The Experience of Realizing a Semantic Web Urban Computing Application

Emanuele Della Valle[1,2], Irene Celino[1], and Daniele Dell'Aglio[1]

[1] CEFRIEL – ICT Institute, Politecnico of Milano,
Via Fucini 2, 20133 Milano, Italy
name.surname@cefriel.it
[2] Dipartimento di Elettronica e Informazione, Politecnico di Milano,
Piazza Leonardo da Vinci 32, 20133 Milano, Italy
emanuele.dellavalle@polimi.it

Abstract. Urban Computing is a branch of Pervasive Computing that investigates urban settings and everyday lifestyles. A lot of information to develop pervasive applications for urban environments is often already available, even if scattered and not integrated: maps, points of interest, user locations, traffic, pollution, events are just a few examples of the digitalized information which we can access on the Web. And applications for mobile users that leverage such information are rapidly growing.
In this paper, we report our experience in applying techniques developed by the Semantic Web and geo-spatial communities to Urban Computing application development. We refer to the early achievements of the LarKC project, in which we developed the described demonstrator. We highlight the positive sides of our experience and we discuss open issues and possible advances.

Keywords: Urban Computing, SPARQL, Geo-data, Events, Linked Data

1 Introduction and motivation

Urban Computing [1] was defined in 2007 by IEEE Pervasive Computing as "The integration of computing, sensing, and actuation technologies into everyday urban settings and lifestyles."

Pervasive Computing [2] has been mostly studied either in relatively homogeneous rural areas – such as farms, glaciers and coral reefs – or, in small scale built environments – such as smart rooms for elderly or people with disabilities. In both such cases, researchers add their own sensors and actuators to the environment and the focus of their research includes sensing technologies, in-loco energy production, low power consumption and ad-hoc networks to allow data collections.

Urban settings are different. They include streets, squares, metro stations, buses, taxis, monuments, museums, shops, pubs, cafés – just to cite a few examples of the semi-public spaces that people daily use. Deploying your own sensors and actuators in urban settings is often hard; however, in many cases, useful information to develop urban computing applications is already available. Maps of the cities, collections of points of interest (e.g., schools, hospitals, hotels, landmarks, monuments, public transportation stops), voluntarily-provided user

locations, geo-tagged user generated contents, traffic information (e.g., congestions, accidents, public transportation problems), pollution conditions, etc.: this is the kind of digitalized information that today is largely made available over the Web[1].

Moreover, we perceive an increasing demand for implementations of *Mobile Applications* that use such urban-centric information. Apple Store has an entire section[2] dedicated to iPhone applications that help users in searching the urban space around them. Among others, Citysense[3] is a good and representative example. It shows San Francisco nightlife activity in real time; it allows to see top nightlife hot-spots, checking if the level of activity is unusual, and it permits to visualize on the mobile phone what's going on.

We believe that both Urban Computing and Mobile Application development can largely benefit from the techniques developed by the Semantic Web and geospatial communities because:

1. only a part of urban-related data is natively managed by geographic information systems;
2. the rest of such data has some sort of geographic reference such as a street address, a toponymy or simple geographic coordinates; and
3. a central problem of Urban Computing is information integration for which Semantic Web technologies already proved to be a valid solution.

In order to experimentally verify the applicability of such techniques to Urban Computing, we are running a use case of the European research project Large Knowledge Collider (LarKC, for short, pronounced "lark") [3] dedicated to Urban Computing. LarKC aims at removing the scalability barriers of currently existing reasoning systems for the Semantic Web and implementing a platform for massive distributed incomplete reasoning.

In this paper, we describe an experience in realizing this kind of urban computing applications using Semantic Web technologies as a backbone for geospatial information integration. In Section 2, we present the user need we aim to satisfy and we motivate why using Semantic Web and geospatial technologies. In Section 3, we describe the Web sources from which we take urban-related information. Section 4 is dedicated to describing our approach and the resulting application. Finally, we close the paper with Section 5 discussing open issues and possible advancements.

2 A Urban Computing Scenario

A sample scenario for a Urban Computing application is the following. A user is in a (potentially unknown) city and would like to organize a day/night by visiting some places, meeting his friends, attending a music concert, etc. Therefore, he would like to *plan his movement to his destinations*, by possibly using a combination of transportation means (e.g. car, parking, subway, pedestrian).

[1] As an example of urban data availability on the Web, see the UK government initiative Show Us a Better Way – `http://www.showusabetterway.co.uk/`

[2] `http://www.apple.com/iphone/apps-for-everything/going-out.html`

[3] `http://www.citysense.com`

If the user does not know in advance the destination of his route, he could *express some requests or ambitions* and the destination is selected on the basis of those preferences. For example, the user says that he would like to go and visit some interesting monuments or venues of the city, or that he would like to attend some music concert or cultural event that night, or that he would like to meet some of his friends that happen to be in the same city. Therefore, *his destinations can be known places or some dynamically-chosen locations*, like a monument – selected between the relevant ones of a city, which are the closest to the user's current position –, an event – among those published on the Web and taking place at a specific date-time – or a friend – whose position can change over time.

In order to fulfill the user request, *numerous distributed and heterogeneous data sources* should be accessed and several different parameters should be taken into account to calculate the "most desirable" path for the user. The impact of the dynamic destination selection on the application business logic is that, first of all, a query should be routed to an appropriate data source (an archive of points of interests, a source of events schedules, a localization systems for a social network) and should select some possible destinations; then, for each destination, a suitable strategy to find the most desirable path[4] should be adopted.

To address this scenario today, the user would have to know in advance:

- *what to search for*, but if he is new in the city, he can be unaware of what is interesting to visit;
- *where to search*, but general-purpose search engines could be the wrong place where to search, because they provide a lot of information, possibly hiding the data interesting for the user; moreover, specialized information sources with more tailored content, can be unknown to the user; and
- *if the found information is correct*, consistent and updated, but if he finds two contrasting pieces of information he cannot tell the right one.

All in all, he would have to use multiple services, to check his requirements by hand, to manually pass intermediate results from a service to another one, etc.

As a consequence, the achievement of this scenario would become a *very long and expensive activity*. A better solution, however, can come from the employment of Semantic Web and related technologies, which can help in understanding what to search for (e.g. by applying query-expansion techniques), where to search (e.g. by querying the Semantic Web or Web of Data) and in verifying the correctness of the information (e.g. by filtering data and double-checking data or provenance). Section 4 illustrates how we implemented this Urban Computing scenario in an application built on the LarKC platform.

3 Gathering Data from the Web

As mentioned in the introduction, the Web today is becoming more and more the primary source of information for a large variety of topics and subjects. Urban

[4] The *most desirable path* depends on the user request; it can be the quickest run by car, the shortest distance on foot, the less polluted path by bicycle, etc.

environments as well are represented on the Web with a lot of different and distributed pieces of information: maps, events, interesting places, traffic data, etc.[5] Moreover, local governments' awareness of the need for publishing data on the Web for public usefulness is increasing [4].

Concentrating our attention onto the city of Milano in Italy, we identified several data sources, including some geo-spatial ones, to take into account for our Urban Computing scenario. Those data sources are diverse and heterogeneous not only in content, but also in format and availability conditions. For those reasons, we not only gathered data, but we also elaborated, processed and, in some cases, converted them into a more suitable format. Since one of our objectives is to integrate information coming from different sources, we adopted RDF [5] as interchange format and, since we are employing Semantic Web technologies, we tried to link data to existing and shared ontologies whenever possible. In the following we provide an insight into the data we gathered and processed and the problems and issues we encountered.

3.1 City topology data

Milano is one of the largest Italian cities and the Municipality of Milano established an Agency[6] able to give a support for the tasks of planning and programming mobility management and environmental control. This Agency releases data about the city topology and its traffic that can be freely used for non-commercial purposes by registered users who are requested to acknowledge the source. The format of the downloaded files is the ESRI shapefile, compatible with some GIS systems[7]. Since we decided to convert those data into RDF format, the process we adopted can be summarized as follows.

Data Preparation via GIS: we loaded the data into a GIS application, namely PostGIS[8]. The gathered data are about the road network of Milano and some municipalities around it (the hinterland) and contain the directed graph of the road network, with information about: *links* (street portions) and *nodes* (streets intersections), *road typology* (main roads, secondary roads, etc.), *jurisdictions*, turning prohibitions, etc. Loading the data into the GIS application allowed us to convert the original geographic coordinates – expressed in the Gauss-Boaga system[9] – into WGS-84 coordinates, which can be more frequently found in other data sources. Moreover, PostGIS stores its data in a PostgreSQL[10] database, which can therefore be accessed directly.

Ontology Modeling: on the basis of the analysis, we derived some ontological schemata to represent the data. Whenever possible, we used or linked to exist-

[5] We are running a survey about publicly available data sources; please, contribute at `http://wiki.larkc.eu/UrbanComputing/PublicAvailableDataSources`.

[6] Agenzia Mobilità, Ambiente e Territorio (AMAT) `http://www.amat-mi.it/`

[7] More information available here: `http://webhelp.esri.com/arcgisdesktop/9.3/index.cfm?topicname=Shapefile_file_extensions`

[8] PostGIS `http://postgis.refractions.net/`.

[9] The Gauss-Boaga Projection is the standard projection used in Italian topography by the Istituto Geografico Militare `http://www.igmi.org/`.

[10] PostgreSQL `http://www.postgresql.org/`.

ing and wide-spread ontologies, first of all W3C Geo Positioning RDF vocabulary[11]. Details are available at `http://wiki.larkc.eu/LarkcProject/WP6/WorkInProgress/AMAData`.

SPARQL Endpoint: in order to access the selected data sources as SPARQL [6] endpoint, we used a mapping tool – namely D2R [7] – and configured it to expose on the Web the data in the relational database described above, and to have the possibility to query them via SPARQL using the aforementioned ontologies; the D2R server is available on line at `http://seip.cefriel.it/ama/`[12]. Moreover, the mapping tool offers some facilities to get RDF dumps out of the wrapped data source. In this way, we obtained different RDF dumps (one with the whole graph of Milano roads and a bunch of others containing meaningful subsets of the whole graph) whose usefulness will be explained in the following.

AllegroGraph Endpoint: since the Urban Computing scenario involves geographical data, we looked for tools that are able to deal and to easily query RDF data that include location information. AllegroGraph[13] is an RDF store which offers some facilities to deal with geographic information. We used AllegroGraph to have a SPARQL endpoint to query the RDF data and to get the possibility to geo-spatially index the data. Therefore, the data can be also queried with a custom SPARQL extension which allows for selecting RDF triples on the basis of the location information.

3.2 City points of interest and events information

Like several major cities around the world, Milano is a very lively environment, full of cultural and leisure attractions. For our Urban Computing scenario, we need to access several different sources of information, which describe point of interests (monuments, tourist attractions, relevant places, etc.) and events (exhibitions, music concerts, sport matches, meetups, etc.).

Ontology Modeling: also in this case, trying to minimize the addition of custom concepts/properties, we re-used whenever possible existing and popular ontologies. Among them, we employed the schemata coming from SKOS [8], DBpedia, RDF Calendar, tags and address vocabularies[14].

Monuments and Points of Interest: with this regards, due to the large availability of general-purpose data on the Semantic Web, we decided to directly query linked data sources like Sindice[15] and DBpedia[16]. While querying the latter gives immediately RDF data, searching on the former need the invocation of REST [9] services by passing triple-based patterns and getting back references to RDF sources.

[11] W3C Geo Positioning RDF vocabulary `http://www.w3.org/2003/01/geo/`.

[12] Web access is password-protected; please, ask the authors for proper credentials.

[13] AllegroGraph `http://agraph.franz.com/allegrograph/`; SPARQL geo-extension `http://franz.com/agraph/support/documentation/current/sparql-geo.html`.

[14] RDF Calendar from W3C: `http://www.w3.org/2002/12/cal/`; tagging ontology by Richard Newman: `http://www.holygoat.co.uk/owl/redwood/0.1/tags`; address schema by Talis `http://schemas.talis.com/2005/address/schema`.

[15] Sindice semantic web index `http://sindice.com/`.

[16] DBpedia `http://dbpedia.org/`.

Events Information: on the Web, it is more and more frequent to find aggregator Web sites, which collect information about happenings and their respective venues and details. One of the most famous and popular is Eventful[17]; it enables its community of users to discover, promote, share and create events. Moreover, it allows the community of developers to take advantage of its content by using their REST services, which return events information under an XML, JSON or YAML format. By invoking the XML REST services, and then by applying suitable XSL transformations as per the GRDDL [10] approach, we were able to obtain in RDF also the data about events in Milano. This approach (REST service invocation + GRDDL transformation) is a general process that can be easily applied to other data sources.

4 The alpha Urban LarKC Implementation

We implemented the scenario described in Section 2 by using the data sources described in Section 3 and by leveraging the first public release of the LarKC Platform[18]. The LarKC platform is both a SPARQL endpoint exposed to client applications and a pluggable Semantic Web framework that allows developers to compose *pipelines* [11] made up of plug-ins which belong to five different types:

1. *Deciders* that receive the client SPARQL query, decide the set of plug-ins to assemble in a pipeline in order to answer the client query, iterate the pipeline until a *good enough answer* [12] is found, and sends the result back to the client.
2. *Identifiers* that identify relevant data sources and fetch contents potentially useful to answer the client query.
3. *Transformers* that help in overcoming format heterogeneity among independently developed plug-ins; a Transformer can turn the SPARQL client queries into the data source-specific query format, or it can transform the fetched contents from the data source-specific format into RDF, or even transform from RDF into some plug-in specific data representation.
4. *Selectors* that select a subset of the fetched information by applying filtering policies.
5. *Reasoners* that perform the required reasoning tasks.

As depicted in the upper right corner of Figure 1, we configured the LarKC platform using a common Decider that serves as a gateway to three pipelines: two of them select destinations in Milano (either monuments or events) while the third one finds the most desirable paths to such destinations. The client application can be tried at `http://seip.cefriel.it/alpha-Urban-LarKC/`.

When the LarKC platform is started, the *Decider* assembles the three pipelines in a static way and then waits for SPARQL queries of the kinds shown in the following listings. Being aware of those three possible kinds of SPARQL queries, when a client issues a query, the Decider can route such query to the correct pipeline to process it. After selecting the pipeline, like any LarKC decider, it controls the execution of the queries through the different plug-ins in the

[17] Eventful `http://eventful.com/`; Eventful API `http://api.eventful.com/`.
[18] See `http://wiki.larkc.eu/LarkcProject/WP5/GForgeSVN` for details.

Fig. 1. The alpha Urban LarKC application.

chosen pipeline, collects the results from the Reasoner and returns them back to the client.

```
SELECT DISTINCT ?monument ?geopoint ?img ?name ?desc {
  {{?monument skos:subject ?subject.
     ?subject skos:broader dbpedia:Visitor_attractions_in_Milan .}
  UNION
    {?monument skos:subject dbpedia:Visitor_attractions_in_Milan .}}
  ?monument georss:point ?geopoint .  ?monument foaf:depiction ?img .
  ?monument rdfs:label ?name .        ?monument rdfs:comment ?desc .
  FILTER ( lang(?name) = "en" && lang(?desc) = "en" )                }
```

Listing 1. An example of the SPARQL query that selects attractions in Milano.

The pipeline that selects monuments in Milano is able to answer SPARQL queries as in Listing 1. Notably, the query in Listing 1 does not name any specific visitor attraction of Milano, but it asks for DBpedia resources that are categorized as "visitor attractions in Milano" or any of its direct sub-categories. It does so through a *Transformer* that analyzes the query to get the triple patterns, an *Identifier* that takes such triple patterns and queries the Semantic Web search engine Sindice to fetch relevant RDF documents, a *Selector* that filters the fetched RDF documents to extract information about relevant monuments and a *Reasoner* that answers the query.

The geo-extended AllegroGraph endpoint and the Reasoner are required to deduce, starting from the geo-position of each monument (i.e., the value of the `georss:point` property), the closest node of Milano street topology (see Section 3.1) where each monument is placed. This node will be used as destination node in the query to find the most desirable path (see property *map:pathTo* in Listing 3).

To select an event in Milano, we set up a pipeline able to answer queries of the kind shown in Listing 2.

```
SELECT ?e ?s ?summary ?l ?label ?lat ?long
WHERE{
  ?e rdf:type rdfcal:Vevent .   ?e rdfcal:summary ?summary .
  ?e geo:location ?l .          ?l rdfs:label ?label .
  ?l geo:lat ?lat .             ?l geo:long ?long  .
  ?e rdfcal:dtstart ?s .        ?l addr:countryName "Milano".
  FILTER(?s > xsd:dateTime("2009-07-15T00:00:00Z")
      && ?s < xsd:dateTime("2009-07-15T23:59:59Z")).       }
```

Listing 2. An example of the SPARQL query that selects events in Milano.

As in the previous pipeline, a *Transformer* intercepts the SPARQL query and extracts the parameters to invoke the REST service exposed by Eventful. An *Identifier* queries Eventful to get a list of events and passes the references to a *Transformer*, which uses GRDDL to translate the XML results of the REST invocations into a set of RDF graphs, each representing an event. The rest of the pipeline is composed by the same Selector and Reasoner described above. As in the previous pipeline, the geo-extended AllegroGraph endpoint and the Reasoner deduce the node of Milano street topology (see Section 3.1) which is closer to each event venue.

The last pipeline in our Urban Computing application is the one in which Semantic Web and geo-spatial technologies are more strictly tied. This pipeline finds the most desirable path from the user current position to one of the destinations selected by the two previous pipelines (see SPARQL query in Listing 3).

```
SELECT ?p ?w ?n1 ?l ?n2
WHERE {
  ?p rdf:type     map:Paths    ;  map:pathWeight ?w
     map:pathFrom map:node2509 ;  map:pathTo  map:node16198 ;
     map:contain  ?l .
  ?l rdf:type map:Link ;
     map:from ?n1      ;  map:to ?n2 .                      }
```

Listing 3. An example of the SPARQL query that finds the most desirable path.

The core of this pipeline is a *Reasoner* plug-in that wraps a computational service able to find the shortest path in a graph. The pipeline adopts three different strategies to identify and select a subset of the Milano street topology data to reason on; this lets the Reasoner process only the minimum useful subset of information. If the start and the end point are close-by (i.e., their distance is less then 2 km) but outside Milano center, it fetches all the streets in a circular area containing the two points. If the start and the end points are within Milano center jurisdiction, it loads the corresponding RDF graph that includes all streets in the jurisdiction[19]. In all other cases, it fetches the graph containing only the streets in two small circular areas around the start and end points and all the main roads of Milano. In the last two cases, the geo-extended AllegroGraph endpoint is used to fetch the streets in a given circular area.

[19] This jurisdiction is the historic center where roads are mostly one-way, therefore it is sensible to select the whole jurisdiction instead of a smaller circular area.

5 Open Issues and Possible Advancements

As developers of applications based on Semantic Web and geo-spatial technologies, we can report that our experience in building a Urban Computing application was a positive experiment even if it left us with many open issues.

On the positive side, we can report that implementing Semantic Web applications based on LarKC makes the development more modular and, thus, easier to plan, execute, parallelize and control. Since LarKC is a pluggable framework, we were able to easily integrate geo-spatial tools and make them work together with the rest of our Semantic Web plug-ins. Modularity and seamless integration do not only apply to geo-spatial plug-ins, but they represent a generic characteristic of the LarKC platform, which allows for extensibility of applications both in terms of data and software components. Last but not least, respecting the Web principle of leaving the information management to those that publish it, we were able to develop a system that is always up-to-date with no data locally replicated (but for caching purposes).

However, if we compare what we were able to realize with the requirements and challenges for the Semantic Web we defined in [13], we can easily see that most of our requirements remain unaddressed. In our experience, we successfully managed to use precise and consistent reasoning techniques to resolve geo-spatial ambiguities, but we believe that, in the general case, Urban Computing needs also forms of approximate reasoning, e.g. forecasting the availability of free parking lots close to an event venue. We are running the entire application under the "close world assumption" and the "unique name assumption", whereas most of the data we are working with are incomplete and inconsistent by their own nature; therefore, we need reasoning systems that work under the "open world assumption" and are able to handle data quality issues.

In particular, we would like our system to be able to solve two problems: eliminate duplicated events found on aggregator Web sites like Eventful and to perform on-the-fly reconciliation of contradictory geo-coordinates. The former case is caused by the fact that Eventful is a open collaborative platform where everybody is free to publish an event; in case of important events, e.g. a rock star concert, more then one user will publish a copy of the same event with slightly different descriptions. The latter case happens when the geo-coordinates of an event venue or a visitor attraction are imprecise or incorrect. For instance, a venue may report contradictory postal address and geo-coordinates (e.g., the address is correct, but the geo-coordinates point to a default location in the middle of the city), or may appear in different Web sites with different geo-coordinates. To address both cases, we are currently working on a rule-based approach to detect duplicates and contradictions, in order to construct a unified representation of the location resources.

Finally, we would like to report two other possible advancements of our work. On the one hand, we intend to extend the current demonstrator to a world-wide scale by integrating the street topology of OpenStreetMap[20] made available

[20] OpenStreetMap `http://www.openstreetmap.org/`.

by the LinkedGeoData project[21]. On the other hand, we intend to generalize the approach we manually followed to virtually publish in RDF the shape files containing street topology; the result will be a mapping tool that natively wraps PostGIS, as tools like D2R [7] do with relational databases.

Acknowledgments

This research has been partially supported by the LarKC EU co-funded project (ICT-FP7-215535).

References

1. Kindberg, T., Chalmers, M., Paulos, E.: Guest editors' introduction: Urban computing. IEEE Pervasive Computing **6**(3) (2007) 18–20
2. Hansmann, U., Merk, L., Nicklous, M.S., Stober, T.: Pervasive Computing : The Mobile World (Springer Professional Computing). Springer (2003)
3. Fensel, D., van Harmelen, F., Andersson, B., Brennan, P., Cunningham, H., Della Valle, E., Fischer, F., Huang, Z., Kiryakov, A., il Lee, T.K., School, L., Tresp, V., Wesner, S., Witbrock, M., Zhong, N.: Towards LarKC: a Platform for Web-scale Reasoning, IEEE International Conference on Semantic Computing (ICSC 2008) (2008)
4. Berners-Lee, T.: Putting Government Data online - W3C Design Issue. Available on the Web at `http://www.w3.org/DesignIssues/GovData.html` (June 2009)
5. Manola, F., Miller, E.: RDF Primer - W3C Recommendation. Available on the Web at `http://www.w3.org/TR/rdf-primer/` (10 February 2004)
6. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF - W3C Recommendation. Available on the Web at `http://www.w3.org/TR/rdf-sparql-query/` (15 January 2008)
7. Bizer, C., Cyganiak, R.: D2R-Server - Publishing Relational Databases on the Web as SPARQL-Endpoints. In: Proceedings of the 15th International World Wide Web Conference (WWW2006). (2006)
8. Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System Reference - W3C Proposed Recommendation. Available on the Web at `http://www.w3.org/TR/skos-reference` (15 June 2009)
9. Fielding, R.: Representational state transfer (REST). Ph. D. Thesis, University of California, Irvine, CA (2000)
10. Connolly, D., *et al.*: Gleaning Resource Descriptions from Dialects of Languages (GRDDL) - W3C Recommendation. Available on the Web at `http://www.w3.org/TR/grddl/` (11 September 2007)
11. Celino, I., Dell'Aglio, D., Della Valle, E., Kim, K., Park, S., Steinke, F., Grothmann, R., Hauptmann, W.: Deliverable D6.5 Urban Computing environment v1. Technical report, LarKC Project Consortium (September 2009)
12. Shvaiko, P., Giunchiglia, F., Bundy, A., Besana, P., Sierra, C., Van Harmelen, F., Zaihrayeu, I.: Benchmarking methodology for good enough answers. Technical report, DISI-08-003, Informatica e Telecomunicazioni, University of Trento (2008)
13. Della Valle, E., Celino, I., Dell'Aglio, D., Kim, K., Huang, Z., Tresp, V., Hauptmann, W., Huang, Y., Grothmann, R.: Urban Computing: a challenging problem for Semantic Technologies. In: Workshop on New forms of Reasoning for the Semantic Web: scalable, tolerant and dynamic (NEFORS 2008), colocated with the 3rd Asian Semantic Web Conference (ASWC 2008), Bangkok, Thailand. (2008)

[21] LinkedGeoData `http://linkedgeodata.org/`.