GeTS project: Geo2Tag LBS Platform Usage in eTourism

Kirill Kulakov

Petrozavodsk State University
Department of Computer Science

AMICT'2015 conference

May 13, 2015, Petrozavodsk, Russia

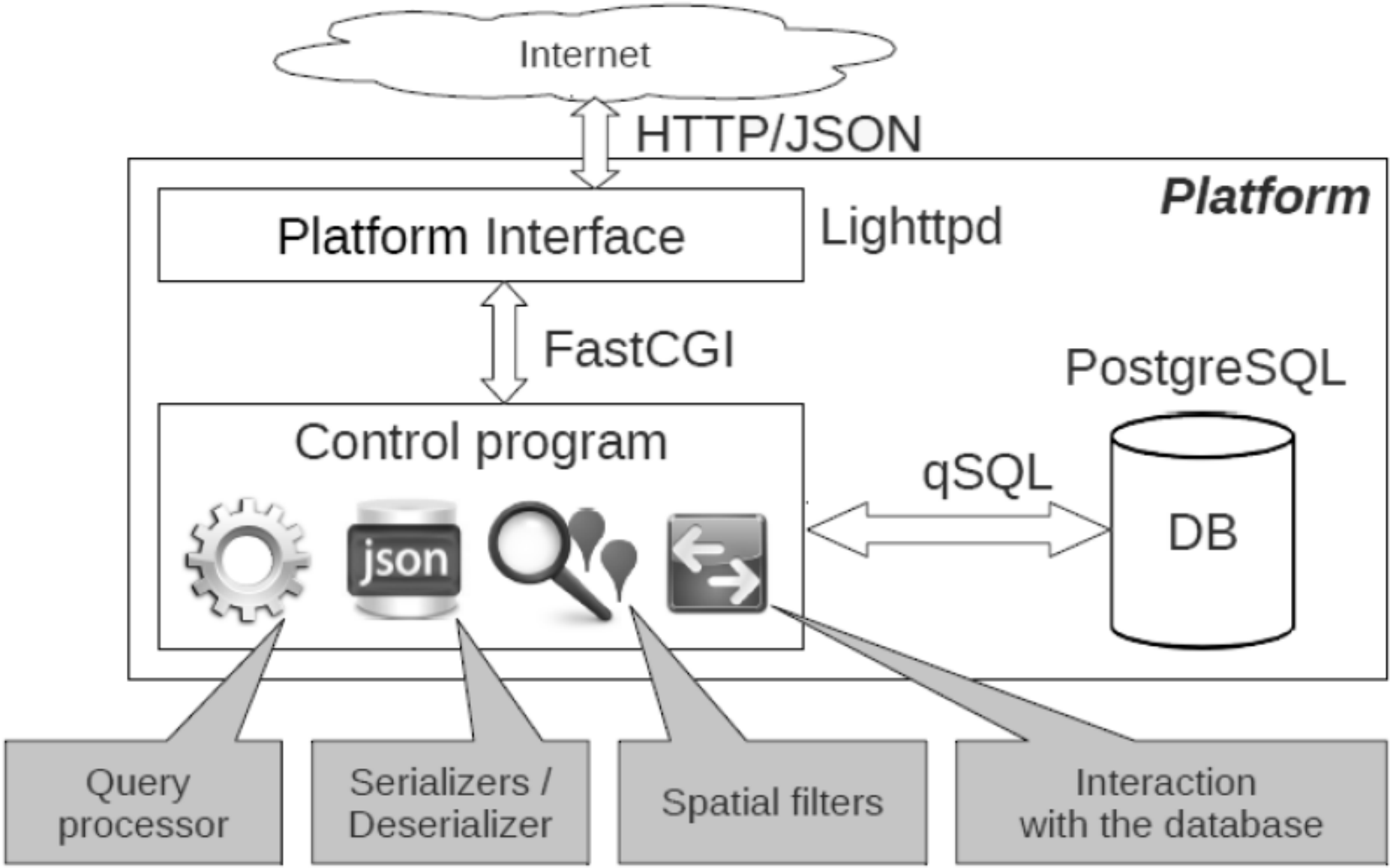
Problem description

- Various touristic and non-touristic applications
- Various geographical objects (point, route, polygon)
- Various object properties
- Object classification (categories)
- User identification without user profile storage
- Anonymous access to public data
- Private space for registered users
- Simple API based on universally accepted formats

Geo2tag platform

- Open-source Location Based Services Platform (http://geo2tag.org)
- Geo2tag objects:
  - Tag – geographical point (coordinates, name, description, uri, time);
  - Channel – sequence of geographical points (name, description, uri);
  - User – any Geo2tag user (e-mail, name, list of subscribed channels).
- Geo2tag functions:
  - User authentication (login, logout);
  - Manage subscribed channels (list, add, remove);
  - Collect tags (list, add);
  - Collect channels (list, add);
  - “2D” requests (Circle, Rectangle, Polygon);
  - “3D” requests (Cylinder, Box, Fence).
  - Other functions (filters, database selection, etc.)

Geo2tag architecture



Proposed list of functions

Required function	Geo2tag implementation
Authentication using OAuth	Internal authentication
User registration	Removed/not implemented
View points, tracks and polygons	View channels and tags
Add, edit and remove points, tracks and polygons	Only add operation are supported
Search query	Various “2D” and “3D” queries
Grouping by categories	Grouping tags in channels
Private and public space	Only subscribed channels
Anonymous access to public space	Not supported
User restrictions	Full access to all data
Various object properties	Name, Description and Link fields with JSON support

Point object implementation

- ”Channel.Name” includes prefix ”ch”, user ID and category ID
- Data properties encoded in JSON format in ”Tag.Description”
- URI properties encoded in JSON format in ”Tag.Url”
- Channel name example: ”ch+28+32” — private channel for user ID “28” and channel “32”
- Tag description example: ”{”index”:”4”,”description”:”some description”,”uuid”:”9db04572-d270-43cf-80f1-1dccd803edf1”}”
- Tag url example: ”{”photo”:”http://gets.cs.petrsu.ru/audio/petrozavodsk/ptz-4.jpg”,”audio”:”http://gets.cs.petrsu.ru/audio/petrozavodsk/ptz-4-ru.mp3”,”url”:”http://ticrk.ru”}”

Track and polygon objects implementation

- ”Channel.Name” includes prefix ”tr” (”pol” for polygons), user ID, system track name and language ID
- ”Channel.Description” includes category ID, visible track name
- ”Tag.Description” includes point index
- Channel name example: ”tr+ivpetboy@gmail.com+Presentation track+en_US”
- Channel description example: ””description”:”Presentation track”,”category_id”:14,”lang”:”en_US”,”hname”:”Presentation track””
- Tag description example: ””description”:”Next pit”,”uuid”:”c1d9961d-7b10-44af-b4af-d6aaf1e1d62f”,”radius”:”4”,”index”:”4””

Category objects implementation

- Separate table ”Category”
- ”Url” field includes links to the full description and icon
- ”Description” field includes various text data
- ”Owner_id” field links category to the project (Super user)
- Any object assigned to the project by the category
- Treelike structure: point separates category levels
- Category name example: ”audio.guide”

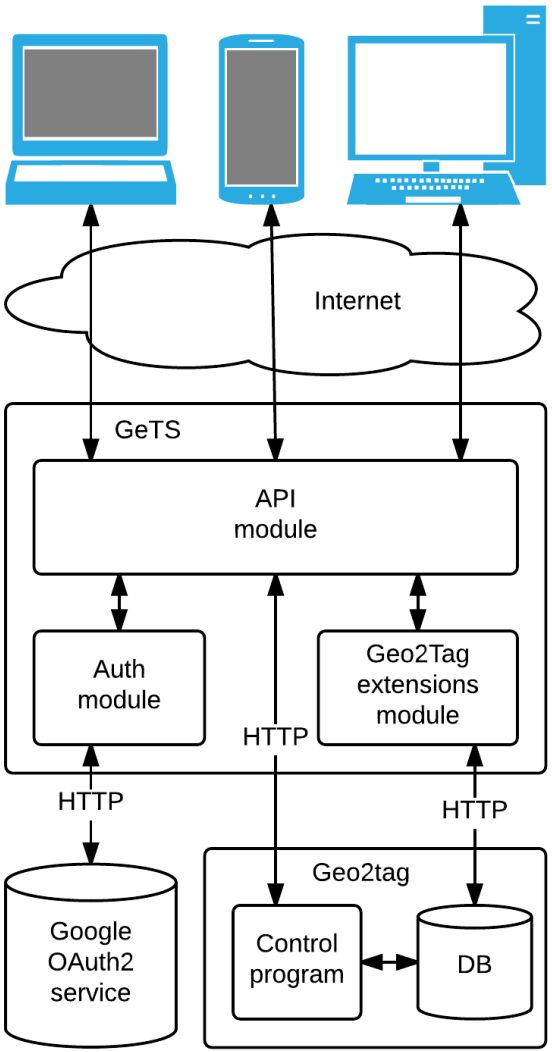
Types of user

- Anonymous user = Read public data (points, tracks, polygons, categories)
- Regular user = Anonymous user functions + private space (read, write, modify)
- Trusted user = Regular user functions + publish/unpublish own data
- Administrator = Trusted user functions + modify all data (points, tracks, polygons, categories)
- Super user = Administrator functions + project ID

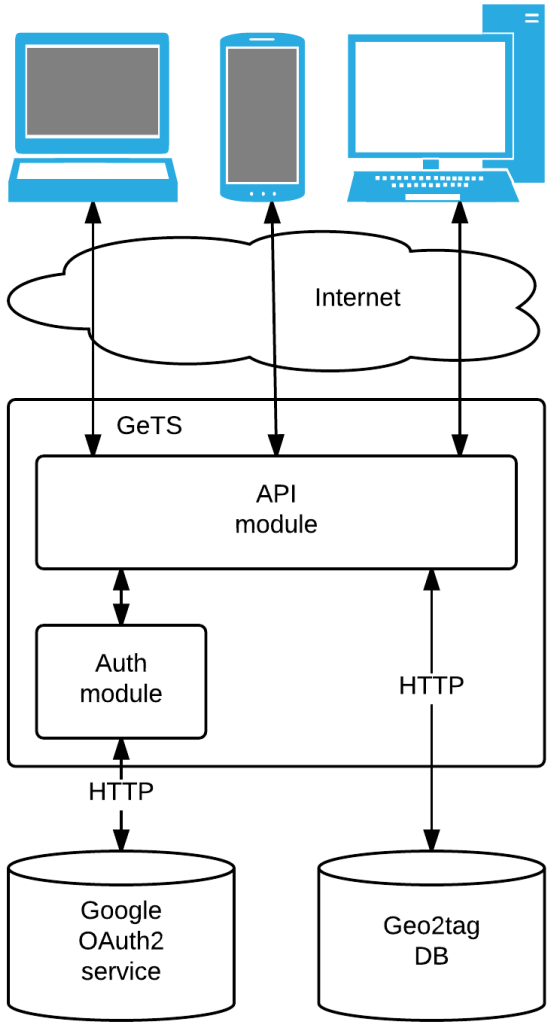
Space division

- Each user has own channels = private space
- Public space = Super user channels
- Project space = all channels with categories with project ID
- Available channels identified by subscription
- Sharing function:
  - User can generate key for any own channel
  - Another user can subscribe to the channel using key
  - Sharing includes subscription number (fixed or infinite)

Service architecture



With Geo2tag program module



Without Geo2tag program module

Service implementation

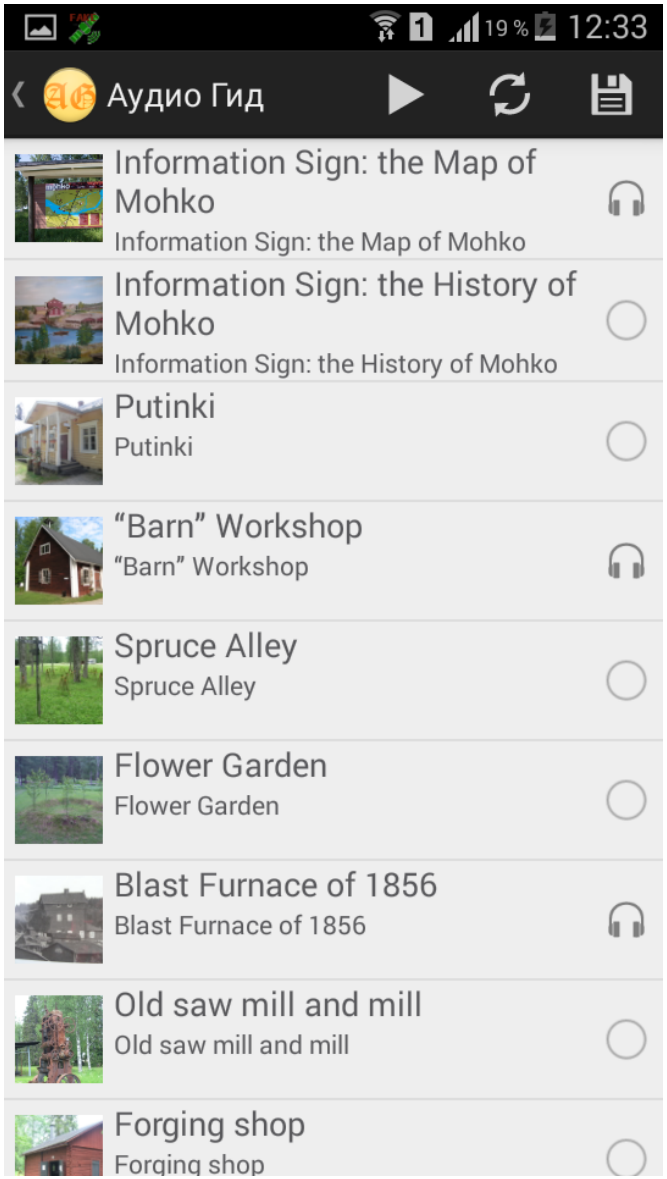
Implementation

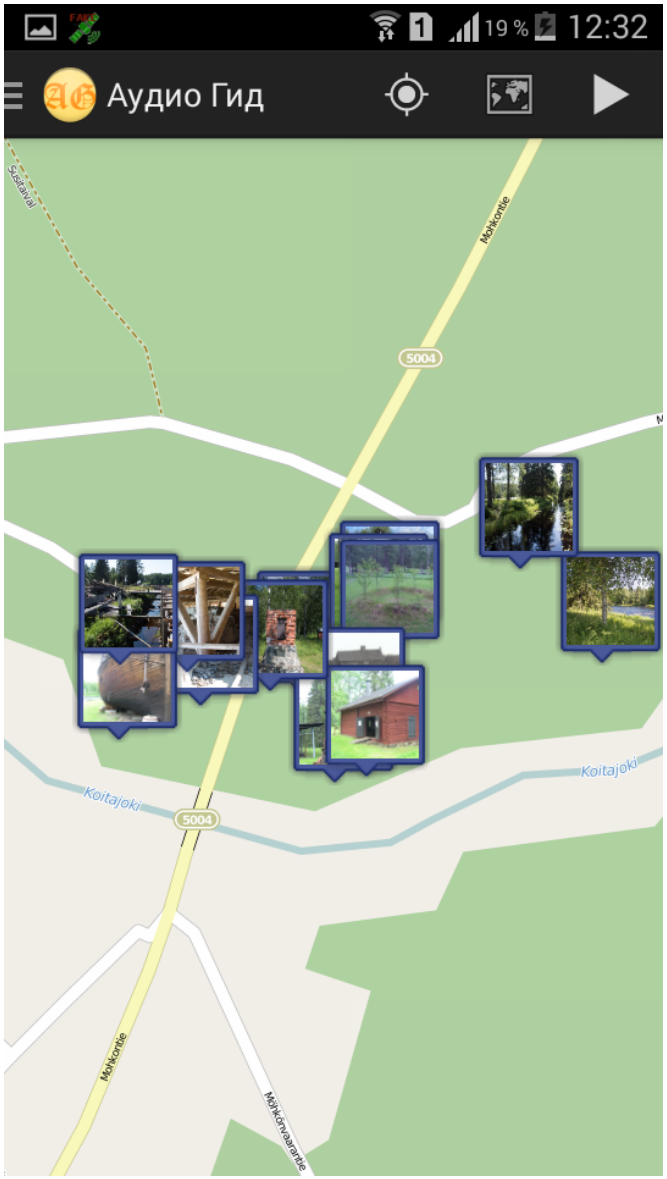
Language	Files	LOC	Comments	Empty strings
PHP	41	3027	274	667
XSD	21	463	1	10
Total:	62	3490	275	677

Testing

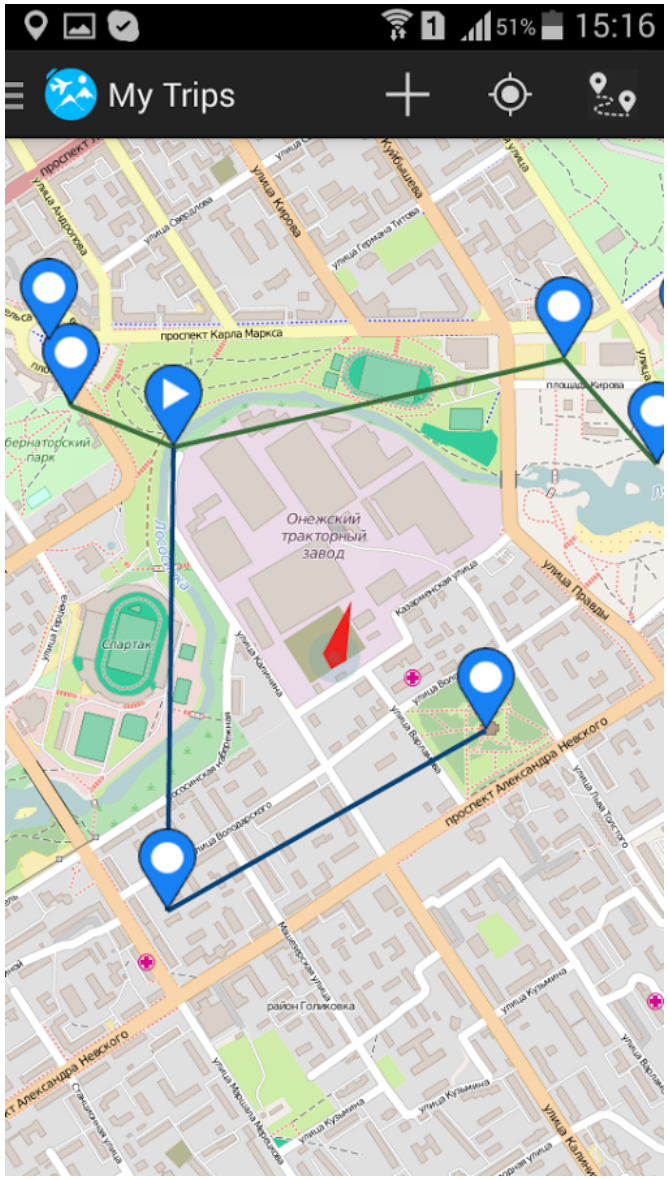
Architecture	With Geo2tag	without Geo2tag
5 tags from channel with 100K tags	1m17.731s	0m0.728s
All tags from all channels (120K) in radius 90km	1m17.692s	0m0.766s
All tags from small channels (1000 tags) in radius 500km	0m0.088s	0m0.049s

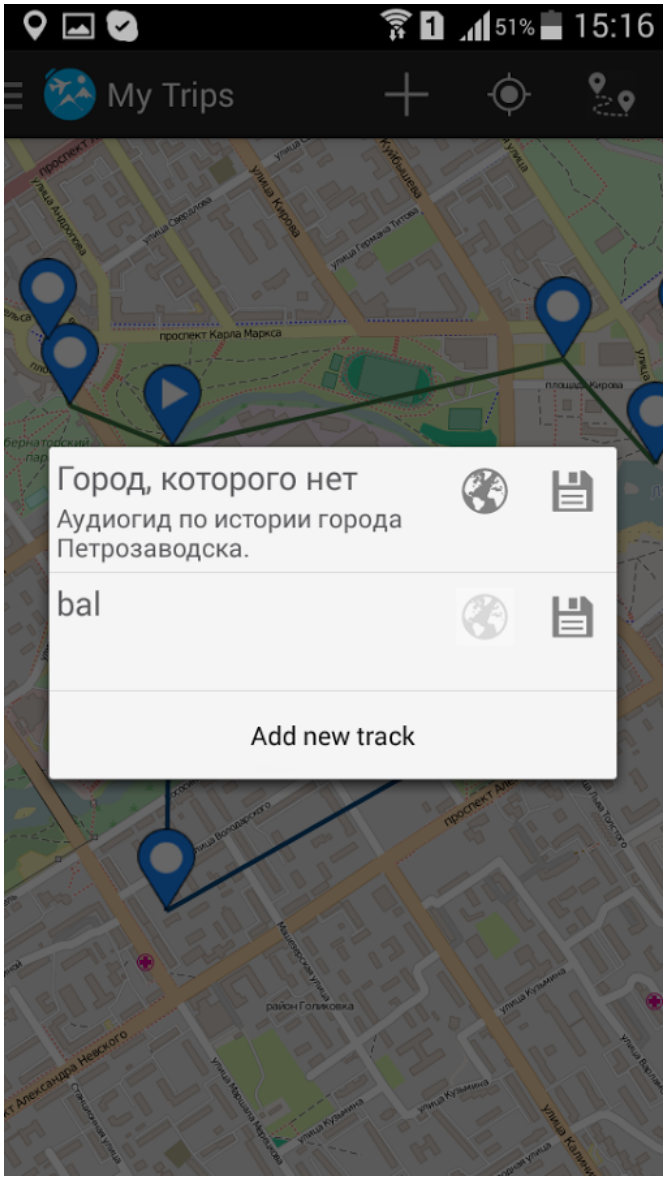
Tourist applications: Audio guide



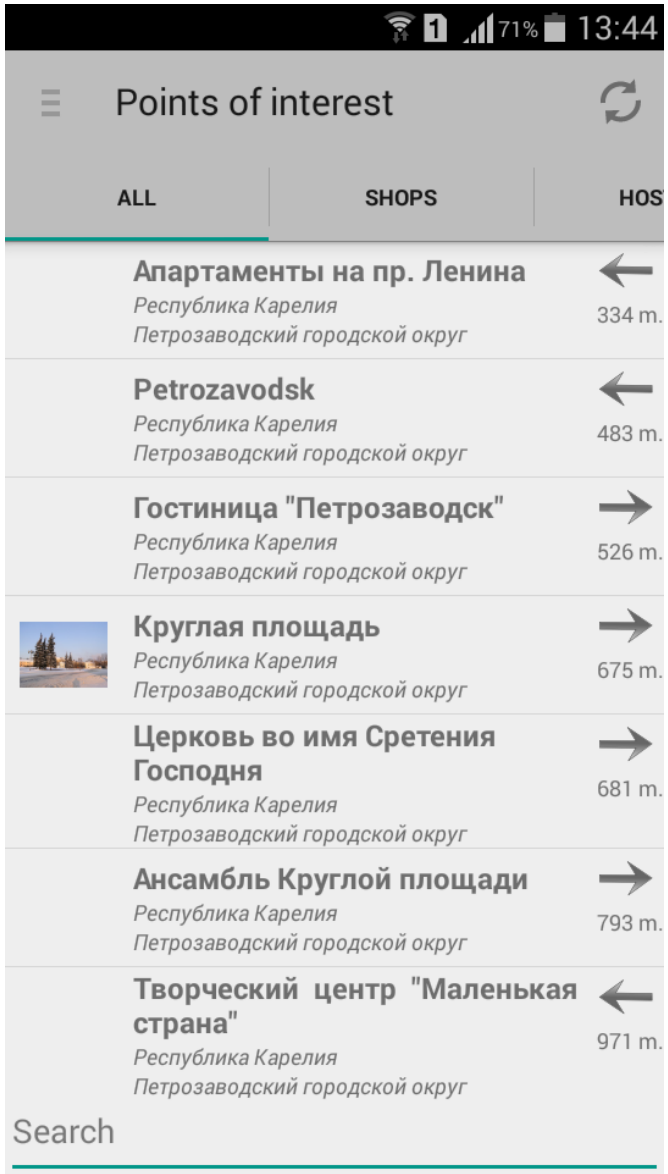


Tourist applications: MyTrips

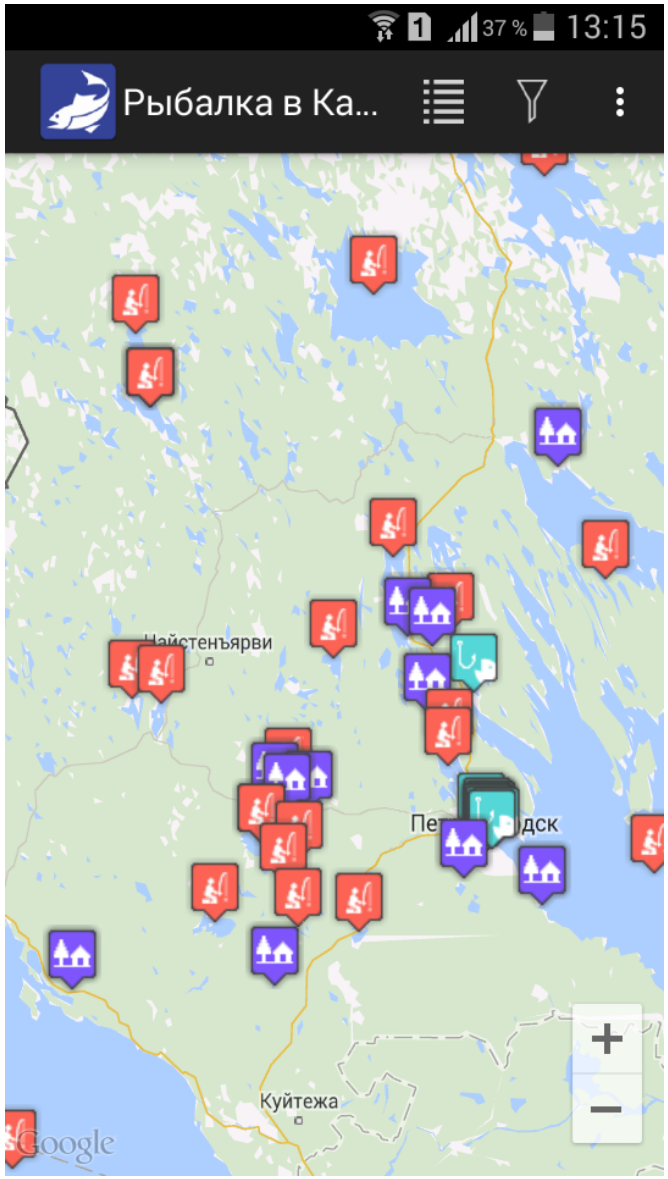




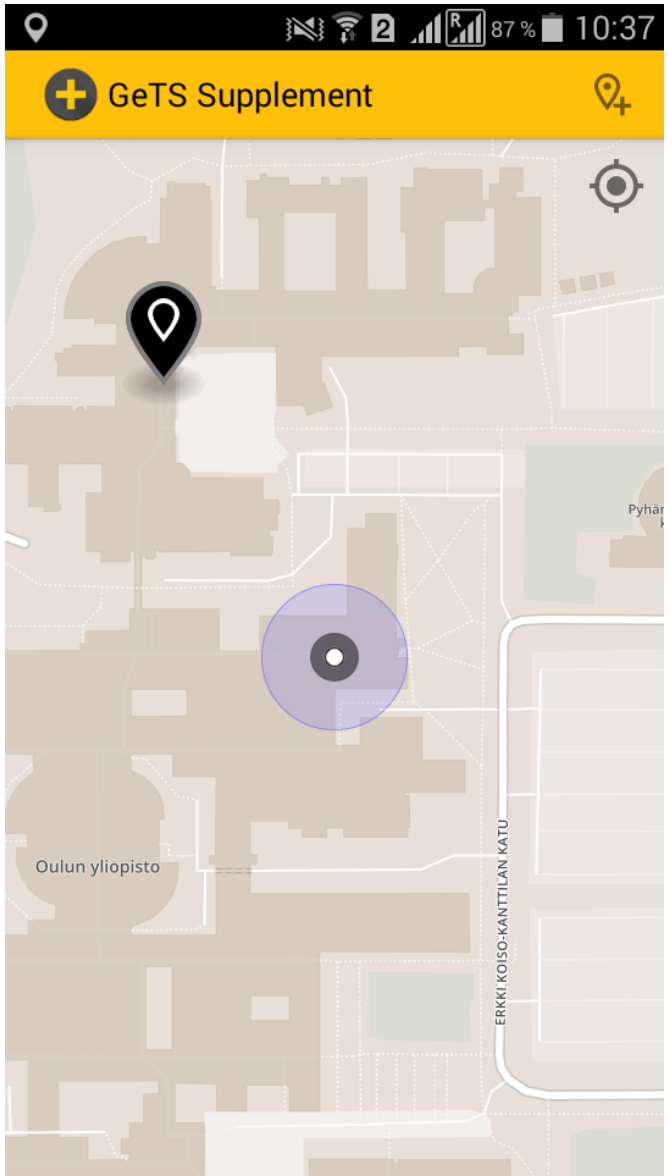
Tourist applications: Virtual road signs

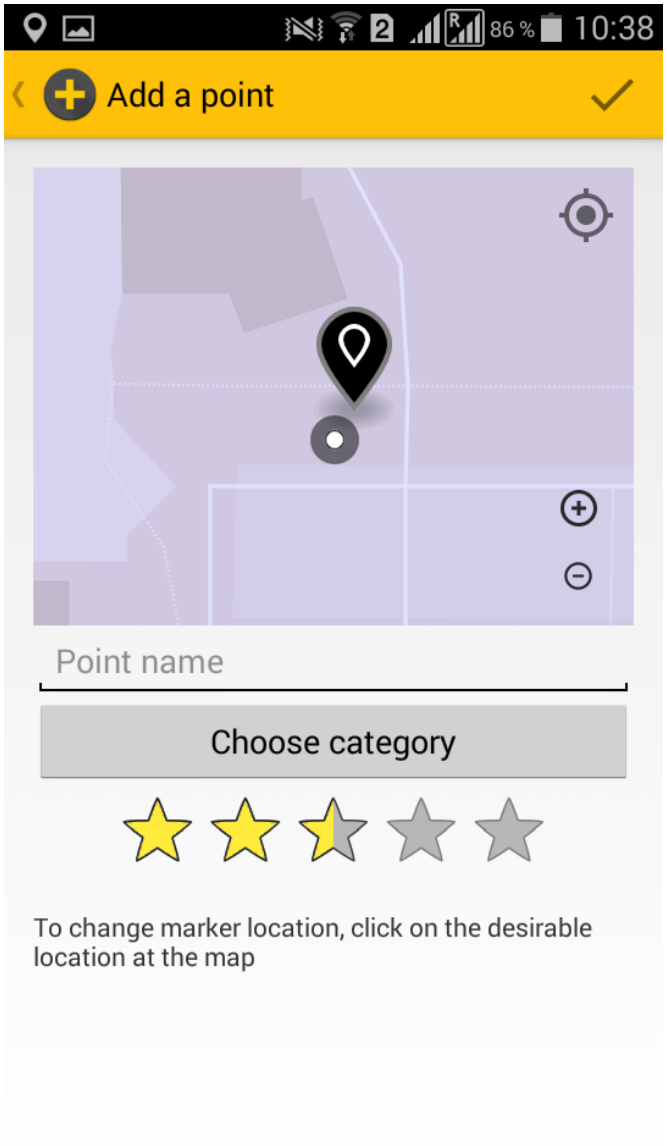


Tourist applications: Karelia fishing

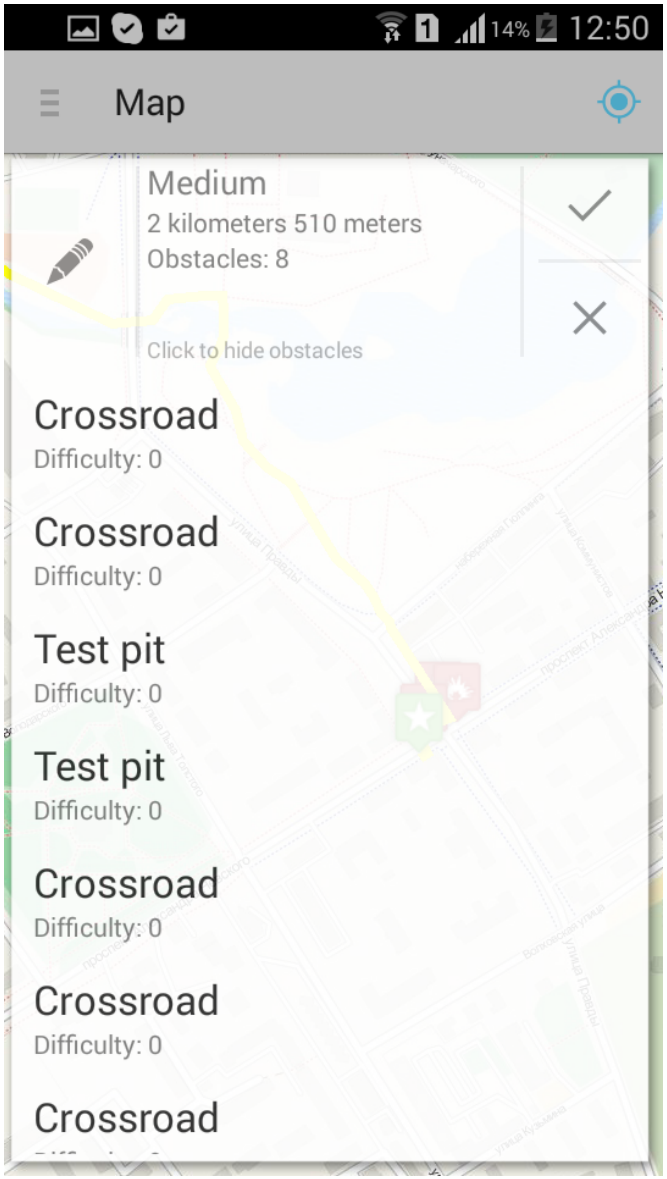


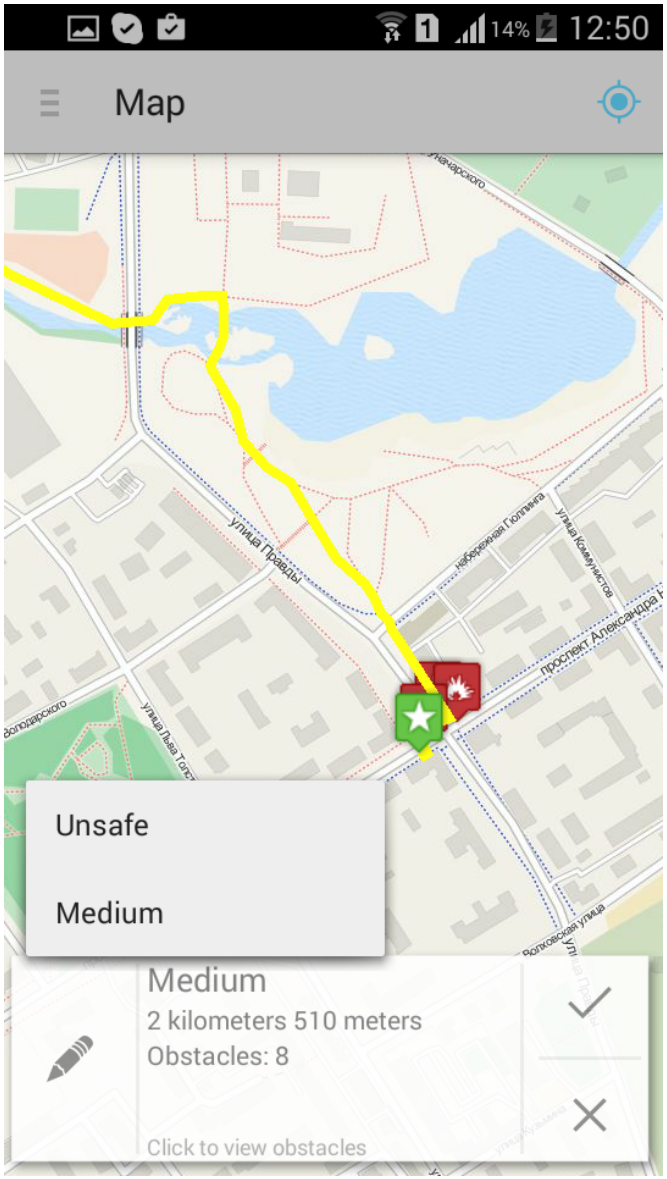
Social applications: GeTS supplement



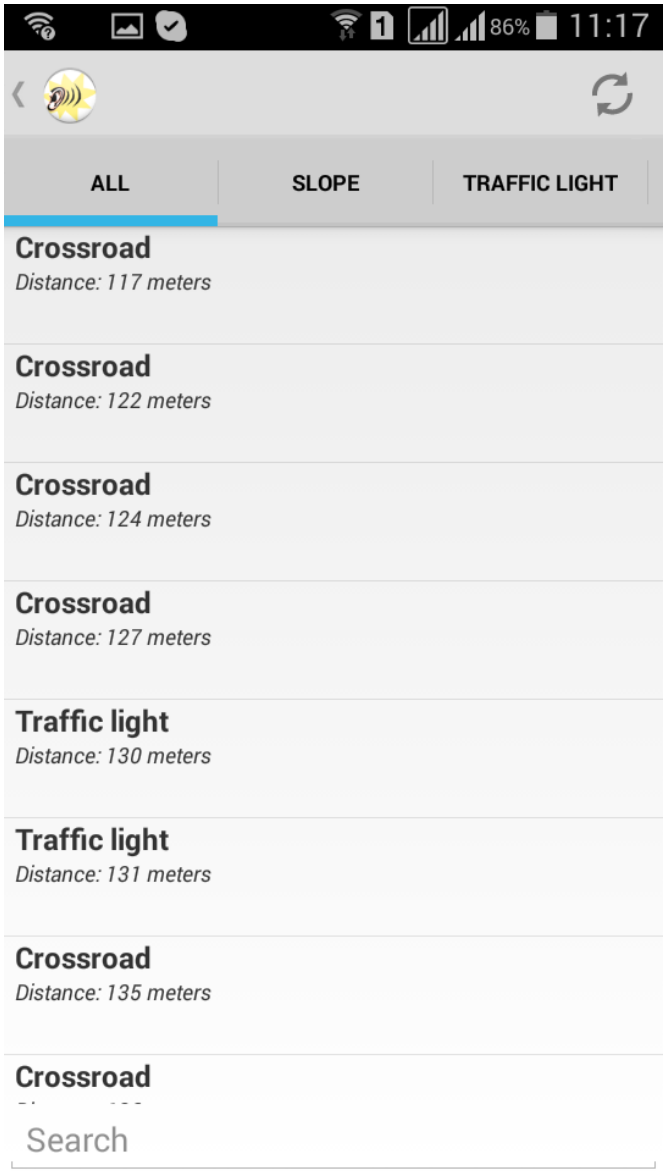


Social applications: Social navigator

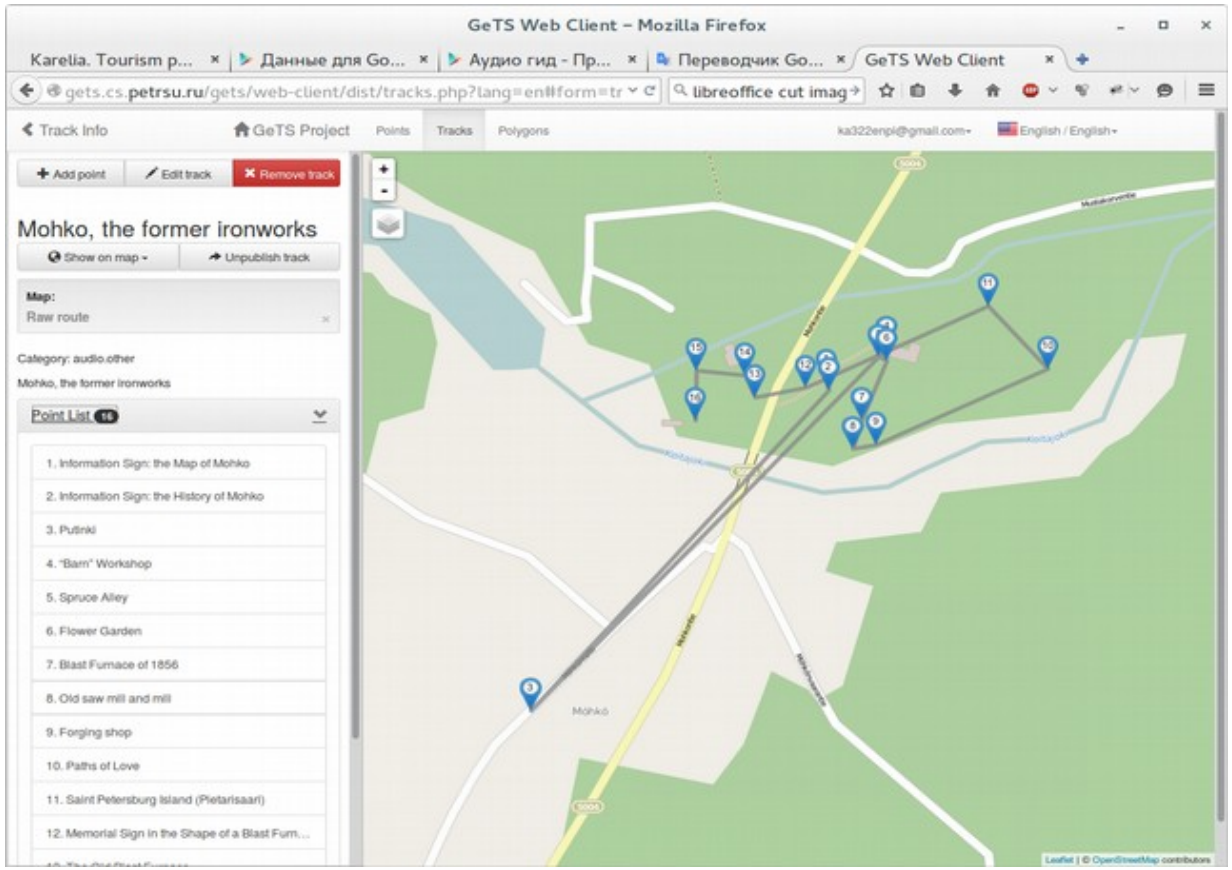




Social applications: Audible alert



GeTS web client



GeTS service: `http://gets.cs.petrsu.ru/gets/service`

Web interface: `http://gets.cs.petrsu.ru/gets`

Source code: `https://github.com/oss-fruct-org/gets`