Deliverable 4.5.2: Final Release of the Mobile Spatial-Semantic Visualization, Exploration, and Authoring Tool

Dissemination Level | Public
--- | ---
Due Date of Deliverable | Month 36, 30/11/2015
Actual Submission Date | Month 36, 30/11/2015
Work Package | WP4
Task | Task T4.5
Type | Prototype
Approval Status | Final
Version | 1.0
Number of Pages | 20
Filename | D4.5.2_GEM_Final_release.pdf

Abstract: The work presented in this deliverable describes the final version of GEM (Geospatial Exploration on the Move), 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 leverages smartphone capabilities to deliver faceted browsing and semantic routing based on open, crowd-sourced 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>0.0</td>
<td>08/11/2015</td>
<td>First version and section outline</td>
<td>Valentina Janev, Uroš Milošević</td>
</tr>
<tr>
<td>0.1</td>
<td>14/08/2015</td>
<td>First draft - description of Jassa</td>
<td>Vuk Mijović</td>
</tr>
<tr>
<td>0.2</td>
<td>18/08/2015</td>
<td>Description of GEM</td>
<td>Vuk Mijović, Dejan Paunović</td>
</tr>
<tr>
<td>0.3</td>
<td>26/11/2015</td>
<td>Interim version ready for peer review</td>
<td>Vuk Mijović</td>
</tr>
<tr>
<td>0.4</td>
<td>27/11/2015</td>
<td>Internal review – comments</td>
<td>Matthias Wauer</td>
</tr>
<tr>
<td>0.5</td>
<td>27/11/2015</td>
<td>Internal review comments addressed</td>
<td>Vuk Mijović</td>
</tr>
<tr>
<td>1.0</td>
<td>30/11/2015</td>
<td>Final approval and modifications</td>
<td>Sanja Vraneš</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>Vuk Mijović</td>
<td><a href="mailto:vuk.mijovic@pupin.rs">vuk.mijovic@pupin.rs</a></td>
</tr>
<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>IMP</td>
<td>Valentina Janev</td>
<td><a href="mailto:valentina.janev@pupin.rs">valentina.janev@pupin.rs</a></td>
</tr>
<tr>
<td>IMP</td>
<td>Dejan Paunović</td>
<td><a href="mailto:dejan.paunovic@pupin.rs">dejan.paunovic@pupin.rs</a></td>
</tr>
<tr>
<td>IMP</td>
<td>Sanja Vraneš</td>
<td><a href="mailto:sanja.vranes@pupin.rs">sanja.vranes@pupin.rs</a></td>
</tr>
</tbody>
</table>
Executive Summary

The work presented in this deliverable describes the final version of GEM (Geospatial Exploration on the Move), 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 leverages smartphone capabilities to deliver faceted browsing and semantic routing based on open, crowd-sourced and semantically linked information found in publicly available sources, such as the LOD Cloud. GEM is the first mobile faceted browser based on Linked Open Data paradigm, as well as the first application to enable semantic routing, which is immensely useful feature for users on the go, especially when they find themselves in an unknown environment. In addition to semantic routing, the final version that is presented in this deliverable brings many other improvements that will boost community engagement and enhance the user experience by preventing cluttering in the user interface, while integration with Mappify allows reuse of Mappify configurations within GEM as well. Lastly, GEM’s potential in other, more specialized use cases, such as e-commerce, was explored.
Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</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>RDFS</td>
<td>Resource Description Framework Schema</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol and RDF Query Language</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>GEM</td>
<td>Geospatial Exploration on the Move</td>
</tr>
<tr>
<td>JASSA</td>
<td>Javascript Suite for Sparql Access</td>
</tr>
<tr>
<td>POI</td>
<td>Point Of Interest</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
</tbody>
</table>
Table of Contents

1. INTRODUCTION ............................................................................................................. 7
2. MOBILE SPATIAL-SEMANTIC VISUALIZATION, EXPLO-RATION, AND AUTHORING .................................................. 8
   2.1 REQUIREMENTS ........................................................................................................... 8
   2.2 FOUNDATION ............................................................................................................. 9
   2.3 MAPPIFY (CONFIGURATION SHARING/IMPORTING) .................................................. 9
3. GEM ...................................................................................................................................... 11
   3.1 ARCHITECTURE .......................................................................................................... 11
   3.2 INSTALLATION .......................................................................................................... 12
   3.3 APPLICATION DESCRIPTION ...................................................................................... 12
      3.3.1 Slippy Map ........................................................................................................... 13
      3.3.2 Source Manager ................................................................................................. 13
      3.3.3 Resource Details ............................................................................................... 15
   3.4 APPLICATION DEFAULTS .......................................................................................... 16
4. E-COMMERCE USE CASE .............................................................................................. 17
5. CONCLUSIONS .................................................................................................................. 19
6. REFERENCES ...................................................................................................................... 20
List of Figures

Figure 1 Saving Configuration in Mappify................................................................. 10
Figure 2 GEM Technology Stack .............................................................................. 11
Figure 3 GEM User Interface .................................................................................. 12
Figure 4 GEM Slippy Map: a) resource view; b) navigation ...................................... 13
Figure 5 Source Manager: a) Main view; b) Edit properties; c) Edit prefixes ................ 14
Figure 6 Resource recommendations........................................................................ 15
Figure 7 Resource Details: a) Information view; b) Directions view ............................ 16
Figure 8 Validating polygons .................................................................................... 17
Figure 9 Finding a perfect place for a new hotel.......................................................... 18

List of Tables

Table 1 Functional software requirements specification ............................................. 8
Table 2 Non-functional software requirements specification ...................................... 9
1. Introduction

In the recent years, there have been substantial developments with regards to software solutions for browsing and exploring geospatial information, as well as navigation. However, these solutions are often closed and proprietary, and more importantly, they are very limited with respect to the supported sources of information. Even the support for several sources of information often requires a significant effort imposed by different schemas, while enabling a user to manually add arbitrary sources of information is next to impossible with traditional technologies such as relational databases. Linked Data on the other hand enables to easily overcome these issues. Being based on RDF (Resource Description Framework), a flexible schema-less data model, Linked Data enables straightforward integration of different datasets, while SPARQL allows to uniformly query arbitrary sources of information.

One example of an application that leverages Linked Data to offer browsing of geospatial information is Facete, a faceted browser for the Web of Data. Facete enables browsing and exploration of any number of arbitrary SPARQL endpoints, where a user can filter information to be shown on the map by defining facets, i.e. types of resources and constraints on their properties. GEM builds on top of Facete, and offers browsing of geospatial information that is optimized for smartphones, meaning that it addresses hardware limitations (smaller screen, lower resolution, less processing power, lack of buttons, etc.) and leverages advantages of mobile devices (such as positioning). Therefore, just like Facete, GEM can browse any number of SPARQL endpoints and filter resources based on their type and constraints on properties, as well as leverage GPS positioning to deliver semantic routing, while the user interface was designed to provide comfortable user experience despite inherent limitations of mobile devices.

Section 2 will describe core libraries developed in WP4 that serve as a basis for GEM. The application itself is described in Section 3, while Section 4 provides an example extension and adaptation of GEM for the e-commerce use case. Conclusion and future steps are given in Section 5.
2. Mobile Spatial-Semantic Visualization, Explo-ration, 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 gives the user the ability to choose arbitrary datasets by specifying desired SPARQL endpoints, and retrieves data on request, instead of (over)loading the mobile map with irrelevant information from the start. Furthermore, after the resources are filtered based on semantics provided by the user, the tool allows to pick the most interesting ones for which to generate an optimal route. This section will briefly revisit the requirements and technological foundation, while full descriptions can be found in the previous deliverable [1].

2.1 Requirements

The goal of this tool is to provide a highly customizable and information rich slippy map to the geospatial data consumers on the move. Therefore, the user interface consists of the following:

- 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.

Furthermore, in order to maximize its impact, the tool needs to be available on at least one major mobile platform (Android, iOS, Windows Phone). More formally, list of the most essential software requirements is given 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>
<tr>
<td>RF6</td>
<td>The application will enable authoring of information on the go</td>
</tr>
<tr>
<td>RF7</td>
<td>The tool will enable routing between the selected resources</td>
</tr>
</tbody>
</table>

Requirements RF5, RF6, and RF7 were addressed entirely in the final version presented in this deliverable, while the requirement RF1 was already addressed in the previous deliverable [1] and is
now further satisfied. 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 2 Non-functional software requirements specification**

<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>

### 2.2 Foundation

For the sake of completeness, we will now briefly reflect on the GEM’s technological foundation, while the reader is referred to sections 2.2 and 2.3 in the previous deliverable [1]. Having in mind that the goal was to exploit the existing results of Task 4.1 which is based on web technologies, and to make the application available for at least one major platform (preferably more with little effort), we decided to build the tool on top of Apache Cordova, an open-source mobile application development framework that provides a set of device APIs that allow accessing native device function such as camera or accelerometer from JavaScript. GEM’s faceted filtering capabilities are based on *Jas* (Javana), a library that provides several RDF and SPARQL foundation classes that mimic the well-known and proven API of the Java-based Apache Jena1 project, and its modules, primarily Facete which provides SPARQL-based faceted search. Since the previous release, Facete was extended with additional map factories, which enabled the support for more coordinate systems (other than WGS84) in both Facete and GEM. Finally, GEM builds on top of *jassa-ui-angular-core*2, a set of reusable AngularJS directives (widgets) for user interface components that takes full advantage of Facete and other modules in the Jassa library.

### 2.3 Mappify (Configuration Sharing/Importing)

Mappify is a Web application for creating map views over RDF data that was developed within Task 4.1. Once created, map views can be exported directly as JSON configuration objects for reuse in other applications as preconfigured AngularJS project templates. The main goal is to provide a basis for building a Web service that creates embeddable HTML snippets from such configuration objects. However, such functionality is very appealing for GEM as well. Namely, due to the difference in screen size, it is much more comfortable to specify desired views on desktop computers than on mobile devices. Therefore, it would be very beneficial to be able to create sophisticated configurations on desktop and then import them into GEM, after which a user would be able to further adjust the view with respect to the changing needs. For example, one might prepare a view

---

1. [https://jena.apache.org/](https://jena.apache.org/)
beforehand, and simply load it on the move when needed. This also enables users to share views between themselves. Let’s say Jim is travelling to London, and his friend John was there just a few months ago. John can use Mappify to prepare a view with bars, or sights that he found most interesting and then send it to Jim. Once he gets to London, Jim can load John’s configuration in GEM, further adjust it to suit his current needs, and then navigate to the chosen points of interest.

Figure 1 Saving Configuration in Mappify
3. GEM

This section describes the architecture and provides installation instructions for GEM, followed by the detailed description of the application itself and a user guide.

3.1 Architecture

GEM technology stack starts with the Web of Data, where any SPARQL endpoint can be browsed. For querying the underlying data layer, and employ faceted filtering, GEM relies on JASSA libraries and its modules that were described in section 2. After the querying and filtering phase, data is shown on the map using LeafletJS\(^3\), a modern open-source JavaScript library for mobile-friendly interactive maps. Leaflet imposed itself as a good fit by being lightweight, simple, and delivering good performance, while still offering a plethora of features. Moreover, it can be extended with numerous plugins that are available on the web, such as the Leaflet Routing Machine\(^4\), an easy, flexible and extensible plugin that adds routing to a Leaflet map. The plugin supports several routing engines, and enables routing from start to destination, with the possibility to add, edit, and remove waypoints through both address input and using the map, thereby powering GEM’s semantic routing capabilities. Finally, GEM is an Apache Cordova / Adobe Phonegap application and can therefore be packaged for multiple platforms, though Android was chosen as the main target platform and is the only platform GEM is tested on at the moment. However, none of GEM’s dependencies rule out any platforms, meaning that GEM should be easy to port to other platforms with little to no effort, such as UI tweaking.

![Figure 2 GEM Technology Stack](image)

---

\(^3\) [http://leafletjs.com](http://leafletjs.com)

3.2 Installation

GEM is an open source application that is hosted on GitHub under organization GeoKnow. Sources and instructions can be found at https://github.com/GeoKnow/GEM. Automation and package dependency management are controlled via Grunt⁵ and Bower⁶, respectively, with installation/update scripts provided for both Unix-based and Windows operating systems (bower-update.sh and bower-update.bat). Source code is compiled by running the following command:

```
cordova build <platform>
```

where <platform> can be any of the Cordova/Phonegap supported platforms (e.g. android). The application can also be compiled and deployed on a connected device by running:

```
cordova run <platform>
```

Furthermore, to facilitate application take-up, pre-compiled versions for the Android platform in the form of Android application packages (APKs) will be hosted on http://geoknow.imd.bg.ac.rs/gem/apks. These files are analogous to Debian packages in Debian-based operating systems and enable simple, single-tap installation.

3.3 Application Description

GEM’s user interface was designed with the goal of maximizing available space for the slippy map, which represents the base layer and the only stationary component. In addition, there are four widgets that are opened on demand (see Figure 3):

- **Facets slide drawer:** holds the loaded resource facet tree that enables a user to define facets that will be used to filter resources on the map;
- **Source Manager side drawer:** used to: 1) add/edit/remove available SPARQL endpoints / Linked Open Data sources, 2) define properties that will be shown in the resource details widget, 3) edit prefixes, and 4) import a Mappify configuration;
- **Filter text box:** filters resources on the screen and shows the orientation indicator (i.e. compass);
- **Resource details bottom drawer:** used to: 1) display and edit relevant information (selected properties) for the selected feature, 2) add the source, destination, and waypoints to be used for routing, and 3) display navigation instructions.

---

⁵ http://gruntjs.com/
⁶ http://bower.io/
Different icon colors are used to indicate different resources/statuses, while marker clustering was used to prevent the map from overpopulating, and avoid overloading the mobile device resources in situations where multiple features are in each other’s vicinity, as described in [1].

The rest of this section will provide detailed descriptions of aforementioned UI components, with the exception of the Facet slide drawer and Filter text box. General descriptions of these two components are given above, and since there were no changes in their functionality or appearance since the previous release, reader is referred to the previous deliverable [1] for detailed descriptions.

### 3.3.1 Slippy Map

Slippy map represents the main view which is always shown. It displays all resources that are located in the currently viewed area and satisfy selected criteria (active data sources, defined facets, and filter text phrase). Additionally, map is used to select a resource whose details are then shown in the Resource details bottom drawer, which was available in the previous release as well, while the new versions also allows to display a route based on the destination and waypoints selected in the Resource Details view (see Figure 4).

![Figure 4 GEM Slippy Map: a) resource view; b) navigation](image)

### 3.3.2 Source Manager

As described in the previous deliverable [1], source manager can be used to add/remove/edit available data sources, as well as to select which of the available sources will be retrieved/displayed on the map, while the settings are being preserved across sessions by using HTML5 Local Storage. Additionally, the new version enables to import a Mappify configuration, a feature discussed in section 2.3, by tapping the “Add From Mappify” button (see Figure 5a) and entering the URL of the desired configuration. After that, GEM fetches the configuration and based on it initializes a new data source, which then results in corresponding resources being displayed on the map. Furthermore, the new version introduced a catalogue of data sources, which is populated
by querying a dedicated SPARQL endpoint. The catalogue is shown when “Catalogue” button is clicked, after which the user can choose any of the predefined data sources with a single tap.

Additional improvement in the new version of GEM is the possibility to specify which properties are to be shown in the Resource Details bottom drawer upon selecting a resource on the map. Namely, when editing or adding a new source, the user can click on “Properties” which will open a view where it is possible to specify any number of properties (see Figure 5b). New settings are saved by clicking the “Save” button after which, every time a resource on the map is tapped, the Resource details drawer will slide up, and display defined properties for the selected resource.

Source Manager was also extended with the possibility to define prefixes to be used to shorten URIs, thereby making the interface clearer. To manage prefixes, a user needs to click the cog button in the upper right corner of the Source Manager. After that, a list of currently defined prefixes will be shown, where he/she will be able to edit existing prefixes, add new ones by clicking the “Add” button, or remove existing ones by clicking the “Remove” button (see Figure 5c).

![Figure 5 Source Manager: a) Main view; b) Edit properties; c) Edit prefixes](image)

Finally, the Source Manager was also enhanced with automatic resource recommendation to further improve usability. Namely, when editing source properties such as type of resources or properties to be shown in the Resource Details view, the applications automatically provides recommendations based on currently entered text as well as types and properties that are available in the endpoint (see Figure 6).
3.3.3 Resource Details

In the previous version of GEM, the Resource Details widget only showed relevant (pre-defined) properties for the resource selected/tapped on the slippy map. As mentioned earlier in section 3.3.2, this view can now be configured to show exactly those properties a user is interested in. Furthermore, the widget was extended with a feature that allows authoring of displayed information. Authoring is initiated by clicking the “Pencil” button next to the item to be modified (see Figure 7a). After this, input field is shown in the item’s place, and “Pencil” icon changes to “Accept” icon that is to be clicked when the authoring is completed. After this, GEM executes a SPARQL Update query that is generated based on entered content, taking into account the specified type (number, URI, or text).

Moreover, this widget was extended to enable the user to define the destination and waypoints for navigation, as well as to display a textual representation of the calculated route. Navigation is initiated by clicking the “Directions” button in the Resource Details view, while additional waypoints are added by first selecting the desired point of interest (POI) on the map, and then clicking the “plus” button. Selected resource can also be removed from the list of waypoints by clicking the “remove” button (see Figure 7b).
Figure 7 Resource Details: a) Information view; b) Directions view

3.4 Application Defaults

In order to facilitate take-up, the application is initialized with certain defaults, so that a user doesn’t have to configure anything in order to be able to try out its features. Source catalogue is by default initialized with sources representing architectural structures from DBpedia, all resources from Serbian DBpedia, and bars from LinkedGeoData. Furthermore, architectural points of interest from the DBpedia data source are displayed on the map when the user opens the app for the first time. Finally, in order to prevent cluttering in the user interface caused by long URIs, the application is by default initialized with the most common prefixes, such as skos, dct, dbpedia, rdf, rdfs, owl, etc.
4.E-Commerce Use Case

This section provides insights into an additional, more specialized application of GEM. In WP6, Unister computed geospatial motives that enable to visually identify regions according to the relevance of different motives, based on the concentration of respective entities. Polygons of these regions can be displayed on the map, thereby enabling to easily identify regions of interest, such as regions with high density of cultural venues and tourist attractions. Moreover, this information can be combined with GEM’s faceted browsing capabilities to show various points of interest on top of the polygons. These points of interest could be used either to validate polygons by showing entities that were used for their generation, or to introduce another dimension in our analysis.

First, let’s take a look at the validation example. By plotting the theatres motive along with entities of type Theatre from the LinkedGeoData dataset, we can validate that the generated polygon indeed represents a theatre-packed area (see Figure 8). Now, let’s say Unister wanted to build a hotel that would target people with preference for theatres and tourist attractions. This could be achieved by displaying polygons for theatres and tourist attractions along with entities of type Hotel from the LinkedGeoData dataset. In this case, the intersection of the two polygons provides us with an area that would satisfy preferences of our target clientele, while the hotels acquired from LinkedGeoData enable us to find areas where competition presence is low, thus indicating the perfect spot for our new hotel (see Figure 9).
Figure 9 Finding a perfect spot for a new hotel
5. Conclusions

The GEM prototype described in this deliverable 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 WP4, 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, and Leaflet Routing Machine. It is deployed using Apache Cordova / Adobe Phonegap, thus allowing single-branch development for multiple major mobile platforms.

The final version was extended with many new features. The user interface is clearer thanks to prefix management, while users can now also take active role in the crowdsourcing efforts by authoring content. Furthermore, GEM is now integrated with Mappify in a sense that configurations created on a desktop computer using Mappify can now be imported into GEM, thereby enabling to pre-define views on the desktop and then re-use them while on the go. Most importantly, GEM is the first ever application to support semantic routing, where potential destination and waypoints can be filtered using semantics prior to their selection and inclusion in the route.

Future work will focus mainly on extending options for routing/navigation and creating additional services around Mappify configurations, such as exporting and sharing configuration from GEM to other mobile devices. Furthermore, we intend to explore different applications for GEM, from e-commerce to crowdsourced crisis management.
6. References

