Deliverable 4.5.1
Initial Release of the Mobile Spatial-Semantic Visualization, Exploration and Authoring Tool

Dissemination Level | Public
---|---
Due Date of Deliverable | Month 24, 30/11/2014
Actual Submission Date | Month 24, 28/11/2014
Work Package | WP4, Spatial-Semantic Browsing, Visualisation and Authoring Interfaces
Task | T4.5
Type | Prototype
Approval Status | Final
Version | 1.0
Number of Pages | 16
Filename | D4.5.1_initial_release_of_the_mobile.spatial.semantic_visualization.exploration_and_authoring_tool.docx

Abstract: The work presented in this deliverable describes GEM, the first mobile faceted browser that fully exploits the potential of the Linked Open Data paradigm. The tool relies on the existing Task 4.1 work and complements the desktop GeoKnow Generator component through a rich mobile Linked Data experience and further rises above the common mobile geospatial visualization limitations by relying on open, crowdsourced and semantically linked information found in publicly available sources, such as the LOD Cloud.

The information in this document reflects only the author’s views and the European Community is not liable for any use that may be made of the information contained therein. The information in this document is provided “as is” without guarantee or warranty of any kind, express or implied, including but not limited to the fitness of the information for a particular purpose. The user thereof uses the information at his/her sole risk and liability.
History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Reason</th>
<th>Revised by</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>24/11/2014</td>
<td>Initial version</td>
<td>Uroš Milošević Claus Stadler</td>
</tr>
<tr>
<td>0.9</td>
<td>26/11/2014</td>
<td>Peer review</td>
<td>Robert Isele</td>
</tr>
<tr>
<td>1.0</td>
<td>28/11/2014</td>
<td>Final version</td>
<td>Uroš Milošević Claus Stadler</td>
</tr>
</tbody>
</table>

Author List

<table>
<thead>
<tr>
<th>Organisation</th>
<th>Name</th>
<th>Contact Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMP</td>
<td>Uroš Milošević</td>
<td><a href="mailto:Uros.Milosevic@pupin.rs">Uros.Milosevic@pupin.rs</a></td>
</tr>
<tr>
<td>InfAI</td>
<td>Claus Stadler</td>
<td><a href="mailto:CStadler@informatik.uni-leipzig.de">CStadler@informatik.uni-leipzig.de</a></td>
</tr>
</tbody>
</table>
Executive Summary

The work presented in this deliverable describes GEM (Geospatial-semantic Exploration on the Move), the first mobile faceted browser that fully exploits the potential of the Linked Open Data paradigm. The design and usability choices of desktop applications make them often hard or impossible to interact with on mobile devices due to both hardware and software constraints. Therefore, the work on the GeoKnow spatial-semantic visualization and exploration component being delivered in Task 4.1 might prove impractical for a user on the go. The extension in Task 4.5 (Mobile spatial-semantic visualization, exploration and authoring), relies on the existing Task 4.1 work and complements the desktop GeoKnow Generator component through a rich mobile experience that exploits all strengths of Linked Data and further rises above the common mobile geospatial visualization limitations by relying on open, crowd-sourced and semantically linked information found in publicly available sources, such as the LOD Cloud. This information is loaded and filtered according to user’s needs, on demand, in order to prevent maps from overpopulating. Finally, special attention is paid to client-side optimization to deliver both acceptable performance and comfortable user/visual experience.
## Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOD</td>
<td>Linked Open Data</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol and RDF Query Language</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CLI</td>
<td>Command Line Interface</td>
</tr>
<tr>
<td>GEM</td>
<td>Geospatial-semantic Exploration on the Move</td>
</tr>
<tr>
<td>JASSA</td>
<td>JAvascript Suite for Sparql Access</td>
</tr>
</tbody>
</table>
Table of Contents

1. INTRODUCTION ........................................................................................................................................ 7

2. MOBILE SPATIAL-SEMANTIC VISUALIZATION, EXPLORATION AND AUTHORING ......................... 7
   2.1 REQUIREMENTS .......................................................................................................................................... 7
   2.2 APACHE CORDOVA ...................................................................................................................................... 8
      2.2.1 Installation .......................................................................................................................................... 9
   2.3 FOUNDATION ............................................................................................................................................ 9
      2.3.1 Jassa ..................................................................................................................................................... 10
      2.3.2 Facete .................................................................................................................................................. 10

3. GEM - GEOSPATIAL-SEMANTIC EXPLORATION ON THE MOVE ......................................................... 11
   3.1 LEAFLET.JS ................................................................................................................................................. 11
   3.2 INSTALLATION .......................................................................................................................................... 11
   3.3 APPLICATION ............................................................................................................................................ 12
      3.3.1 Data source management ..................................................................................................................... 12
   3.4 USAGE ....................................................................................................................................................... 13

4. CONCLUSION ............................................................................................................................................... 15

5. REFERENCES ............................................................................................................................................... 16
List of Figures

Figure 1: Screenshot of the Mappify user interface with highlighted concept configuration panel (upper box) and map preview area (lower box) .................................................................................................................. 9
Figure 2: GEM technology stack ........................................................................................................................................... 11
Figure 3: GEM: a) Application splash screen; b) User interface ......................................................................................... 14
Figure 4: GEM: a) Location & orientation awareness; b) Maximizing visible map area ...................................................... 14
Figure 5: GEM: a) Facet tree side drawer; b) Source manager side drawer .................................................................. 15
Figure 6: GEM: Landscape mode ............................................................................................................................................ 15

List of Tables

Table 1: Functional software requirements specification ........................................................................................................ 8
Table 2: Non-functional software requirements specification ................................................................................................ 8
1. Introduction

The explosion of location aware technology has made the move of geographical information to their, perhaps, more natural setting, i.e. mobile devices, inevitable for any geospatial software striving to survive the demands of the ever-growing market. However, the functionality of the majority of available navigation systems is developed upon closed and proprietary solutions for both maps and software applications. Furthermore, such applications are unable to offer information specifically tailored to user’s needs, and cannot be extended by third parties. Although recent attempts propose ways of overcoming some of these barriers [1, 2, 3], none leverage the full power of the Linked Data paradigm.

The design and usability choices of desktop applications make them often hard or impossible to interact with on mobile devices due to both hardware (smaller screens, lower screen resolutions, lack of buttons, less processing power etc.) and software constraints. Therefore, the work on the GeoKnow spatial-semantic visualization and exploration component (Task 4.1: Spatial-semantic visualization and exploration) might prove impractical for a user on the go (e.g. in a car, on a bike, on foot etc.). The extension in Task 4.5 (Mobile spatial-semantic visualization, exploration and authoring) will complement the desktop component through a rich mobile experience that will exploit all strengths of Linked Data and further rise above the common mobile geospatial visualization limitations by relying on open, crowd-sourced and semantically linked information found in publicly available sources, such as the LOD Cloud. This information will be loaded and filtered according to user’s needs, on demand, in order to prevent maps from overpopulating. Moreover, in order to reach a larger target population and spark community engagement and contribution (through a live authoring component), the tool will be able to run on at least one major mobile platform (Android, iOS, Windows Phone). The resulting component will also be used as a basis for the mobile version of the consolidated spatial view of the supply chain (Task 5.6), that will enable the field personnel involved in managing and executing logistics in the supply chain make on-the-fly decisions (optimize, prevent bottlenecks, etc.) by having real-time access to the transactions in the supply chain.

2. Mobile spatial-semantic visualization, exploration and authoring

The goal of Task 4.5 is to develop a mobile version of the spatial-semantic visualization and exploration component described in Task 4.1, with an additional authoring component that will allow for online updates of geospatial and related information by a user on the move. The tool is envisioned to give the user the ability to load the application with a custom dataset through one of the available SPARQL endpoints and retrieve the data they might be interested in on request, instead of (over)loading the mobile map with irrelevant information from the start.

2.1 Requirements

GeoKnow’s goal is to deliver an easy to use, yet powerful mobile visualization and exploration tool that will provide a highly customizable and information rich slippy map to the geospatial data consumers on the move. The component needs to target major mobile platforms to increase the impact on the society as a whole, as well as the possibility of igniting community engagement and contribution. More specifically, the user interface is planned to consist of:

- An interactive map component, to be used for quick and easy exploration of geographical areas, showing the user’s own position (GPS coordinates) and the surrounding area.
A semantic facet filtering component and a result view, similar to the desktop/web-based components described in Task 4.1.

An authoring component, to be used for quick editing of existing information (e.g. fixing outdated information) and adding new information on the move.

As previously mentioned, the tool needs to be available on at least one major mobile platform (Android, iOS, Windows Phone) in order to reach a larger target population. Moreover, as the resulting software solution will be used as a basis for Task 5.6, joint evaluation and testing activities for both components will be performed within Task 5.5 and reported on in Deliverable 5.5.1. More formally, we list the most essential software requirements in the table below:

<table>
<thead>
<tr>
<th>No.</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF1</td>
<td>The user interface will be made mobile friendly</td>
</tr>
<tr>
<td>RF2</td>
<td>The tool will be made location aware (e.g. by relying on GPS, WiFi and cellular networks)</td>
</tr>
<tr>
<td>RF3</td>
<td>The tool will enable on-the-fly management of data sources (i.e. SPARQL sources)</td>
</tr>
<tr>
<td>RF4</td>
<td>Information on the map will be searchable through a faceted-filter component</td>
</tr>
<tr>
<td>RF5</td>
<td>The application will enable information sharing between the desktop (Facete2) and mobile versions of the tool</td>
</tr>
</tbody>
</table>

| Table 1: Functional software requirements specification |

Additionally, based on the above table, we provide the following list of non-functional requirements the mobile spatial-semantic visualization, exploration and authoring tool should also fulfill.

<table>
<thead>
<tr>
<th>No.</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNF1</td>
<td>The tool will exploit the existing results of Task 4.1: Spatial-semantic visualization and exploration</td>
</tr>
<tr>
<td>RNF2</td>
<td>The application will be made to work on at least one major mobile platform (Android, iOS, Windows Phone)</td>
</tr>
<tr>
<td>RNF3</td>
<td>The tool will be made open-source</td>
</tr>
</tbody>
</table>

| Table 2: Non-functional software requirements specification |

### 2.2 Apache Cordova

The above listed requirements have narrowed down the choice of mobile development frameworks to *Apache Cordova* / *Adobe Phonegap* (formerly known as *Apache Callback*), which is an open-source set of device APIs that allow accessing native device function such as the camera or accelerometer from JavaScript. Combined with a UI framework such as jQuery Mobile or Dojo Mobile or Sencha Touch, this allows a smartphone app to be developed with just HTML, CSS, and JavaScript, which in turn, allows us to build upon the work already delivered in Task 4.1, while focusing on the mobile user experience.

2 [http://phonegap.com/](http://phonegap.com/)
Since these JavaScript APIs are consistent across multiple device platforms and built on web standards, GEM is expected to be portable to other device platforms with minimal to no changes (Cordova is available for: iOS, Android, Blackberry, Windows Phone, Palm WebOS, Bada, and Symbian). This mobile development framework provides a set of uniform JavaScript libraries that can be invoked, with device-specific native backing code for those libraries. Moreover, the app will still be packaged using the platform SDKs and can be made available for installation from each device’s app store (e.g. Google Play, Apple App Store, Windows Phone Store etc.).

2.2.1 Installation

The easiest way to deploy Cordova is via Node.js\(^3\). Following installation, the user is able to invoke node and npm via their preferred command line interface (CLI). To install/download the cordova module using the npm utility of Node.js, we run:

```
  npm install -g cordova
```

2.3 Foundation

To provide means to easily explore and visualize spatial content retrieved from the Web of Data, new paradigms like faceted browsing have to be explored. Apart from this, new techniques need to be developed to combine semantic information with geospatial data and display them with state-of-the-art map rendering tools. Moreover, to support the reuse of certain application units and separate different concerns like browsing and presentation, a software implementing these paradigms and techniques should consist of distinguishable components.  Mappify\(^4\) (Task 4.1) comprises different libraries and widgets supporting these aims. Hence, a separate library for the facet based browsing component is used. The task of providing a unified model to access semantic and geospatial data is also solved by a dedicated software library. The work on Mappify and its components is described in deliverable D4.1.1 and summarized further below.

\(^3\) [http://nodejs.org](http://nodejs.org)

\(^4\) [https://github.com/GeoKnow/Mappify](https://github.com/GeoKnow/Mappify)
2.3.1 Jassa

Mappify makes use of two novel technologies for JavaScript-based interaction with SPARQL endpoints, which are provided by the Javascript Suite for Sparql Access (Jassa) library:

- **Sponate** is a SPARQL-to-JSON mapper with the goal of offering a MongoDB-like API for accessing virtual JSON documents based on a SPARQL endpoint and a set of mapping definitions. Sponate is also in use on the http://js.geoknow.eu website, which lists all JavaScript libraries published as part of GeoKnow.
- **Facete** is a library for faceted search on SPARQL accessible data.

The Jassa code base can be used both on the client and the server side. The only difference between these settings lies in the dependencies that need to be included. Furthermore, Jassa provides several RDF and SPARQL foundation classes which are nearly identical to those of the excellent API of the Java-based Apache Jena project. The rationale followed by Jassa is to exploit existing, well-known API designs, rather than to invent new ones. At present, Jassa comprises the following modules:

- **util**: A utility module. Contains collections, such as HashMap and HashSet.
- **rdf**: The module that holds core classes related to RDF.
- **vocab**: A module for vocabularies, expressed in terms of classes of the rdf module.
- **sparql**: A module for core classes related to SPARQL. Builds upon the prior modules.
- **service**: Abstraction layer for SPARQL endpoints.
- **facete**: A faceted search module.
- **sponate**: A SPARQL-to-JSON mapper. Particularly powerful in combination with the generation of web frameworks that offer a clean separation between DOM and application logic, such as angular.js and Ember.js.

The Jassa core classes aim to serve as a solid foundation for JavaScript-based Semantic Web applications. The rdf module provides the Node class for encapsulating RDF terms. In contrast to approaches that are based on plain JSON, a Jassa node object provides methods such as isLiteral() and getLiteralDatatypeUri(). Furthermore, the toString() method is implemented to yield meaningful string representations. The sparql module contains several classes for the syntactic representation of SPARQL queries. Their use is demonstrated below.

2.3.2 Facete

**Facete** is a module that builds upon all prior modules and offers SPARQL-based faceted search. Facete is conceptually based on three fundamental constructs:

- **Concept**: A concept is a pair comprised of a SPARQL graph pattern and a variable thereof.
- **Path**: A list of steps, whereas a step is comprised of an RDF property name and a direction. A step’s direction is either **forward** or **backward** and specifies whether the step leads to the set of resources occurring in the subject or the object position of that property.
- **FacetNode**: A facet node represents a variable mapping for a given path. Equivalent paths are always mapped to the equivalent variables.

Facete is configured with the following essential parameters:

- The **Base Concept** specifies the initial set of resources on which the faceted search is performed.
- The **Constraint Manager** provides a container for constraints.

---

5 https://jena.apache.org/
• The *Root Facet Node*: The root node of a tree structure with the purpose of mapping paths to SPARQL variables and vice versa.

3. **GEM**

Our approach to the problem of putting the power of the above components in the hands of the mobile data consumers, titled *GEM (Geospatial-semantic Exploration on the Move)*, relies on tailor-made and open-source Web technologies that are optimized for handheld devices.

![GEM technology stack](image)

**Figure 2**: GEM technology stack

3.1 **Leaflet.js**

*Leaflet.js* is a modern open-source JavaScript library for mobile-friendly interactive maps. Weighing just about 33 KB of JS, it has all the features most developers ever need for online maps. The library is designed with *simplicity, performance* and *usability* in mind. It works efficiently across all major desktop and mobile platforms, out of the box, taking advantage of HTML5 and CSS3 on modern browsers while still being accessible on older ones. It can also be extended with numerous plugins, and is an adequate lightweight replacement for the OpenLayers library being used with Jassa.

3.2 **Installation**

All of the GEM source code is made available through the official GeoKnow GitHub repository[^8]. Automation and package dependency management are controlled via Grunt[^5] and Bower[^10], respectively.

[^5]: http://leafletjs.com/
[^6]: http://openlayers.org/two/
[^8]: https://github.com/GeoKnow/GEM
Batch installation/update scripts are provided for both Unix-based and Windows operating systems (bower-update.sh and bower-update.bat), and the mobile application source code is easily compiled by simply executing the following command from the desired CLI:

```
cordova build <platform>
```

where <platform> can be any of the Cordova/Phonegap supported platforms (e.g. android). It is worth noting that the Phonegap equivalent of the cordova command is phonegap. To compile and deploy the application to the desired device, we execute:

```
cordova run <platform>
```

3.3 Application

The design philosophy behind GEM aims at maximizing the usable application/screen area through one stationary design component, and four on-demand widgets. The slippy map component represents the base layer on top of which the control widgets appear on request. The four components are highlighted in Figure 3b, and their functionalities described below:

1. The Facets side drawer holds the loaded resource facet tree;
2. The Source Manager side drawer is used to add/edit/remove available SPARQL endpoints / Linked Open Data sources;
3. The Filter text box, as the name suggests, is used to filter the resources on screen. The widget also holds the orientation indicator (i.e. compass);
4. The Resource details bottom drawer pops-up to display the relevant information (e.g. label, URI and related triples) for the selected feature.

Different icon colors are used to indicate different resources/statuses. Moreover, to further prevent the map from overpopulating, and avoid overloading the mobile device resources in situations where multiple features are in each other’s vicinity (relative to the map zoom level), we resort to marker clustering, i.e. grouping. Marker cluster groups are indicated by circles with resource counts. The color of the circle depends on the number of resources being grouped (ranging from green to red; green indicating small groups; Figure 3b).

3.3.1 Data source management

The information shown on the screen is easily controlled using the Source Manager widget, which lets the user specify the name of the source (for convenience), the corresponding SPARQL endpoint and graph, as well as the desired resource type to be retrieved / displayed on the map. As Cordova relies on the host device’s built-in Internet browser facilities, we exploit the browser’s HTML5 Local Storage to maintain the user preferences across sessions (i.e. automatically load the last used resource list the next time the application starts). Before HTML5, application data had to be stored in cookies, included in every server request. Local storage is more secure, and large amounts of various data can be stored locally, without affecting website/application performance.

9 http://gruntjs.com/
10 http://bower.io/
3.4 Usage

We hereby outline the essential functionalities of the mobile geospatial-semantic browser provided by the above described facilities, and the basic steps to using them:

- The first time the application starts, GEM automatically populates the list of available LOD sources with DBpedia and LinkedGeoData¹¹ SPARQL endpoints, using Airports as Base concepts (this is currently for demonstration purposes and might change in the future). The user can easily change this, by either editing the existing source list, or by creating a new one (i.e. by adding new endpoints / data types and/or removing the default ones; Figure 5b).
- On startup, if the host OS location services are available (e.g. GPS), the map will zoom in on the user’s own location (indicated by a blue gem icon; Figure 4a.1), and the orientation indicator / compass in the filter widget will show (and update, based on) the direction the user’s device is pointing to (Figure 4a.2).
- Browsing the map will load the corresponding resources for the visible map area.
- To further make best use of the available screen space, upon detecting map interaction / movement, GEM will hide away the GUI elements the user cannot interact with while browsing through the map (Figure 4b). Additionally, the application can also work in landscape mode (Figure 6).
- Should the user decide to narrow down their search, they can do so either by using the Filter box, or by using the Facet tree side drawer (Figure 5a), which functions much in the way the Facette facet tree does. The user first picks the facets (i.e. properties), and then the values to restrict the results to only those that fit the given description.
- Finally, to retrieve the details related to a single resource, the user only has to select the corresponding marker on the map. The bottom drawer reveals the feature’s name (i.e. label) and URI. Clicking on the resource label expands the bottom drawer to display other relevant attributes (i.e. corresponding triples).

¹¹ http://linkedgeodata.org/
Figure 3: GEM: a) Application splash screen; b) User interface

Figure 4: GEM: a) Location & orientation awareness; b) Maximizing visible map area
4. Conclusion

The GEM prototype described above represents a one-of-a-kind mobile faceted geospatial-semantic browser for Linked Open Data. It builds on top of the efforts already invested in Work Package 4, Task 4.1, and state of the art open source technologies, ranging from Mappify, Jassa, Facete and Sponate, to well-known third-party, community-maintained frameworks and libraries, such as AngularJS and Leaflet.js. It is deployed using Apache Cordova / Adobe Phonegap, allowing single-branch development for multiple major mobile platforms.
In its early stages, GEM development was focused primarily on making the planned integration with the existing WP4 work possible, enabling the essential functionalities and providing an adequate mobile user experience.

In its next incarnation, GEM is planned to deliver many new features, starting with backend improvements that will enable importing data based on other coordinate systems, allowing for richer and more complex data integrations. Moreover, the final version of the tool will also come with an authoring component that will give an entirely new dimension to the two-way interaction between the user and the application, and hopefully make a push towards wide community acceptance through crowdsourcing. Finally, further user experience improvements will be considered as well, such as automatic (re)source recommendations and tighter integration with Mappify, enabling easy sharing of information between the desktop and mobile clients.

5. References

