Spatial filters implementation for Geo2Tag LBS platform

Mark Zaslavskiy
Vasily Romanikhin
Kirill Krinkin
Open Source & Linux Lab

FRUCT-11, Saint-Petersburg
2012
Three-dimensional mapping services

- Google maps
  - Google maps for mobile in 3D

- Open Street Map
  - OSM-3D

- Nokia maps
  - Nokia maps 3D
Requirements

- Support in the platform of the three geographic coordinates
- Three-dimensional marking of content
- Two-dimensional spatial filters
- Three-dimensional spatial filters
Description of the Geo2Tag LBS platform

Geotag
- description
- reference to the content
- channel
- coordinates (X, Y, Z)
- time

Geo filter
- time
- shape (2D or 3D)
Geo2Tag platform architecture

Internet

HTTP/JSON

Platform Interface

Lighttpd

FastCGI

Control program

PostgreSQL

qSQL

DB

Query processor

Serializers / Deserializer

Spatial filters

Interaction with the database
Sequence of request processing

Platform steps

- Initialize connection to the DB
- Receiving query
- Analysis of the query
- Running query
- Creating response
- Sending response

Used libraries

- qSQL module
- fcgi (work with FastCGI)
- qJSON API (work with JSON)
- QSQL module
- qJSON API (work with JSON)
- fcgi (work with FastCGI)
Platform API

- Registration
- Login
- Add channel
- Subscribe
- Unsubscribe
- Available channels
- Subscribed channels

Core:
- Base API
- Add Geotag

Spatial filtration:
- 2D Geo filtration
- 3D Geo filtration

Service provider:
- Geo filters:
  - "circle"
  - "rectangle"
  - "polygon"

- Geo filters:
  - "cylinder"
  - "box"
  - "geofence"

Marking of content is produced in three geographic coordinates and time.

(client, X, Y, Z, time)
Logical structure of the control program

```plaintext
src/json
- JsonSerializer
  + getjson() : QByteArrayList
  + parsejson(d : QByteArrayList)
- FilterRequestJSON
  - m_shape : FShape
  - m_timeFrom : QDateTime
  - m_timeTo : QDateTime
  - m_altitude1 : double
  - m_altitude2 : double
- FilterDefaultResponseJSON
- FilterCircleRequestJSON
- FilterCylinderRequestJSON

src/service
- server
  + serve()
  + process()
  + run()
- DbSession
  + process(type : QString, body : QByteArrayList) : QByteArrayList
  + processAddNewMarkWithQuery(d : QByteArrayList) : QByteArrayList
  + processFilterCircleQuery(d : QByteArrayList) : QByteArrayList
  + processFilterRectangleQuery(d : QByteArrayList) : QByteArrayList
  + processFilterPolygonQuery(d : QByteArrayList) : QByteArrayList
  + processFilterCylinderQuery(d : QByteArrayList) : QByteArrayList
  + processFilterBoxQuery(d : QByteArrayList) : QByteArrayList
  + processFilterFenceQuery(d : QByteArrayList) : QByteArrayList
  + internalProcessFilterQuery(d : QByteArrayList) : QByteArrayList
- QueryExecutor

src/common
- CompositeFilter
  + filter(d : QList<DataMark>) : QList<DataMark>
  + addFilter(f : Filter)
- Filter
  + satisfy(d : DataMark) : bool
- TimeFilter
- ShapeFilter
- AltitudeFilter
- FShape
  + satisfy(d : DataMark) : bool
- FShapePolygon
- FShapeRectangle
- FShapeCircle

interface for work with DB
```
Spatial filtration. Stack of the filters:

- Spatial data (marked content)
- Composite Filter
  - Time Filter
  - Shape Filter
  - Altitude Filter
- Filtered spatial data
Example of filter query "Cylinder"

POST /service/filterCylinder

```
{  
  "auth_token": "ql4safsaeddwl22ea",
  "time from": "04 03 2011 15:33:47.630",
  "time to": "31 12 2011 15:33:47.630",
  "longitude": 24.8593,
  "latitude": 60.1632,
  "radius": 0.05,
  "altitude_shift":
  {  
    "altitude1": 18.00
    "altitude2": 21.00
  }
}
```

HTTP/1.1 200 OK

```
{  
  "errno": 0,
  "channels":
  [  
    "channel": { ... list of tags ... }
  ]
}
```
Client library for Android OS

- Abstraction from the platform interface
- Provides developers with a convenient programming interface for interacting with the platform

Android
- 53% - part in the world
- Android Market

Android SDK
- ANT
- JDK

Geo2Tag
- Android lib

Developer

Platform objects
- Mark, Channel, etc...

Base platform API

Spatial platform API
- 2D, 3D

Library

Android application

HTTP/JSON

LBSP Geo2Tag
Future plans

● Implement client library for Symbian/MeeGo
● Implement client library for iOS
● Complex spatial objects support and events support
● Platform performance

Project Links

● Project page: http://geo2tag.org
● Project wiki: http://osll.spb.ru/projects/geo2tag/wiki
● Main project repo: https://github.com/OSLL/geo2tag
Questions & Answers

Vasily Romanikhin, vasily.romanikhin@gmail.com
Mark Zaslavskiy, mark.zaslavskiy@gmail.com

Open Source & Linux Lab,
http://osll.furct.org, osll@fruct.org