A Framework with Tools for Designing Web-based Geographic Applications

The Nhan Luong, Sébastien Laborie, Thierry Nodenot

A Framework with Tools for Designing Web-based Geographic Applications

The Nhan Luong, Sébastien Laborie and Thierry Nodenot
T2i – LIUPPA – Université de Pau et des Pays de l'Adour
2 Allée du Parc Montaury, 64600 Anglet, France
{thenhan.luong, sebastien.laborie, thierry.nodenot}@iutbayonne.univ-pau.fr

ABSTRACT

Many Web-based geographic applications have been developed in various domains, such as tourism, education, surveillance and military. However, developing such applications is a cumbersome task because it requires several types of components (e.g., maps, contents, indexing services, databases) that have to be assembled together. Hence, developers have to deal with different technologies and application behavior models. In order to create Web-based geographic applications and overcome these design problems, we propose a framework composed of three complementary tasks: identifying some desired data, building the graphical layout organization and defining potential user interactions. According to this framework, we have specified a unified model and we have encoded it using Semantic Web technologies, such as RDF. Through a prototype named WINDMash, we have implemented some tools that instantiate our model and automatically generate concrete Internet geographic applications that can be executed on Web browsers.

Categories and Subject Descriptors

D.2 [**Software Engineering**]: Design Tools and Techniques; I.7.2 [**Document and Text Processing**]: Document Preparation.

General Terms

Design, Languages.

Keywords

Document generation, Authoring tool, Mashups.

1. INTRODUCTION

Currently, a fair amount of research and development has been conducted on Web-based application generation thanks to Web 2.0 technologies. Particularly in the domain of geographic information system (GIS), specific terms appeared in order to designate Internet Geographic Applications [10]: "GeoWeb", "Geospatial Web" and "Web Mapping 2.0".

Indeed, many Web-based geographic applications have been developed in different application domains (e.g., tourism, education, surveillance, military) and are using online mapping services (e.g., Google Maps[1], OpenLayers[2], Yahoo! Maps[3]).

[1] `http://maps.google.com`
[2] `http://openlayers.org`
[3] `http://maps.yahoo.com`

However, developing such applications is a cumbersome task. The reasons are twofold: (1) they mix several components (e.g., maps, multimedia contents, indexing services, databases) which have to interact together and (2) developers have to deal with several technologies and different data structures as well as application models.

In this paper, we propose a framework for designing Web-based geographic applications. This framework is composed of three complementary tasks: (1) identifying the desired data that have to be manipulated by the system, (2) specifying the graphical layout of the application, and (3) defining potential end-user interactions between components. According to this framework, we have specified a unified model with three facets and proposed to encode this model using RDF [16]. We have chosen such a W3C standard because it enables to describe, to aggregate, to share and to reuse information embedded into each facet.

In order to experiment our proposal, we have developed some tools allowing designers to easily instantiate our unified model, i.e., create Web-based geographic applications. Moreover, from our proposed model, the designed descriptions can be computed into concrete Internet geographic applications and finally executed on a Web browser.

The remainder of the paper is structured as follows. In Section 2, we define the characteristics of a Web-based geographic application and illustrate an example. Moreover, we explain why it is currently difficult to design this kind of application. Thereafter, in Section 3, we describe some related existing systems. In Section 4, we specify our framework for designing Web-based geographic applications and detail in Section 5 its corresponding unified model. Section 6 demonstrates our framework through a prototype named WINDMash, especially some specific tools are presented in Section 7. Finally, we conclude in Section 8.

2. DESIGNING WEB-BASED GEOGRAPHIC APPLICATIONS

Designing interactive Web-based geographic applications that allow, for instance, completing and/or increasing the discovery of a territory, such as in tourism or in education, is currently a cumbersome task. Indeed, such applications may generally include several components that should interact with each other, such components are:

- Mapping components with, for instance, spatial annotations and various types of geovisualization (e.g., street map, satellite view);
- Content components, such as texts, images, audios and

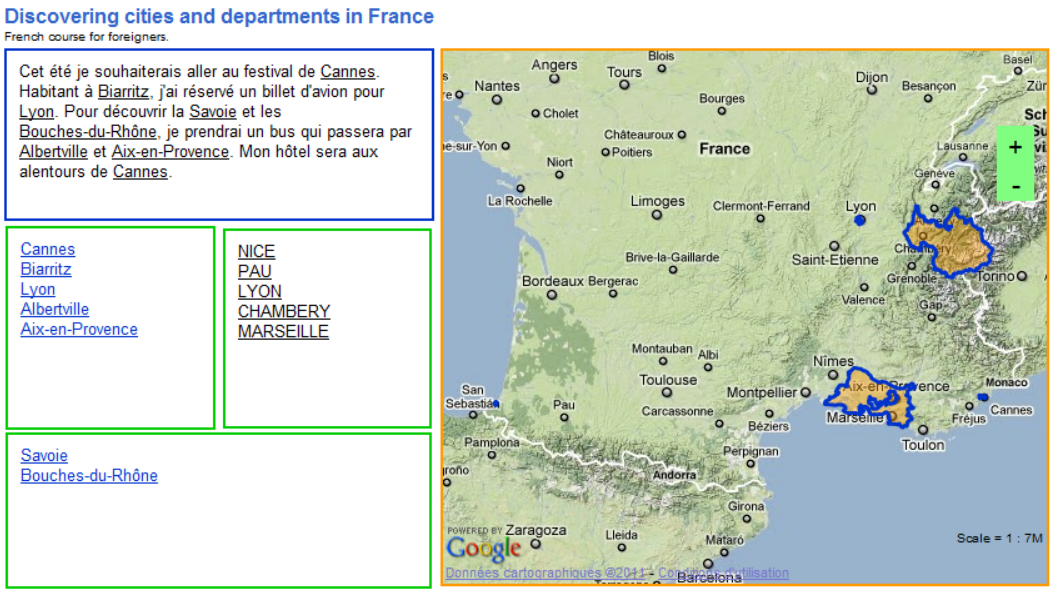

Figure 1: A Web-based geographic application for discovering French places.

videos. In this paper, we will exclusively focus on textual components;

- Indexing components which automatically extract spatial entities from some given contents and identify spatial metadata, like geolocation information;

- Communication components in order to exchange data with database servers, e.g., GeoNames[4].

In order to illustrate a typical Web-based geographic application, we present a use-case scenario in the following section (§2.1). This example will be used all during the paper. Furthermore, we also explain why developing such applications is not straightforward (§2.2).

2.1 A use-case scenario

Suppose a French course for foreigners where a teacher wants to explain the following concepts: Town, "Département" (a sort of province inside the country) and "Préfecture" (an administrative town inside a "Département"). More precisely, the teacher wants students to study a text written in French[5] which contains several places names, e.g., Cannes (Town), Lyon ("Préfecture") and Savoie ("Département").

For pedagogical purposes, the teacher would like a Web-based geographic application, like the one illustrated in Figure 1. Indeed, it contains the text in French, a map containing spatial annotations and three lists of towns, "Préfectures" and "Départements". These lists should be automatically computed from the text. Furthermore, students are allowed to select a specific place in the lists. When clicking on a place, the corresponding term in the text is highlighted and the map is focused on this place.

[4] `http://www.geonames.org`

[5] The English version of the text in Figure 1 is available at the following address: `http://www.iutbayonne.univ-pau.fr/~slaborie/DocEng11/EnglishTranslation.txt`

2.2 Some requirements

Developing the interactive Web-based geographic application illustrated in Figure 1 with the behaviors described above is not a trivial task. It requires:

- Some programming skills, e.g., using JavaScript or AJAX, in order to create an interactive Web application;

- Knowledge about several databases schemas, and especially geographical databases in order to query and get spatial data, such as geolocations on a map;

- Manipulation of Web services (e.g., indexing services), particularly their inputs/outputs. Naturally, data structures have to be homogenized in order to confront and/or aggregate different services outputs.

Consequently, there is a need of a general framework which contains models and supports some tools for designing Web-based geographic applications. This framework is even more motivated in the next section.

3. RELATED WORK

End-user mashup programming environments are a new generation of online visual tools enabling users to quickly create, for example, Web-based applications [3]. They rely on metaphors that are easy to grasp by non-professional coders. They may bind together spreadsheets, the flow of linked processing blocks and the visual selection of GUI actions. In [5] and [21] several types of mashup environments have been summarized, some examples are: Yahoo! Pipes[6], Microsoft Popfly[7], Google Mashup Editor[8], IBM Mashup Center[9], MashMaker [8], Marmite [23], Vegemite [14], Exhibit [13] and Bill Organiser Portal [18].

[6] `http://pipes.yahoo.com/pipes/`

[7] `http://www.deitel.com/Popfly/`

[8] `http://code.google.com/intl/en/gme/`

[9] `http://www-01.ibm.com/software/info/mashup-center/`

However, developing the geographic application example presented in Section 2 with these mashup environments is still difficult, while impossible with some of these systems. In fact, many of them do not take into account the specification of end-user interactions and especially the system reactions on the different application components. Furthermore, these systems are not designed to exclusively build geographic applications, hence they do not propose a framework for building such kind of applications.

Some recent works on Web Engineering, especially focusing on geographic applications, proposed architectures based on mapping services. For instance, Mashlight [2] is a Web framework for creating and executing mashup applications by building data processing chains. The generated applications contain by default one mapping component with geolocation information. Besides, DashMash [6] offers to end-users more flexibility for creating geographic applications. Through a set of viewers, they can specify a graphical layout and visualize specific data inside them. Nevertheless, these systems do not allow to design end-user interactions.

SPARQLMotion[10] is a visual scripting language and an engine for semantic data processing. Scripts implementing sophisticated data services and processing, such as queries, data transformations and mashups, can be quickly assembled with user-friendly graphical tools. Such a system is using Semantic Web technologies, like RDF [16] and SPARQL [17]. However, as the previous systems, it does not provide solutions to specify end-user interactions and a general framework for constructing geographic application.

PhotoMap [22] offers graphical interfaces for navigation and query over some photo collections of users. Especially, with the map-based interface, itineraries followed by users are illustrated and photos with information are attached to specific places. Hence, end-users are able to retrieve where some photos have been taken either by clicking on specific places or by selecting a group of photos. This environment is another use-case for illustrating Web-based geographic applications. Nevertheless, this environment has been hard-coded by an expert and no design framework has been proposed by the authors.

In the next section, we propose a new framework for designing Web-based geographic applications. It should be considered to overcome the limits of the tools presented above.

4. A FRAMEWORK FOR DESIGNING WEB-BASED GEOGRAPHIC APPLICATIONS

To design a Web-based geographic application, we propose the framework depicted in Figure 2. This framework is composed of three complementary tasks:

1. Identifying the desired data that have to be manipulated by the geographic application. The data could refer to multimedia contents[11] (e.g., the text written in French in Figure 1), some extracted information (e.g., in Figure 1, the list of towns automatically identified from the text) and some computed information (e.g., in Figure 1, the lists of "Départements" and "Préfectures" which correspond to the list of towns).

2. Specifying the graphical layout of the geographic application. This layout may be composed of several viewers, such as textual viewers, list viewers or map viewers. Thanks to the data that have been defined during the previous step, viewers might display some data sets. For instance, if some data about towns are determined, these towns will be displayed on a specific map viewer.

3. Defining potential end-user interactions on the data that are contained inside viewers. For instance, if a user clicks on a specific town listed in Figure 1, this town will be highlighted in the text written in French and the map will be focused on this town.

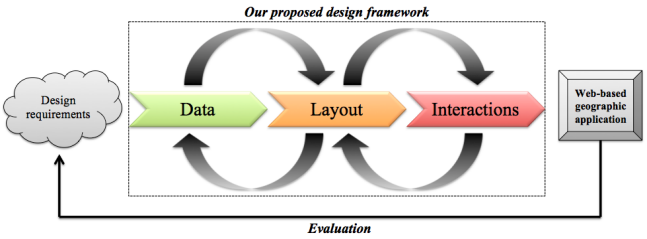

Figure 2: Our framework for designing Web-based geographic applications.

As illustrated in Figure 2, even if we have ordered the three tasks, it is possible to go backward and forward during the design process. Actually, at any time the application designer may add, modify and/or remove some data, some viewers or some interactions. Furthermore, it is also possible to design a geographic application without interactions.

When all tasks have been completed, the Web-based geographic application can be generated and executed. Obviously, it is possible to update the generated application by repeating our proposed framework.

According to the framework presented in this section, for each task illustrated in Figure 2, we have specified a model and we have encoded it using RDF [16]. Details and examples related to the use-case described in Section 2.1 are presented in the next section.

5. A UNIFIED MODEL FOR DESCRIBING AND COMPUTING WEB-BASED GEOGRAPHIC APPLICATIONS

According to our framework proposed in Figure 2, we have defined a unified model for describing Web-based geographic applications. This model is structured into three facets:

- **The data facet** (§5.1) describes the data that are manipulated by the application.
- **The graphical interface facet** (§5.2) defines the organization of the application layout.
- **The user interaction facet** (§5.3) specifies potential human-computer interactions.

Our unified model corresponds to the merge of all facet descriptions. We particularly show that these facets are complementary through the concept of *Annotation*. Furthermore, in order to encode, to aggregate, to share and to reuse

[10] `http://www.topquadrant.com/products/SPARQLMotion.html`

[11] In this paper, we only consider textual contents.

each facet description, we have proposed an RDF/XML serialization [4]. Thanks to this standard, each facet can be described independently and may refer to other facet descriptions. Moreover, semantic queries can be executed on these descriptions.

Each facet will be described and exemplified in the following subsections.

5.1 The data facet

As presented in Section 2, a Web-based geographic application manipulates data. In this paper, we consider two categories of data: *Content* and *Annotation*. The former refers to multimedia contents that might be composed of several segments, while the latter concerns the description of named entities (e.g., a person, a location).

As illustrated in Figure 3 (upper part), a content may hold several annotations referring to specific segments. Furthermore, an annotation may be connected with other annotations through different properties.

In this paper, we are considering exclusively textual contents inside a geographic application. Consequently, as presented in Figure 3 (lower part), a text may be segmented into paragraphs and/or tokens. Moreover, annotations will correspond to geographic information, e.g., places. [9] stated that geographic information is composed of three complementary features: temporal, spatial and thematic. We followed the same scheme, however the examples in the paper will focus only on the spatial feature.

Figure 4 presents an RDF/XML sample that corresponds to the data facet described in Figure 3. This description indicates that the file `text.txt` (i.e., the text written in French in Figure 1) contains an annotation (`Annotation1`) about an entity named `Cannes`. This geographic entity is located in the first paragraph of the text at the 9th token. Moreover, a `geolocation` and a `geotype` are associated to this entity.

Naturally, the description of Figure 4 could be enhanced with additional information. For instance, it may express that Nice, which is another spatial entity, is the "Préfecture" of Cannes. Since Nice does not appear in the file `text.txt`, this entity will not be related to a specific segment of the text. Furthermore, the description of Figure 4 may describe other spatial entities contained in the text, such as Biarritz or Lyon.

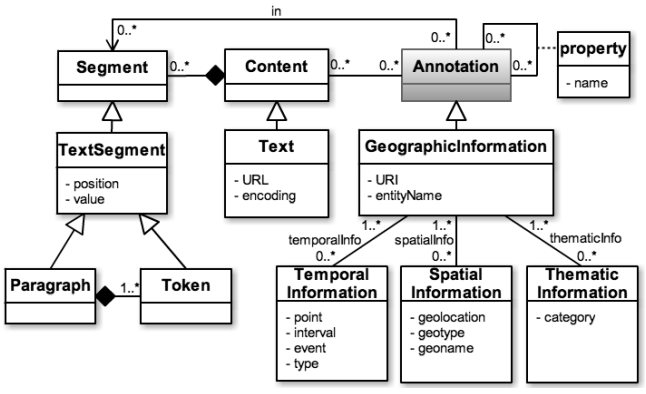

Figure 3: The data facet.

```
<rdf:RDF>
 <wm:Content rdf:about="&ex;text.txt">
  <wm:annotation>
   <wm:GeographicInformation rdf:about="&ex;
       data.rdf#Annotation1">
    <wm:entityName>Cannes</wm:entityName>
     <wm:in>
      <rdf:Description>
       <wm:start rdf:resource="&ex;text.txt#
           Par1-Token9"/>
       <wm:end rdf:resource="&ex;text.txt#Par1-
           Token9"/>
      </rdf:Description>
     </wm:in>
     <wm:spatialInfo>
      <wm:SpatialInformation rdf:about="&
          geotopia;#Cannes">
       <wm:geolocation>MULTIPOLYGON(...)</wm:
           geolocation>
       <wm:geoname>Cannes</wm:geoname>
       <wm:geotype rdf:resource="&ex;data.rdf#
           Town"/>
      </wm:SpatialInformation>
     </wm:spatialInfo>
    </wm:GeographicInformation>
    ...
  </wm:annotation>
 </wm:Content>
</rdf:RDF>
```

Figure 4: An RDF/XML sample of the data facet.

5.2 The graphical interface facet

As illustrated in Figure 1, a Web-based geographic application contains a graphical user interface (GUI). This one may be composed of several viewers organized in the presentation layout. In this paper, we consider three categories of viewers: `TextViewer`, `MapViewer` and `ListViewer`. For instance, in Figure 1 there are a `TextViewer` which contains the text written in French, a `MapViewer` which includes Google Maps and three `ListViewer`s enumerating some places.

Figure 5 presents our GUI facet. In this figure, each viewer may embed several annotations. More precisely, spatial entities may be highlighted in a text, displayed on a map or enumerated in a list. In fact, segments associated to annotations are used to locate entities, geolocations are useful for maps and geonames may be employed in lists.

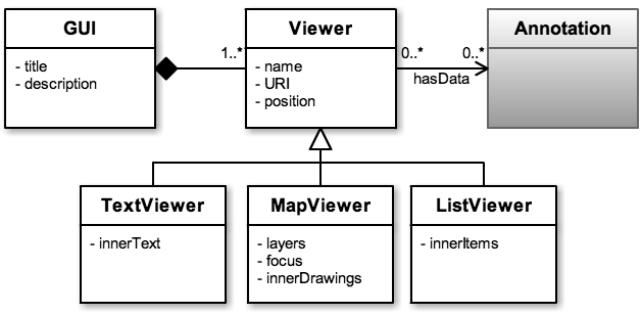

Figure 5: The GUI facet.

Figure 6 presents an RDF/XML sample that conforms to the GUI facet that we proposed in Figure 5. This description shows that the GUI contains a map viewer with a position specified by its top, left, height and width. This viewer embeds an annotation, i.e., `Annotation1`. Since the RDF merge operation has been clearly defined[12], from the RDF/XML samples of Figure 4 and 6, it is possible to display on the map the geolocation of Cannes.

```
<rdf:RDF>
 <rdf:Description rdf:about="&ex;gui.rdf">
  <wm:title>French course</wm:title>
  <wm:viewers>
   <rdf:Bag">
    <rdf:li>
     <rdf:Description rdf:about="&ex;gui.rdf#
         Viewer1">
      <rdf:type rdf:resource="&ex;MapViewer"/>
      <wm:width>400</wm:width>
      <wm:height>300</wm:height>
      <wm:left>300</wm:left>
      <wm:top>20</wm:top>
      <wm:hasData>
       <rdf:Seq">
        <rdf:li rdf:resource="&ex;data.rdf#
            Annotation1"/>
       </rdf:Seq>
      </wm:hasData>
     </rdf:Description>
    </rdf:li>
    ...
   </rdf:Bag>
  </wm:viewers>
 </rdf:Description>
</rdf:RDF>
```

Figure 6: An RDF/XML sample of the GUI facet.

5.3 The user interaction facet

A Web-based geographic application may be composed of several user interactions. For instance, in Figure 1 when a user selects a specific `Town` with a mouse click, this entity is highlighted in the text and the map zoom in on it. In [7] and [20], authors have formalized a vocabulary for defining user interactions, especially they are using the following concepts: *Event* and *Action*. An *Event* is generally characterized by two elements [20]: the type of event (e.g., click, double-click, mouseover) and where the event occurred (e.g., on one entity in a viewer). An event may trigger several *Actions*, e.g., it might be a set of visual effects in the graphical user interface, such as highlight and zoom.

Figure 7 presents the user interaction facet which is inspired from [7]. In this figure, events and actions occur on annotations. Consequently, thanks to the data and GUI facets proposed respectively in Figure 3 and 5, we could retrieve which viewer contains such annotation, its geotype, its geolocation and so on.

Figure 8 presents an RDF/XML sample that corresponds to the user interaction facet described in Figure 7.

In Figure 8, a potential end-user interaction `I1` is defined. This interaction contains an event `Evt1` and an action `Action1`. This description explains that if a user clicks on any annotations about `Town` in `Viewer2`, the `Viewer1` will zoom in on these annotations. Of course, several interactions could be defined and they may refer several times to the same annotations or the same viewers.

Note that the concept of *Annotation* has an important role, especially for designing a Web-based geographic application, because it allows to link the data, the GUI and the user interactions facets.

[12]http://www.w3.org/TR/rdf-mt/

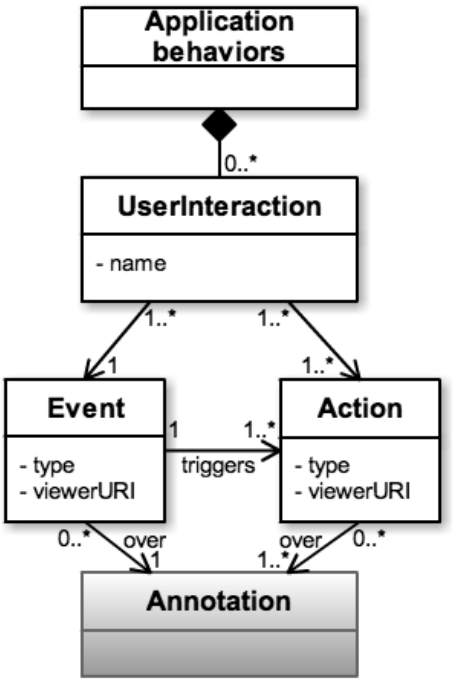

Figure 7: The user interaction facet.

```
<rdf:RDF>
 <rdf:Description rdf:about="&ex;hci.rdf">
  <wm:contains>
   <rdf:Bag>
    <rdf:li>
     <rdf:Description rdf:about="&ex;hci.rdf#I1">
      <wm:event>
       <rdf:Description rdf:about="&ex;hci.rdf#Evt1">
        <rdf:type rdf:resource="&ex;Click"/>
        <wm:over rdf:resource="&ex;data.rdf#Town"/>
        <wm:via rdf:resource="&ex;gui.rdf#Viewer2"/>
        <wm:triggers rdf:resource="&ex;hci.rdf#
            Action1"/>
       </rdf:Description>
      </wm:event>
      <wm:action>
       <rdf:Description rdf:about="&ex;hci.rdf#
           Action1">
        <rdf:type rdf:resource="&ex;Zoom"/>
        <wm:over rdf:resource="&ex;data.rdf#Town"/>
        <wm:in rdf:resource="&ex;gui.rdf#Viewer1"/>
       </rdf:Description>
      </wm:action>
     </rdf:Description>
    </rdf:li>
   </rdf:Bag>
  </wm:contains>
 </rdf:Description>
</rdf:RDF>
```

Figure 8: A user interaction encoded in RDF/XML.

6. THE WINDMASH PROTOTYPE

We have implemented our proposal in a prototype named WINDMash[13]. This one allows designers to create and to automatically generate interactive Web-based applications handling geographic information. Figure 9 presents the architecture of our prototype.

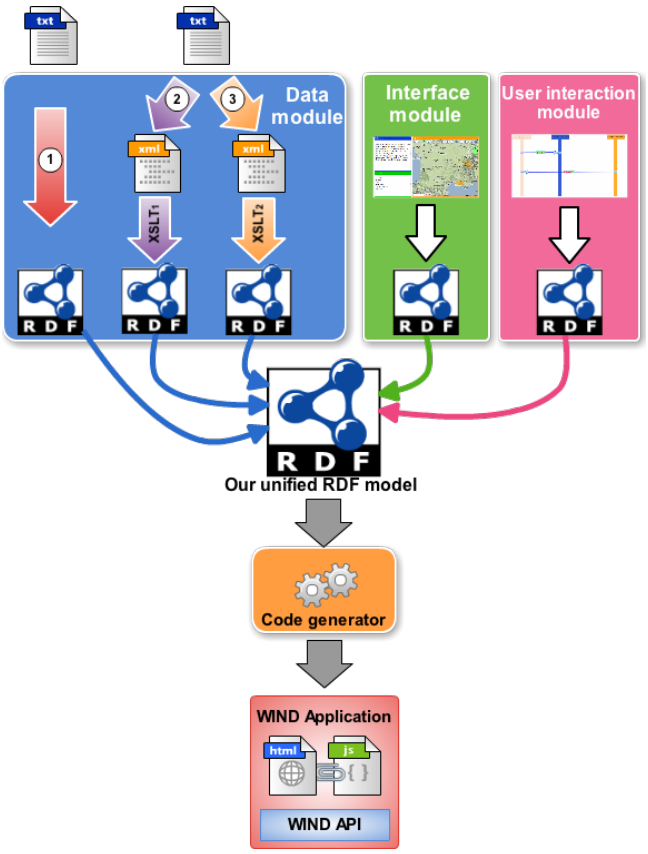

Figure 9: The WINDMash architecture.

As illustrated in Figure 9, our prototype is composed of three modules (i.e., Data, Interface and User interaction) which correspond to our framework proposed in Figure 2. Each module manipulates some RDF/XML descriptions which comply with the model presented in Section 5. Naturally, the merge of all RDF/XML descriptions corresponds to an instance of our unified model which is used to compute and to generate the Web-based geographic application.

In this paper, we only focus on textual contents handled by our prototype. Hence, the data module may invoke several textual indexing services. For instance, the service ① in Figure 9 could directly produce an RDF/XML output complying the data facet proposed in Section 5.1. Besides, other textual indexing services, like the services ② and ③, might be executed. For instance, we have used a service that automatically extracts places from a text and collects places' geolocations [19]. However, the Web service implemented in [19] produces XML outputs that do not correspond to the data facet proposed in Section 5.1. Consequently, we have implemented an XSLT stylesheet in order to translate their XML outputs into some RDF/XML descriptions complying with our data facet.

In order to compute a Web-based geographic application from our unified model, we have implemented a code generator module written in PHP that produces XHTML Web pages containing JavaScript instructions. This module is using the RAP[14] software package (RDF API for PHP) in order to analyze the RDF descriptions. Indeed, this package allows to query the descriptions with the SPARQL query language [17]. Moreover, we have used our WIND[15] API (Web INteraction Design) which is fully described in [15]. This API enables to describe interactions between components (e.g., map, text) without having to know anything about the underlying Web Mapping Services (IGN Geoportail API, Google Maps API, OpenLayers API, etc.).

In the next section, we will present several tools of our WINDMash prototype which allow designers to instantiate our unified model with its three facets, i.e., data, GUI and user interaction.

7. THE WINDMASH TOOLS FOR DESIGNING WEB-BASED GEOGRAPHIC APPLICATIONS

As illustrated in Figure 9, the WINDMash architecture is composed of three modules which concern, respectively, the data management, the GUI organization and the user interactions specification.

Consequently, three tools have been implemented in our WINDMash prototype for instantiating our unified model presented in Section 5. More precisely, these tools are:

1. A pipe editor which allows to combine different services and to filter the data manipulated by the application (§7.1);

2. A graphical layout editor which is used to arrange, for instance, mapping components or multimedia contents (§7.2);

3. A UML-like sequence diagram builder which allows to specify potential end-user interactions on the application (§7.3).

In the remainder, each tool listed above is presented. Moreover, these tools are illustrated with screenshots based on the use-case scenario presented in Section 2.1.

7.1 Our pipe editor

In order to manage the data (i.e., contents and annotations) that should be manipulated by the Web-based geographic application, we have developed a pipes editor. This tool has been inspired from the Yahoo! Pipes editor[16] and enables designers to create a processing chain containing different services.

Figure 10 illustrates a possible processing chain which corresponds to the use-case presented in Section 2.1. Actually, from a given text (A), we would like to extract automatically

[13] http://erozate.iutbayonne.univ-pau.fr/Nhan/windmash3/

[14] http://www4.wiwiss.fu-berlin.de/bizer/rdfapi/

[15] http://erozate.iutbayonne.univ-pau.fr/Nhan/windapi/wind.html

[16] http://pipes.yahoo.com

some places (B). For that purpose, we invoke the `PlaceExtraction` Web service which had been detailed in [19]. Thanks to the identified places (B), we have defined two sets of annotations: towns (C) and "Départements" (D). Since the `PlaceExtraction` service classifies extracted entities, we have implemented some filters (i.e., `Town` and `Department`) in order to select the suitable entities. Finally, from the set of towns (C), we needed to get all corresponding "Préfectures" (E). Thanks to available geographical database (e.g., GeoNames, France IGN Database), the `Prefecture` service computes the appropriate list of "Préfectures" (E) from the list of towns (C).

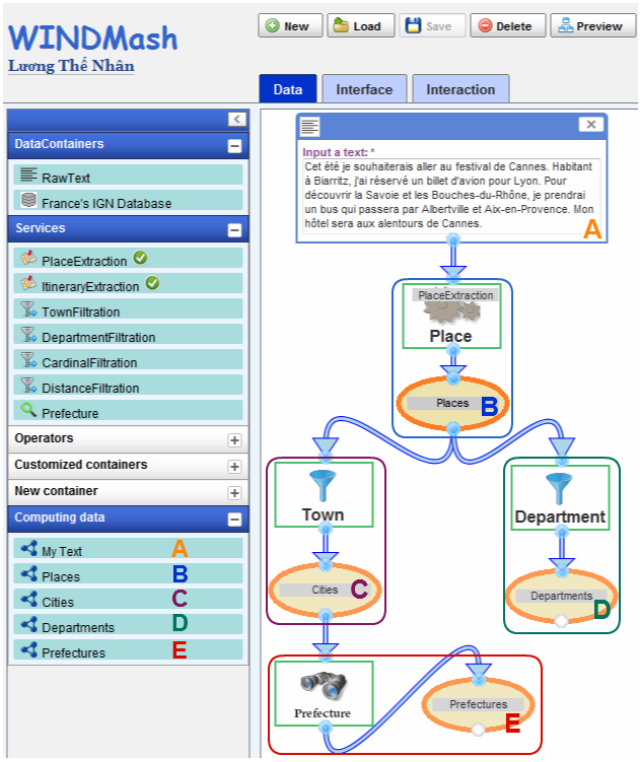

Figure 10: The WINDMash pipes editor.

While constructing the pipes, it is possible to visualize at anytime the computed data by selecting with a double click the B, C, D and E items in Figure 10. Some services may also be customized according to specific needs, e.g., extract places that are situated in a given area. Furthermore, the given text may be updated for computing new datasets.

Each time a dataset is computed, such as the list of extracted places (B), it generates an RDF/XML description which corresponds to the data facet presented in Section 5.1. These descriptions are also accessible at the bottom left of the prototype in Figure 10.

We show in the next subsection, that these descriptions will be used in our graphical layout editor in order to display the data inside viewers.

7.2 Our graphical layout editor

Our graphical layout editor enables a designer to specify the graphical user interface of his/her Web-based geographic application. Indeed, the designer decides which type of viewer he/she wanted in his/her application (e.g., `TextViewer`, `MapViewer`, `ListViewer`) and how these viewers are organized inside the graphical layout.

Figure 11 illustrates how a designer may specify, with our tool, the graphical layout of the use-case presented in Figure 1. The menu on the left hand-side indicates the type of viewers that may be manipulated by the designer, the available dataset that have been computed with our pipes editor (Section 7.1) and the viewers that are currently used. In Figure 11, five viewers have been specified: a text viewer (1), a map viewer (2) and three list viewers (i.e., 3, 4 and 5).

Initially, when the designer drags and drops a viewer inside the graphical layout, this viewer is empty, except the map viewer which contains a map. If the designer wants to display some information inside viewers, from the menu, he/she has to drag the computed datasets and to drop them inside a specific viewer. For instance, if the designer wants to see inside the text viewer (1) the text that has been written in Figure 10, he/she has to drag from the menu the `My Text` element and to drop this element inside the `My Text Viewer 1` viewer. Thereafter, if the designer wants to underline the places that have been extracted from this text, he/she has to drag the `Places` dataset from the menu and to drop it inside the `My Text Viewer 1` viewer.

Similarly with the other types of viewers, if the designer wants to see the places on the map, he/she has to drag the `Places` dataset and to drop it inside the `My Map Viewer 2` viewer. Finally, if the designer wants to see the lists of towns, "Départements" and "Préfectures" that have been defined in Figure 10, he/she has to drag the datasets and to drop them inside the list viewers, i.e., `My List Viewer 3`, `My List Viewer 4` and `My List Viewer 5`.

Of course, it is possible to customize each viewer. For instance, the style of a text inside a text viewer may be modified, such as its font, its size, etc. Furthermore, the type of the lists may also be changed, e.g., with or without different kinds of bullets. Finally, different types of maps may be used, such as Google Maps, Yahoo! Maps, IGN Maps[17]...

In the next subsection, we present how it is possible to specify potential end-user interactions using the layout components that have been designed in this section.

7.3 A "sequence diagram" builder for specifying end-user interactions

As required in the use-case scenario proposed in Section 2.1, when a user selects a specific place in the lists, the map should focus on this place and, if possible, the corresponding terms in the text should be highlighted. [12] proposed to design this kind of interaction through UML-like sequence diagrams.

Effectively, these diagrams are useful for designers because:

- they are based on few concepts to describe the interactions between a user and the system;
- they clearly distinguish events made by a user on the system (i.e., arrows from the user to the system) and the potential actions of the system to the user (i.e., arrows from the system to the user);

[17] `http://www.geoportail.fr`

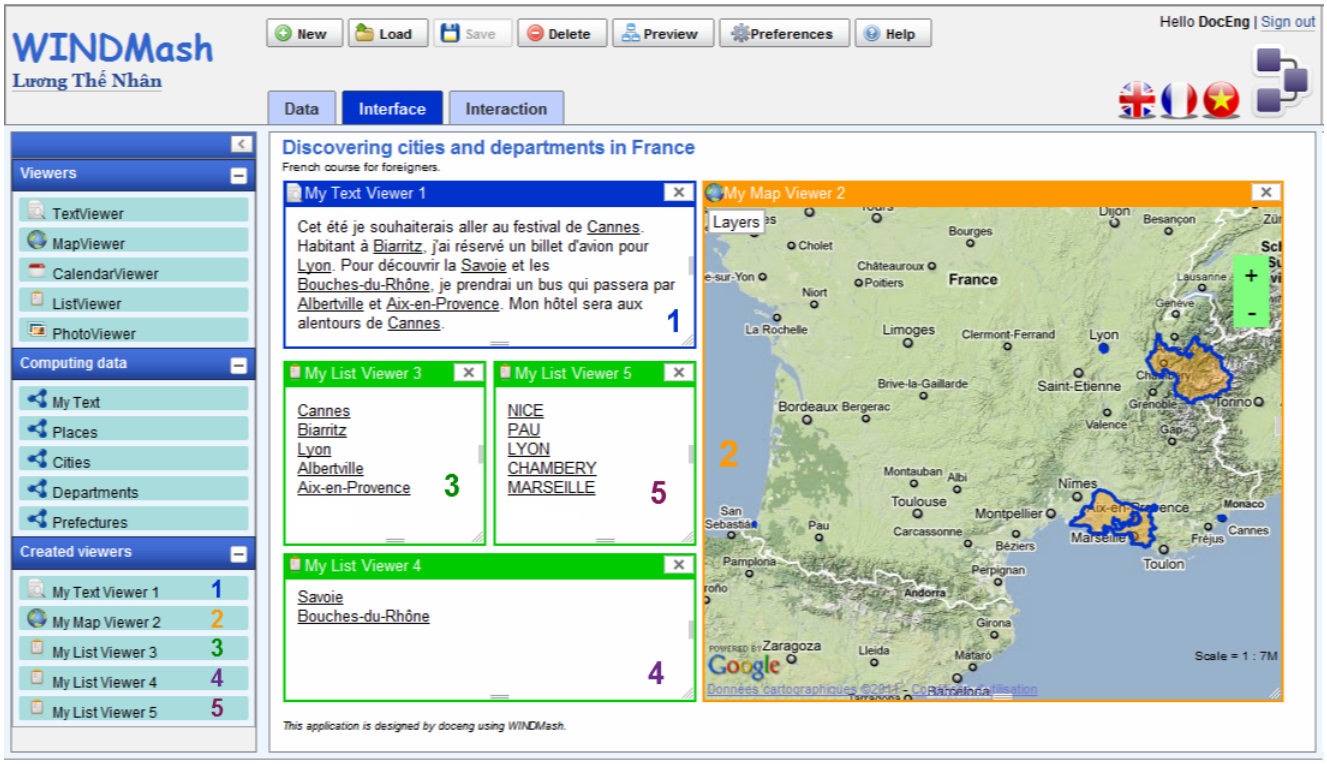

Figure 11: The WINDMash graphical layout editor.

- they can both describe the interactions between the user and the system, as well as the interaction between the components of the system.

Consequently, we have implemented a UML-like sequence diagram builder which is illustrated in Figure 12[18]. On the left hand-side of this figure, the menu is composed of the list of viewers that have been specified in Section 7.2 and the list of datasets that have been defined in Section 7.1.

The sequence diagram example illustrated in Figure 12 describes the following interaction: When a user selects, through a `click` (see the right arrow), a town contained in the viewer named `My list Viewer 3`, this town is highlighted (see the left arrow labeled with the term `highlight`) in the viewer named `My Text Viewer 1`. Moreover, the viewer `My Map Viewer 2` also zooms in on this town (see the left arrow labeled with the term `zoom`).

When the designer has finished to build the sequence diagram, the WINDMash prototype handles a global RDF/XML description which complies with the unified model presented in Section 5. Consequently, from this description and our code generator module, the designer could preview the Web-based geographic application by selecting the `Preview` button (see the buttons on top of Figure 12). If the generated application suits his/her needs, the designer may save the application and/or deploy it on a specific client. Otherwise, the designer may come back to the three WINDMash tools presented in this section (i.e., the figures 10, 11 and 12) in order to add, to modify or to remove some application elements.

From the WINDMash tools presented in this section and, especially the examples illustrated in the figures 10, 11 and 12, the complete generated Web-based geographic application is available at `http://erozate.iutbayonne.univ-pau.fr/Nhan/windmash3/demo/Frenchcourse/app.html`. A video screencast of our WINDMash prototype is also available at `http://erozate.iutbayonne.univ-pau.fr/Nhan/windmash3/demo/Frenchcourse/video.html`

8. CONCLUSION

In this paper, we have presented a framework dedicated to the design of Web-based geographic applications. This framework addresses three complementary tasks which concern the data manipulated by the application, the graphical layout and the user interactions. We have shown through our unified model for describing such kind of applications that annotations are central in the design process. Indeed, they can be used to describe entities, to display information inside viewers and to specify application behaviors. Furthermore, our proposed framework has been implemented in an online prototype named WINDMash. This prototype contains different tools that facilitate the instantiation of our unified model and automatically generates an executable Web-based geographic application.

Currently, our prototype only deals with textual contents. However, the data model presented in this paper is sufficiently generic to be extended in order to deal with multimedia contents, such as videos, audios and images. Furthermore, the geographic information manipulated by our

[18]Currently, this UML-like sequence diagram builder only allows to create lifelines and to specify messages between them via arrows.

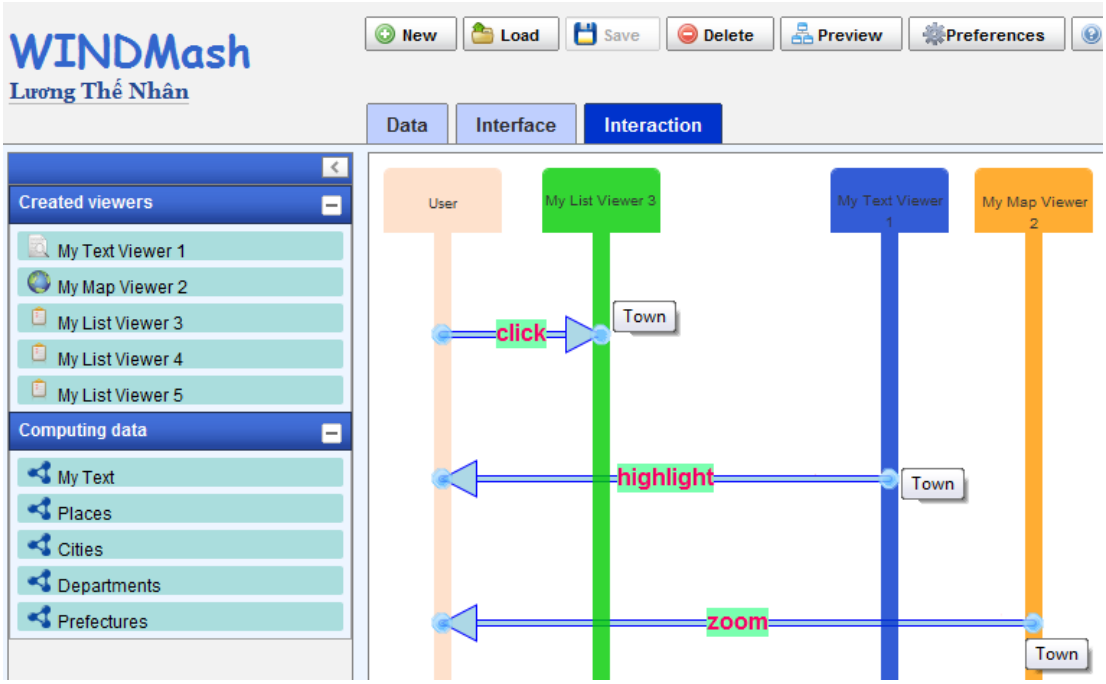

Figure 12: The WINDMash sequence diagram builder.

WINDMash prototype can be imported from other repositories, for example from the LinkedGeoData[19] which exploits the spatial information collected by OpenStreetMap[20]. Hence, a future work would consist in developing a mediator between the imported LinkedGeoData datasets and our unified model. Moreover, we plan to extend our model in order to import non-geographic information, e.g., specific manual annotations.

Finally, we plan to export the generated Web-based geographic applications in XHTML+RDFa [1] instead of XHTML. XHTML+RDFa export will enable WINDMash generated applications to take more advantage of the Semantic Web and, especially the Linking Open Data cloud [11].

9. ACKNOWLEDGMENT

This work has been supported by the French Aquitaine Region (project number 20071104037), the Département of Pyrénées-Atlantiques ("Pyrénées : Itinéraires Éducatifs" project) and the ANR MOANO project (`http://moano.liuppa.univ-pau.fr`).

10. REFERENCES

[1] Adida, B., Birbeck, M., McCarron, S., and Pemberton, S. RDFa in XHTML: Syntax and Processing. Recommendation, W3C, October 2008. `http://www.w3.org/TR/rdfa-syntax/`.

[2] Albinola, M., Baresi, L., Carcano, M., and Guinea, S. Mashlight: A Lightweight Mashup Framework for Everyone. In *Proceedings of the 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web* (2009).

[3] Altinel, M., Brown, P., Cline, S., Kartha, R., Louie, E., Markl, V., Mau, L., Ng, Y.-H., Simmen, D., and Singh, A. Damia: a data mashup fabric for intranet applications. In *Proceedings of the 33rd International Conference on Very Large Data Bases* (2007), VLDB'07, pp. 1370–1373.

[4] Beckett, D., and McBride, B. RDF/XML Syntax Specification (Revised). Recommendation, W3C, February 2004. `http://www.w3.org/TR/rdf-syntax-grammar/`.

[5] Beletski, O. End user mashup programming environments. Tech. rep., Helsinki University of Technology, Telecommunications Software and Multimedia Laboratory, 2008.

[6] Cappiello, C., Daniel, F., Matera, M., Picozzi, M., and Weiss, M. Enabling End User Development through Mashups: Requirements, Abstractions and Innovation Toolkits. In *Proceedings of the Third International Symposium on End-User Development* (2011), pp. 9–24.

[7] Engels, G., Hausmann, J. H., Heckel, R., and Sauer, S. Dynamic meta modeling: a graphical approach to the operational semantics of behavioral diagrams in UML. In *Proceedings of the 3rd International Conference on the Unified Modeling Language: advancing the standard* (Berlin, Heidelberg, 2000), UML'00, Springer-Verlag, pp. 323–337.

[8] Ennals, R. J., and Garofalakis, M. N. Mashmaker: mashups for the masses. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2007), SIGMOD'07, ACM, pp. 1116–1118.

[19] `http://linkedgeodata.org`
[20] `http://www.openstreetmap.org/`

[9] Gaio, M., Sallaberry, C., Etcheverry, P., Marquesuzaà, C., and Lesbegueries, J. A global process to access documents' contents from a geographical point of view. *Journal of Visual Languages and Computing 19* (February 2008), 3–23.

[10] Haklay, M., Singleton, A., and Parker, C. Web Mapping 2.0: The Neogeography of the GeoWeb. *Geography Compass 2*, 6 (2008), 2011–2039.

[11] Heath, T., and Bizer, C. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool, 2011.

[12] Hennicker, R., and Koch, N. Modeling the user interface of web applications with UML. In *Practical UML-Based Rigorous Development Methods - Countering or Integrating the eXtremists, Workshop of the pUML-Group held together with the UML 2001* (2001), pp. 158–173.

[13] Huynh, D. F., Karger, D. R., and Miller, R. C. Exhibit: lightweight structured data publishing. In *Proceedings of the 16th International Conference on World Wide Web* (New York, NY, USA, 2007), WWW'07, ACM, pp. 737–746.

[14] Lin, J., Wong, J., Nichols, J., Cypher, A., and Lau, T. A. End-user programming of mashups with Vegemite. In *Proceedings of the 14th International Conference on Intelligent User Interfaces* (New York, NY, USA, 2009), IUI'09, ACM, pp. 97–106.

[15] Luong, T. N., Etcheverry, P., Nodenot, T., and Marquesuzaà, C. WIND: an Interaction Lightweight Programming Model for Geographical Web Applications. In *Proceedings of the International Opensource Geospatial Research Symposium* (2009), OGRS'09. Available online.

[16] Manola, F., and Miller, E. RDF Primer. Recommendation, W3C, February 2004. `http://www.w3.org/TR/rdf-syntax/`.

[17] Prud'hommeaux, E., and Seaborne, A. SPARQL Query Language for RDF. Recommendation, W3C, January 2008. `http://www.w3.org/TR/rdf-sparql-query/`.

[18] Ro, A., Xia, L. S.-Y., Paik, H.-Y., and Chon, C. H. Bill Organiser Portal: A Case Study on End-User Composition. In *Proceedings of the 2008 International Workshops on Web Information Systems Engineering* (Berlin, Heidelberg, 2008), WISE'08, Springer-Verlag, pp. 152–161.

[19] Sallaberry, C., Royer, A., Loustau, P., Gaio, M., and Joliveau, T. GeoStream: Spatial Information Indexing Within Textual Documents Supported by a Dynamically Parameterized Web Service. In *Proceedings of the International Opensource Geospatial Research Symposium* (2009), OGRS'09. Available online.

[20] Stühmer, R., Anicic, D., Sen, S., Ma, J., Schmidt, K.-U., and Stojanovic, N. Lifting events in RDF from interactions with annotated web pages. In *Proceedings of the 8th International Semantic Web Conference* (Berlin, Heidelberg, 2009), ISWC'09, Springer-Verlag, pp. 893–908.

[21] Taivalsaari, A. Mashware: The future of web applications. Tech. rep., Sun Microsystems Laboratories, 2009.

[22] Viana, W., Filho, J. B., Gensel, J., Villanova-Oliver, M., and Martin, H. PhotoMap: From Location and Time to Context-Aware Photo Annotations. *Journal of Location Based Services 2* (September 2008), 211–235.

[23] Wong, J., and Hong, J. I. Making mashups with Marmite: towards end-user programming for the Web. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (New York, NY, USA, 2007), CHI'07, ACM, pp. 1435–1444.