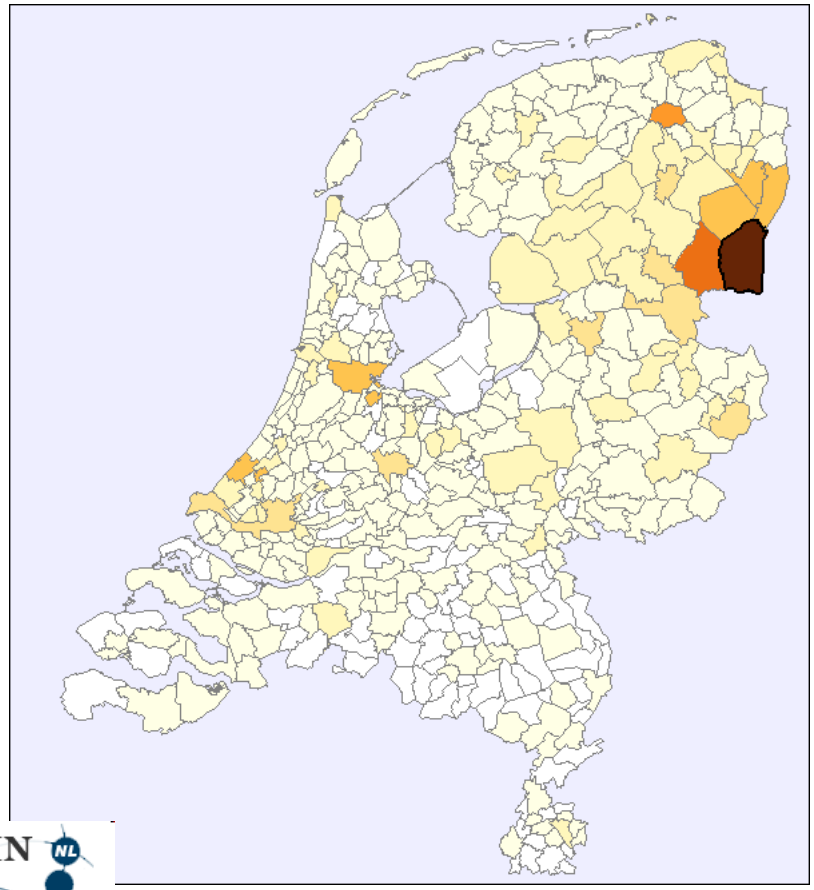


Meertens *Kaart* module

2) "Choropleth" maps in which the distribution of one phenomenon is shown by differently coloured areas. Examples are the migration maps in Migmap, or maps of distribion of last and first names.

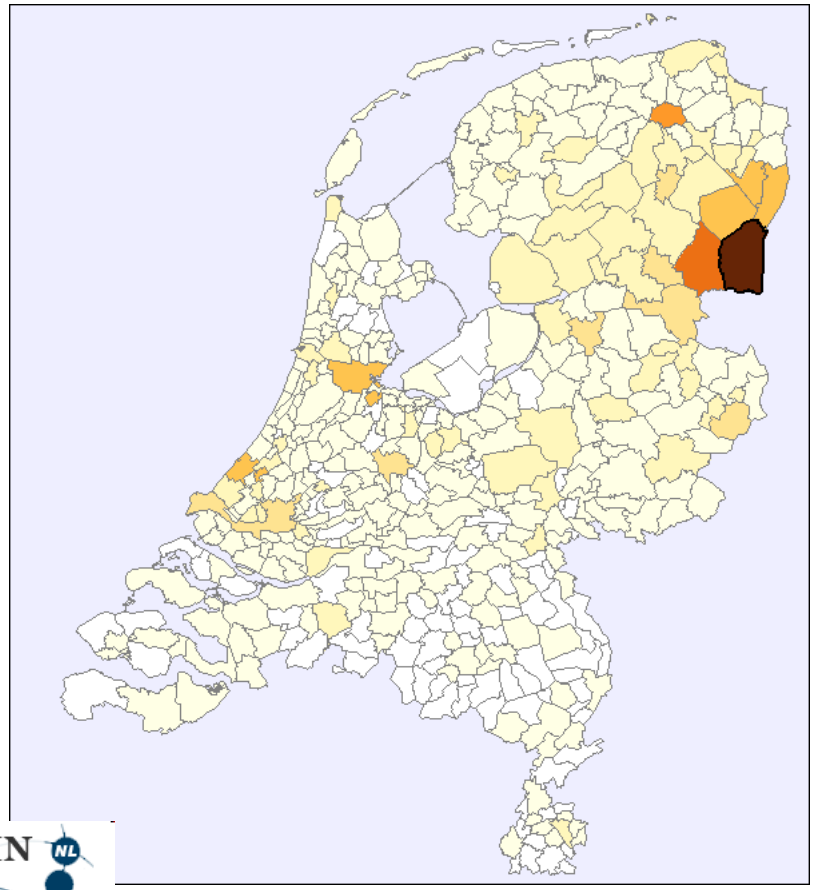
Meertens *Kaart* module

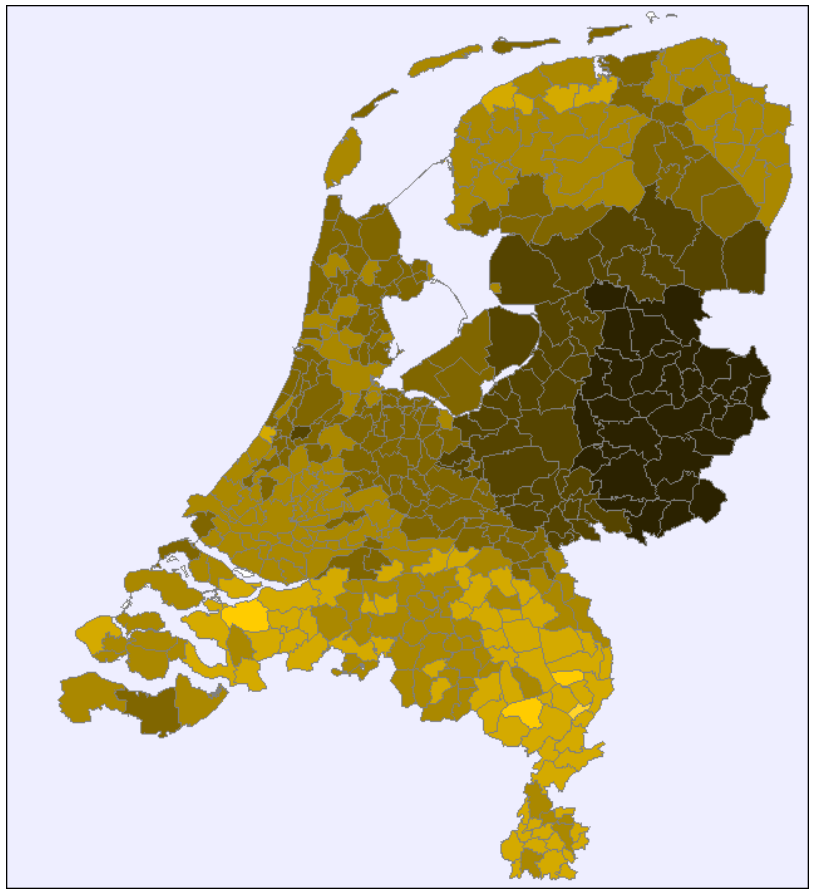
2) "Choropleth" maps in which the distribution of one phenomenon is shown by differently coloured areas. Examples are the migration maps in Migmap, or maps of distribion of last and first names.



Meertens *Kaart* module

2) "Choropleth" maps in which the distribution of one phenomenon is shown by differently coloured areas. Examples are the migration maps in Migmap, or maps of distribion of last and first names.





CLARIN NL

Kaart module: how to use

The *Kaart* module is written in PHP and open sourced under the GPL version 2.

There are four ways for developers to use it.

Kaart module: PHP interface

1) The first way is to download the source code from the Meertens server (currently available is the previous stable version, which does not yet include the choropleth maps; see www.meertens.knaw.nl/kaart) and install it locally. The code is packaged according to the PEAR standard and can be installed via the command line:

```
$ sudo pear install Kaart-2.4.8.tgz
```

After some further configuration (installing the database with place coordinates and editing the config file) the *Kaart* module can be used in PHP code.

Kaart module: PHP interface

Kaart module: PHP interface

Map with symbols:

```
require('Kaart.class.php');

$codes = array('L423p','O275p','O286p','P002p'); // Kloeke-codes for placenames

$kaart = new Kaart();
$kaart->addData($codes);
$kaart->setFullSymbol('square', 6000, 'fill:yellow; stroke:black; stroke-width:500;');
$kaart->show('png');
```

CLARIN NL

Kaart module: PHP interface

Map with symbols:

```
require('Kaart.class.php');

$codes = array('L423p','O275p','O286p','P002p'); // Kloeke-codes for placenames

$kaart = new Kaart();
$kaart->addData($codes);
$kaart->setFullSymbol('square', 6000, 'fill:yellow; stroke:black; stroke-width:500;');
$kaart->show('png');
```

Choropleth map:

```
// official municipality codes, prefixed by 'g_' so that they can be used as id's in HTML
$municipalities = array('g_0534' => '#FFC513', 'g_1740' => '#E30000');

$kaart = new Kaart('municipalities');
$kaart->addData($municipalities);
$kaart->show('png');
```

CLARIN NL

Kaart module: PHP interface

Map with symbols:

```
require('Kaart.class.php');

$codes = array('L423p','O275p','O286p','P002p'); // Kloeke-codes for placenames

$kaart = new Kaart();
$kaart->addData($codes);
$kaart->setFullSymbol('square', 6000, 'fill:yellow; stroke:black; stroke-width:500;');
$kaart->show('png');
```

Choropleth map:

```
// official municipality codes, prefixed by 'g_' so that they can be used as id's in HTML
$municipalities = array('g_0534' => '#FFC513', 'g_1740' => '#E30000');

$kaart = new Kaart('municipalities');
$kaart->addData($municipalities);
$kaart->show('png');
```

Provinces, regions and dialect areas can also be used as areas in a choropleth map: use `$kaart = new Kaart('provinces');` etc.

Kaart module: XML-RPC interface

2) The second way is to use the (pre-Migmap) XML-RPC web service interface for the *Kaart* module. This is more flexible in the sense that nothing needs to be installed locally and the developer does not need to use PHP.

This is the XML-RPC version of the map with two coloured municipalities from slide 10.

(As you can see, XML-RPC is rather verbose.)

The client sends this message to the server and receives back ...

This is the XML-RPC version of the map with two coloured municipalities from slide 10.

(As you can see, XML-RPC is rather verbose.)

The client sends this message to the server and receives back ...

```
<?xml version="1.0" encoding="UTF-8"?>
<methodCall>
    <methodName>kaart.createmap</methodName>
    <params>
        <param>
            <value>
                <string>png</string>
            </value>
        </param>
        <param>
            <value>
                <struct>
                    <member>
                        <name>g_0534</name>
                        <value>
                            <string>#FFC513</string>
                        </value>
                    </member>
                    <member>
                        <name>g_1740</name>
                        <value>
                            <string>#E30000</string>
                        </value>
                    </member>
                </struct>
            </value>
        </param>
        <param>
            <value>
                <struct>
                    <member>
                        <name>type</name>
                        <value>
                            <string>municipalities</string>
                        </value>
                    </member>
                </struct>
            </value>
        </param>
    </params>
</methodCall>
```

Kaart module: XML-RPC interface

... a base64-encode PNG image wrapped in XML:

Kaart module: XML-RPC interface

... a base64-encode PNG image wrapped in XML:


```xml
<?xml version="1.0" encoding="UTF-8"?>
<methodResponse>
   <params>
    <param>
      <value>
        <base64>
iVBORw0KGgoAAAANSUhEUgAAAoAAAALABAMAAADxu958AAAAJ1BMVEXu7v8AAP+lKir//
wAAgAD/AAAAAACAgIAekP+AgID/////xRP/7qobN8FjAAAgAElEQVR4nO29TZLzOK+
mnaOO4C6/Ua/HU7LPoKbUDrrfFZyIs6jPIgngxg9lOW3ZsisZVU/asiSLl0ESAEHw53//
lYfKz//381ceKP/rD+Bj5Q/gg+UP4IPlD+CD5Q/gg+VfDzA/eP2/HWBaHiT4bwdYarqsf9Ml
/e4G/3qAS24iWGq+/OoG/3aAeekES/75HcF/O8BrH9jA5bUV/+YGfwAvP7/s/Xr5twP8yX+
j8K9LWhtvKY/d5N8MsNTLb3s+
Kf9qgI9aIWv5dwLMee36nsHv3wjwCm9ZHrXguPxrAGYuV3r10bFXyr8I4CLlwYEDy78G4I8A
zL80e8PybwGYkvB76o3/LQAv5Yj2+/PvAUgNOC/1uTf+foC9 [...]
        </base64>
      </value>
    </param>
  </params>
</methodResponse>
```

Kaart module: REST interface

3) A more user-friendly way is to use the new (developed in Migmap) REST interface for the *Kaart* module.

This works by sending standard HTTP GET and POST request to a URL. Currently the address is:

http://www.meertens.knaw.nl/kaart/beta/rest/

Kaart module: jQuery plugin

4) Finally, the fourth way to use the *Kaart* software is the jQuery plugin. This is a client-side Javascript function, dependent on the jQuery library, which interacts with the REST interface and makes it easy to use maps on a web page. It is used in the Migmap application.

Basic usage is as follows:

Kaart module: jQuery plugin

```
<script type="text/javascript" src="jquery-1.7.2.min.js"></script>
<script type="text/javascript" src="jquery.kaart.js"></script>

var kaartdata = {
  "type" : "municipalities",
  "width" : "600",
  "format" : "png",
  "data" : {
    "g_0153" : "#993404",
    "g_0164" : "#EC7014"
  }
};

$('#kaart').kaart(kaartdata);


<p id="kaart"></p>
```

CLARIN NL

Thank you for your attention

Any questions?