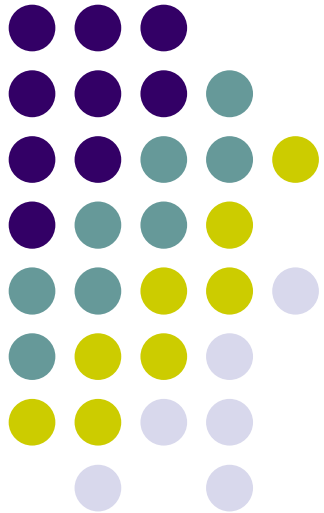
QGIS WMS server – QGIS goes web

INSTITUTE of CARTOGRAPHY
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



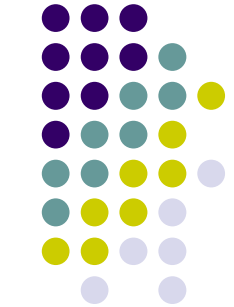
Dr. Marco Hugentobler
Ionuţ Iosifescu-Enescu
Prof. Dr. Lorenz Hurni

Institute of Cartography, ETH Zurich

Overview

- Motivation
- The ORCHESTRA project
- Current design and implementation
- Implemented features
- Outlook

Motivation



- Increasing demand for internet based GIS solutions
- Desire to exchange geoinformation between different systems
- Interoperability increased by use of open standards
- Need for an open service architecture for risk management

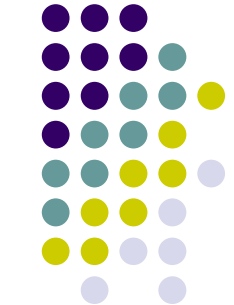
The orchestra project

- Aims to improve the efficiency in dealing with risks by developing an open service-oriented architecture
- The main goal is to improve the interoperability among actors involved in Multi-Risk Management
- Funded by the European Commissions 6th framework program
- Some of ORCHESTRA results offered as input to the INSPIRE and GMES initiatives.

The orchestra project

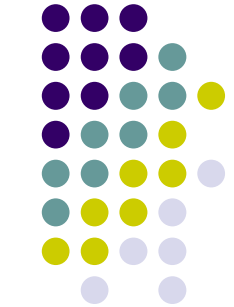
- ORCHESTRA architecture defines abstract and implementation specifications for geospatial services
- ORCHESTRA specifications have roots in open standards (e.g. OGC/ISO)
- QGIS WMS is a part of the ORCHESTRA Map and Diagram Service implementation
- Most service implementations under an open source license

How to transform an existing GIS into a web map server



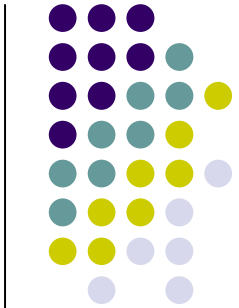
- Get the service specifications
- Write a server application which parses the requests according to the specs
- Link the server application to the GIS libraries
- Map the request parameters to objects already defined in the linked libraries
- Use the libraries to generate maps from the created objects (without opening any GUI!)
- Send the map content back to the client
- Test and debug ... a lot!

Current design and implementation (1)



- CGI executable written in platform independent C++ based on the Qt library
- Could be used together with other server technologies (e.g. Apache dso, FastCGI)
- Uses QGIS libraries and classes for GIS logic and map rendering
- The QGIS WMS classes parse the WMS requests and map them to QGIS objects
- QGIS classes render the map image

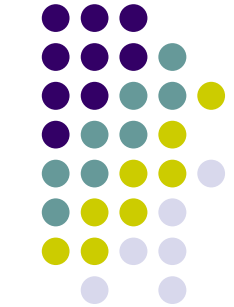
Current design and implementation (2)



- SLD is used as configuration file (similar in purpose to a mapfile)
- SLD is also accepted to allow for user-defined styles

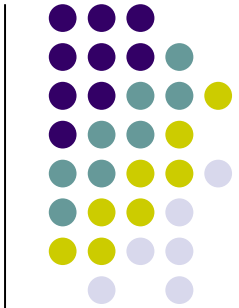
```
<sld:UserStyle>
  <sld:Name>Blue_stroke</sld:Name>
  <sld:FeatureTypeStyle>
    <sld:Rule>
      <sld:PolygonSymbolizer>
        <sld:Stroke>
          - <sld:CssParameter sld:name="stroke">#0000ff</sld:CssParameter>
        </sld:Stroke>
      </sld:PolygonSymbolizer>
    </sld:Rule>
  </sld:FeatureTypeStyle>
</sld:UserStyle>
```

Current design and implementation (3)



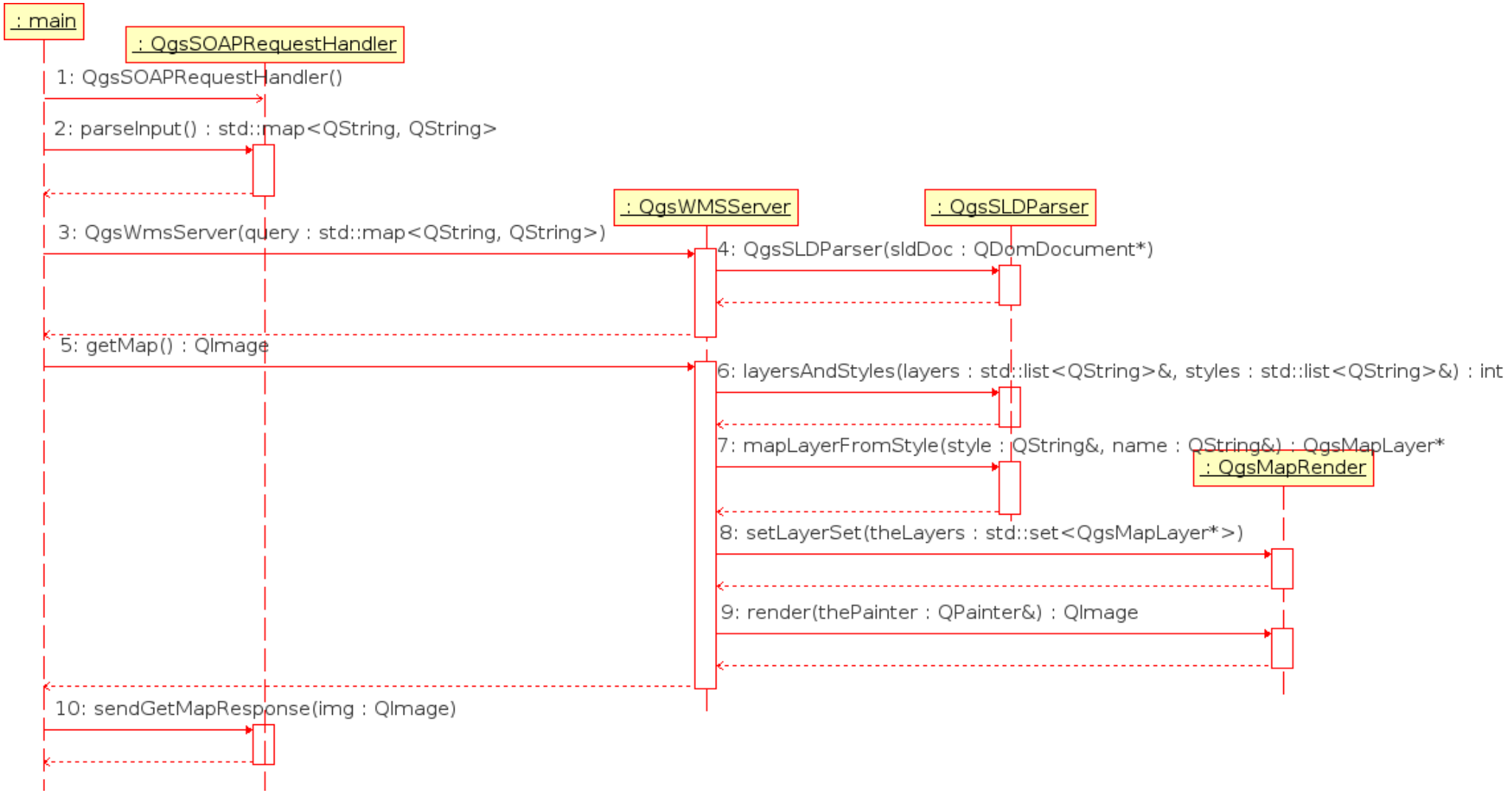
- Main QGIS classes used and their roles
  - QgsMapLayer
  - QgsRasterLayer
  - QgsVectorLayer
  - QgsRenderer
  - QgsMapRender
- QGIS libraries:
  - libqgis_core
  - libqgis_gui
  - libqgis_raster

Current design and implementation (4)



- Main QGIS WMS classes
  - QgsWMSRequestHandler
  - QgsGetRequestHandler
  - QgsSOAPRequestHandler
  - QgsSLDParser
  - QgsWMSServer

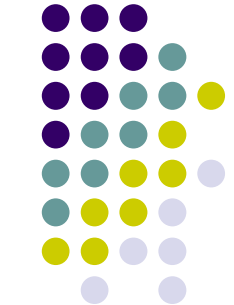
Current design an implementation(5)



Implemented features

- Full implementation of a **basic WMS** (ISO/DIS 19128 – WMS 1.3)
  - GetCapabilities and GetMap operations
- SLD support (not yet all elements)
- Supports a large variety of data formats (shp, GRASS, GML, PostGIS, Interlis, WFS, ...)
- Accepts SOAP requests (specifications developed for the ORCHESTRA project)
- Accepts GML included in a SOAP (or GET) request message as data source

Outlook



- Why another map server?
- New features:
  - SOAP binding
  - Support for thematic mapping (various diagram types, choroplets)
  - Basic user access control with HTTPS
  - Upload layers and styles to the server
  - Share layers and styles with other users
  - Use of cartographic ontologies to make cartographic knowledge available for normal users