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 geo-spatial communities because:

1. only a part of urban-related data is natively managed by geographic information systems (e.g., most of the information about points of interest and events is published on paper or in Web pages);
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 [3].

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") [4] dedicated to Urban Computing.

In this paper, we describe an experience in realizing this kind of urban computing applications using Semantic Web technologies as a backbone for geo-spatial information integration. In Section 2, we present the user need we aim to satisfy and we motivate why using Semantic Web and geo-spatial technologies. In Section 3, we provide some background on Semantic Web technologies and we describe the LarKC approach in removing the scalability barrier currently existing for reasoning at Web scale. In Section 4, we detail what the problems to overcome are and what principles are applied. In Section 5, we describe the Web sources from which we take urban-related information. Section 6 is dedicated to describing our approach and the resulting application. In Section 7, we describe the evaluation of our application and we discuss quantitative and qualitative

[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`

aspect of our results .Finally, we close the paper with Section 8 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, attending a music concert, etc. Therefore, he would like to *plan his movement to his destinations*.

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. 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 –, or an event – among those published on the Web and taking place at a specific date-time.

In order to fulfill the user request, *numerous distributed and heterogeneous data sources* should be accessed. 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. Secondly, 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 6 illustrates how we implemented this Urban Computing scenario in an application built on the LarKC platform.

[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.

3 Background

The scenario we presented above is about bringing context-sensitive and personalized services to mobile users, thus is not difficult to image a wide deployment of it by an telecommunication company. We did a rough estimation of the number of RDF triples to reason about in case of wide deployment and we got the need to reason about 10 billion RDF triples in around 100 ms.

According to a recent technical report [5] on the state of the art of storage, query and inference technology, the average query time of the best OWL-DL [6] compliant reasoner such as SwiftOWLIM 2.9.1 [7] on OUMB benchmark [8] is already 49 ms. when reasoning on 0.2 million RDF triples, but explode to 7,9 seconds when reasoning about 4.4 million RDF triples. Even giving up the OWL-DL expressivity and choosing RDFS [9] do not result in much better results. Reasoning about of the LUMB benchmark [10] for 1 billion RDF triples with BigOWLIM 0.96 [7] requires a minimum of 75 ms. However we are still reasoning on a tenth of the triples we estimated.

The vision of the LarKC project [4] is to overcome current limitation for semantic computing represented by available storage, querying and inference technology. The fundamental assumption taken is that the current paradigms, which are strictly based on logic, are too limiting. By fusing reasoning (in the sense of logic) with search (in the sense of information retrieval) [11], LarKC is aiming at the paradigm shift that is required for reasoning at Web scale.

The LarKC project is building an integrated platform for semantic computing on a Web scale. The platform is designed to fulfill needs in sectors that are dependent on massive heterogeneous information sources such as Urban Computing. The platform has a pluggable architecture in which it is possible to exploit techniques and heuristics from diverse areas such as databases, machine learning, cognitive science, Semantic Web, and Geographic Information Systems. As internal data representation it uses RDF [12] and users are expected to pose query using SPARQL [13].

A LarKC application consist of a number of pluggable components arrange in a workflow executed by the LarKC platform (see [14] for a detailed description). LarKC plug-ins cover a variety of tasks[5]:

- *identification* of sources of information potentially useful to answer the query issued by the client;
- *fetching* information from the identified sources;
- *selection* of relevant subset of the fetched information ;
- *translation* of information from the source format in RDF or of queries from SPARQL to source specific query language;
- *reasoning* on the collected information in order to provide answers to the query issued by the client.

A fifth special kind of plug-ins exist: the *deciders*. A decider is used to compose a workflow and monitor its execution. Results of SPARQL query to LarKC

[5] The up-to-date list of available plug-ins is published on the Web at `http://wiki.larkc.eu/LarkcPlugins`.

can be delivered either in one solution or in an incremental way. In this second case, the decider is responsible to iterate through the workflow. In subsequent iteration, the number of identified sources and the dimension of the selected subset of the information in each source are increased. The decider stops the iterative process when a *good enough answer* [15] is found.

4 Problems and Principles

In this section we describe in more detail what the problems to overcome are and what principles we apply in choosing to combine Semantic Web and geo-spatial technologies. We number the problems and the principles to be able to refer to them in the following sections.

Problem 1. Our Urban Computing scenario requires information stored in several data sources. Those data sources are diverse and heterogeneous not only in content, but also in format and availability conditions.

Principle 1. Since one of our objectives is to integrate information coming from different sources, we adopted RDF [12] as interchange format and, since we are employing Semantic Web technologies, we tried to link data to existing, shared and popular ontologies whenever possible. In the following section we provide an insight into the data we gathered and processed and the problems and issues we encountered.

Problem 2. In many cases duplication of information is not allowed or, as in the case of events, would rapidly result in out-of-date copies, because of the update rate of the sources.

Principle 2. Adopt as much as possible solutions that generate virtual RDF graphs on demand. In the case of relational database this can be done using solution such as D2R [16]. In the case of information encoded in standard XML formats, use GRDDL [17] together with standard XSLT[6] that transform on the fly from XML to RDF using popular ontologies.

Problem 3. Since information is continuously updated a mechanism to discover new information is needed.

Principle 3. Information sources are never access directly, but contents are found by using a search engine. In the case of data already published in RDF we can use Semantic Web search engine like Sindice[7]. In the case of data, such the events, that are not available in RDF, vertical search engines such as Eventful[8] can be used. This allows to hardwire [18] the kind of information our Urban Computing application will elaborate, but not the specific instances that can be discovered.

Problem 4. Information alone is not *actionable* and needs to be interlinked. For instance, knowing the geographic coordinates of a monument is not enough for a GPS navigator compute a path, the monument position has to be linked to

[6] For more information on standard XSLT to be used together with GRDDL we refer to `http://esw.w3.org/topic/CustomRdfDialects`.

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

[8] Eventful `http://eventful.com/`

a node in the road network model that it is used by the path finding algorithm implemented in the GPS navigator.

Principle 4. Data has limited value if not linked, therefore we use as much as possible automatic data linking techniques. In our specific case geo-spatial functionalities are exploited to link the geographic coordinates of a monument to the closest node in the road network model.

Problem 5. Urban Computing application developers should be able to issue their queries in an homogeneous way and without knowing the physical position of information sources.

Principle 5. We adopt SPARQL as query language to issue all the queries. This choice is compatible this the decision to transform (in most cases virtually) all information in RDF. Moreover, in order to hide the physical position of the information sources from the client, the client does not need to tell the SPARQL endpoint where the information sources are located using the FROM clause; the system will identify them and fetch the required information.

Problem 6. In deciding to use RDF and SPARQL, one may be tempt to replicate in a general purpose reasoner tasks that are better solved by dedicated algorithms; for instance, one may be tempt to implement in a rule base engine a shortest path algorithm. Such a "reinventing the wheel behavior" can be fine for fast prototyping, but can become a performance bottle neck on real settings.

Principle 6. We do not want to reinvent the wheel, but dedicated algorithm should be add as built-in to reasoners. The approach we chose is to extend the capabilities of a SPARQL processor using computed properties[9] (also known as magic properties or property functions or functional predicates), i.e. special RDF properties that trigger the execution of built-ins. Example of such properties are predicates for free text search present in ARQ, Virtuoso and AllegroGraph SPARQL implementations.

5 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 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.[10] Moreover, local governments' awareness of the need for publishing data on the Web for public usefulness is increasing [19].

5.1 City topology data

Milano is one of the largest Italian cities and the Municipality of Milano established an Agency[11] able to give a support for the tasks of planning and

[9] For more information on computed properties see `http://esw.w3.org/topic/SPARQL/Extensions/Computed_Properties`.

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

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

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[12]. Since we decided, in accordance with principles 1 and 2, to virtually 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[13]. 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[14] – into WGS-84 coordinates, which can be more frequently found in other data sources. Moreover, PostGIS stores its data in a PostgreSQL[15] 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 existing and wide-spread ontologies, first of all W3C Geo Positioning RDF vocabulary[16]. Details are available at `http://wiki.larkc.eu/LarkcProject/WP6/WorkInProgress/AMAData`.

SPARQL Endpoint: in order to access the selected data sources as virtual RDF using a SPARQL [13] endpoint, we used a mapping tool – namely D2R [16] – and configured it to expose on the Web the data in the PostgreSQL relational database described above. In this way, in accordance with principle 2, we gain the possibility to query them via SPARQL using the aforementioned ontologies without actually translating all data in RDF; the D2R server is available on line at `http://seip.cefriel.it/ama/`[17]. Moreover, the mapping tool offers some facilities to get RDF dumps out of the wrapped data source. Violating the principle 5 for reasons that we discuss in Section 8, we got two RDF dumps: one with the whole graph of Milano main roads and one with all the roads of Milan central jurisdiction.

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[18] is an RDF store which

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

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

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

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

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

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

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

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.

5.2 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 [20], DBpedia, RDF Calendar, tags and address vocabularies[19].

Monuments and Points of Interest: with this regards, due to the large availability of general-purpose data on the Semantic Web, we decided, in accordance with principle 3, to indirectly access DBpedia[20] resources by searching for them using Sindice. While querying the formers gives immediately RDF data, searching on the latter need the invocation of REST [21] services by passing triple-based patterns and getting back references to RDF sources.

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; 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[21], which return events information under an XML, JSON or YAML format. By invoking, in accordance with principle 3, the XML REST services, and then by applying, in accordance with principle 2, suitable XSL transformations as for the GRDDL [17] approach, we were able to obtain in RDF also the data about events in Milano.

6 The alpha Urban LarKC Implementation

We implemented the scenario described in Section 2 by using the data sources described in Section 5 and by leveraging the first public release of the LarKC Platform[22].

[19] 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`.

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

[21] Eventful API `http://api.eventful.com/`.

[22] See `http://wiki.larkc.eu/LarkcProject/WP5/GForgeSVN` for details.

Fig. 1. The alpha Urban LarKC application.

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 workflows: 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 workflows in a static way and then, in accordance with principle 5, 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 workflow to process it. After selecting the workflow, like any LarKC decider, it controls the execution of the queries through the different plug-ins in the chosen workflow, 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 workflow that selects monuments in Milano is able to answer SPARQL queries as in Listing 1. Notably, in accordance with principle 5, the query in Listing 1 does not name any specific visitor attraction of Milano and it does not have a FROM clause, but it asks for DBpedia resources that are categorized as "visitor attractions in Milano" or any of its direct sub-categories.

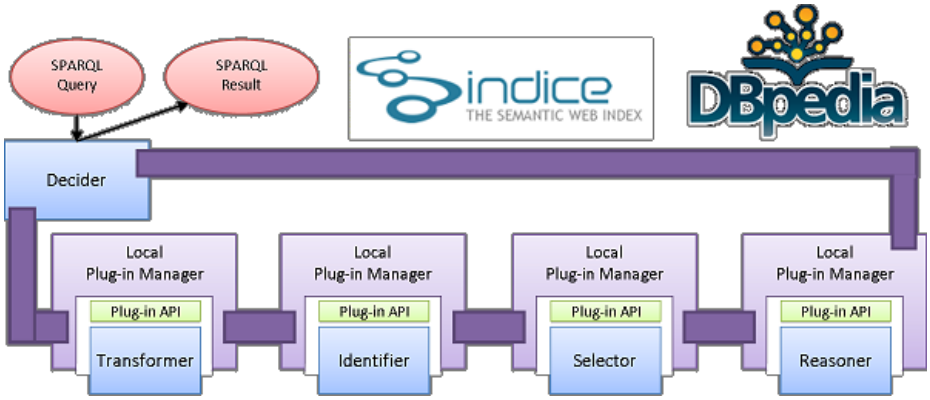

Fig. 2. The Monument selection workflow.

A diagram describing this workflow is shown in Figure 2. It is composed by 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.

In accordance with principle 4, using the geo-position of each monument (i.e., the value of the `georss:point` property) the geo-extended AllegroGraph endpoint and the Reasoner link the URI identifying each monument to the closest node of Milano street topology (see Section 5.1). In this way the monument information becomes *actionable*, meaning that the node linked to the monument can be used as destination 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 the workflow shown in Figure 3 that is 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 workflow, 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 workflow is composed by the same Selector and Reasoner described above.

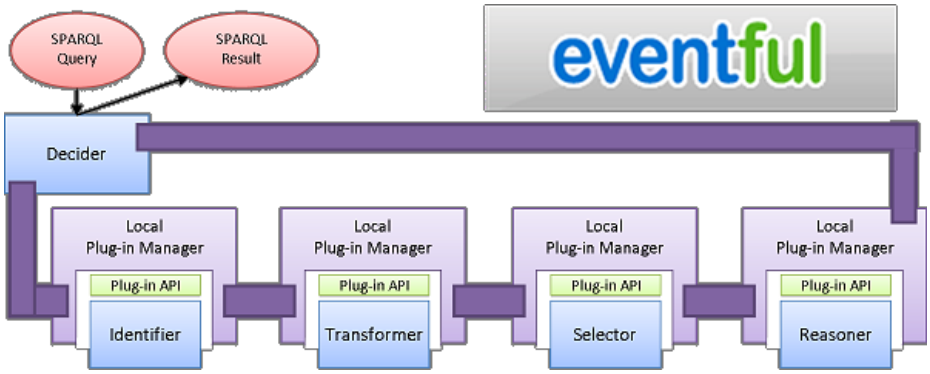

Fig. 3. The event selection workflow.

As in the previous workflow, the geo-extended AllegroGraph endpoint and the Reasoner deduce the node of Milano street topology (see Section 5.1) which is closer to each event venue.

The last workflow in our Urban Computing application is the one in which Semantic Web and geo-spatial technologies are more strictly tied. This workflow finds the most desirable path from the user current position to one of the destinations selected by the two previous workflows (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 workflow is a *Reasoner* plug-in that, in accordance with principle 6, wraps a computational service able to find the shortest path in a graph. We search the literature of route planning looking for approaches that fits LarKC strategy to The workflow adopts three different strategies to identify and select a subset of information (in this case a subset of the Milano street topology) to reason about (in this case compute the most desirable path). We learned that "algorithms for route planning in transportation networks have recently undergone a rapid development, leading to methods that are up to one million times faster than Dijkstra's algorithm"; those methods improve performances by a smarter selection of the data subset on which applying standard algorithms [22].

Taking inspiration from Hierarchical Approaches to route planning [23] we identified three strategies (see Figure 4):

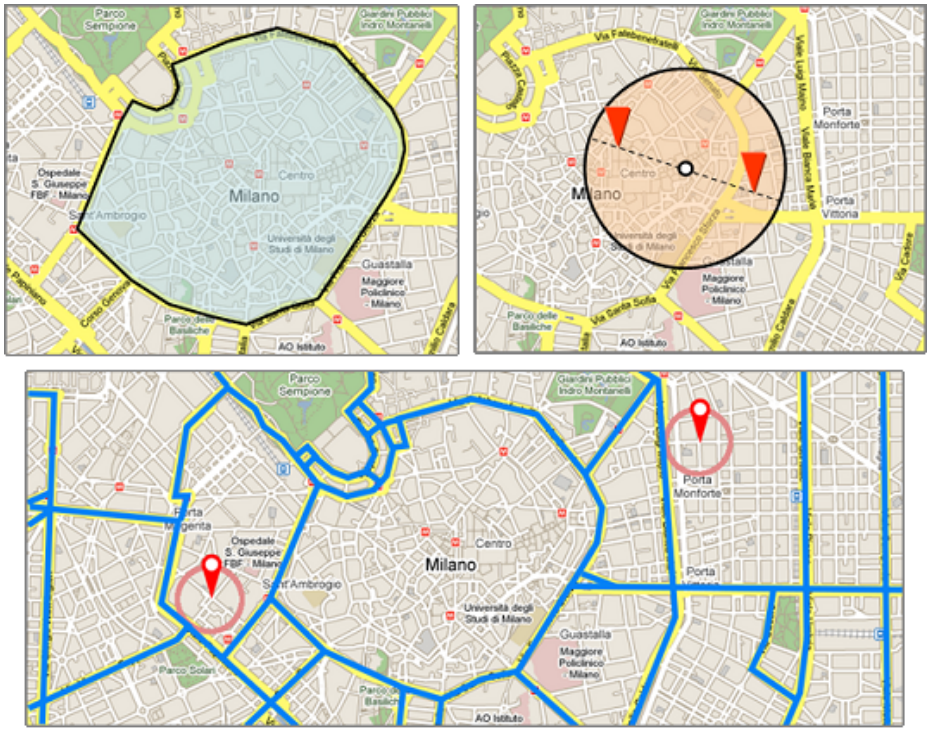

Fig. 4. Path-finding strategies.

1. 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[23].
2. If the start and the end point are close-by (i.e., their distance is less then 2 km) but outside Milano center, the path finding workflow fetches all the streets in a circular area containing the two points.
3. In all other cases, it fetches the graph containing only the streets in two small circular areas around the start and end points and the RDF graph containing 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.

7 Evaluation

In this section we report on the evaluation of the alpha Urban LarKC implementation. Our main motivations is to provide quantitative and qualitative evidence

[23] 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.

for the combined use of Semantic Web and geo-spatial technologies in the context of Urban Computing. This evaluation also offer more insight about how well they performed. We performed 25 tests[24] of three different kind; for a reason of space we only report the result of one or two tests per kind, for a comprehensive description we refer interest readers to [24]. The three kinds of tests are:

1. Performance tests of the three workflows and a comparison of such performances against a solution that do not use the LarKC platform. The purpose of this evaluation is to understand how much the flexibility and extensibility of the LarKC platform impacts on response time.
2. Stress tests of the three workflows. We measured how the response time depends on the number of multiple concurrent users. The purpose of this evaluation is to test the stability and resistance of alpha Urban LarKC workflows at increasing computation loads.
3. Tests of the quality of results computed by the monument and event workflows. Given the variety of the data and of the queries, we manually inspected the content sources (i.e., DBpedia and Eventful) and we compute recall and precision of the two queries discussed in the previous section.

To perform the first kind of tests, we collected the time necessary to find a path using the alpha Urban LarKC workflow and using a standalone implementation of the path finding algorithm (we name it "Term of Comparison", ToC) . The components used in the standalone application are the same deployed as plug-ins in the LarKC workflow. So, by comparing the two performance tests we can evaluate the overhead introduced by the LarKC platform. We distinguished three types of overhead: a) the time spent in warming-up the two implementations, b) the time to cool-down the two implementations and c) the time spent in coordinating the various components.

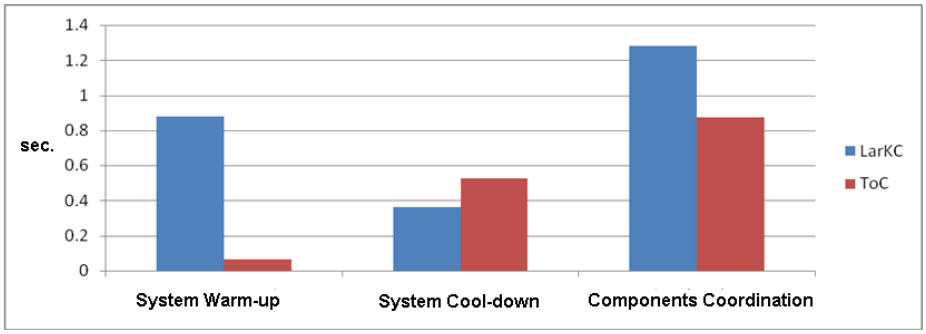

Fig. 5. Path-finding strategies.

[24] All tests were performed using an Intel Xeon 3.60/3.60 GHz (two dual core CPUs) with 4GB of RAM.

In Figure 5, we show the results of such a comparison between the LarKC path finding workflow and the ToC. As the bar chart showes LarKC spends more time than the ToC in warming-up and in coordinating components, but it is a bit more faster than the ToC in cooling-down. What we can conclude is that by using LarKC we trade some performances for flexibility, however at query time this is not too evident.

The main objective of the second kind of tests was to understand how the alpha Urban LarKC behavior is influenced by the number of concurrent requests it have to process. We first performed tests on the path-finding workflow in a close environment (all the machines are located in a LAN), while for the other two workflows we operated in an open environment, given that they both interact with data sources located on the Web (i.e., Sindice, DBpedia and Eventful).

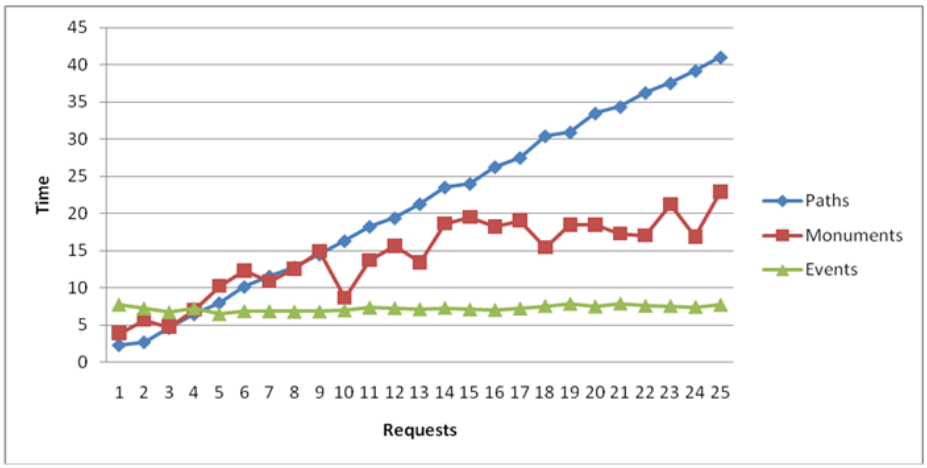

Fig. 6. Average response time of the three workflows in function of the number of concurrent users.

In Figure 6, we show the response time of the three workflows in function of the number of concurrent users. The workflow that performers better is the event one. It's performances appear stable in the number of concurrent users; to have a sensible increase in the response time we had to use a hundred of concurrent users. On the contrary, the other two workflow are very sensible to the number of concurrent user. We examined this in details and we discovered that the difference between the three workflows is the presence of some level of caching. Eventful has some caching facilities, thus asking multiple times the same requests results in a almost invariant response time. DBpedia has also some caching facilities, but they are not as effective as the Eventful once. The LarKC platform has no caching facilities and this is clearly visible in the path finding workflow.

We got to two conclusions. The first is obvious: caches improve over all performances. The lack of caching in the current version of the LarKC platform is a major bottle neck. Future releases will solve this issue with a caching facilities in the LarKC platform data layer that can be controlled by the plug-ins. The second is also obvious: do not run all the workflows on demand. Especially the monument workflow could be run once every while, new monuments are not expected to appear every hour. However, this second conclusion contrasts with the principle 3 (i.e., access information source indirectly to get always up-to-date information). Trading freshness of information for performances, in any case, is appropriate and we will consider it for future version of the Urban LarKC.

Last, but not least, we evaluate quality of the results. We performed these tests only for the monument and the event workflow. For the monument workflow, we manually checked DBpedia in order to take note of the available monuments and we compare this set of monuments with the monuments returned by the alpha Urban LarKC. The table hereafter shows high precision but law recall we experienced.

Available Milan monuments	25
Milan Monuments with coordinates	12
Milan Monuments with W3 geo coordinates	6
Monuments indexed by Sindice with W3C geo coordinates	2

DBpedia has 25 resources describing Milan monuments, only 12 of them have some geographic coordinates and among these only 6 have the geographic coordinates coded in the W3C vocabulary we use in our query. So the maximum recall we can expect for the query we issue is represented by those 6 monuments. However, only two of those monuments are returned by the monument workflow, because Sindice indexed only those two. This is experimental evidence that our principle 3 is difficult to apply in practice. Third-party services are not necessarily trustworthy and, therefore, principle 3 could only hold if service level agreements are put in place.

For the event workflow, we consider the Eventful source for the events data. While data in DBpedia changes rarely, events are continuously added to Eventful; due to this fact, we choose to collect Eventful data for a week and manually inspect the results of the event pipeline. Results retrieved in the week between September 5th and September 11th, 2009 are presented in table hereafter.

Date	Number	Unique	Wrong coords	Wrong date	Not in Milano
9/5/2009	12	11	9	1	1
9/6/2009	12	11	9	1	1
9/7/2009	13	10	8	1	1
9/8/2009	14	12	11	1	1
9/9/2009	19	15	11	1	1
9/10/2009	14	11	7	1	0
9/11/2009	11	9	7	1	2

The table above shows that every day about 10-15 events related to Milano were published. The workflow work perfectly, all events were identified, fetched and transform correctly. However, we noticed several data quality problem of

Eventful: some of events were inserted more than once by different people; coordinates and dates can be wrong; and several cities around the world are named Milano, therefore some of the events retrieved by Eventful where not organized in the Italian city. In this case recall is high and precision is conditioned by the multiple geographic entities that share the "Milano" name.

A part from the previously stated consideration that third-party service are not necessarily trustworthy, we see an important area of improvement. Semantic technologies can be employed to detected duplicates [25] and for handling disambiguation of geographical identifiers [26]. We further discuss these topic in the next section.

8 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. This flexibility comes at the cost of a small increase in response time, but future improvement of the LarKC platform should mitigate this problem. Last but not least, LarKC fosters the respect of principle 2, it pushes for leaving the information management to those that publish it. For these reason, we were able to develop a system that is always up-to-date with no data locally replicated (but for caching purposes).

So far Semantic Web technologies prove to offer effective tool to solve the problems our Urban Computing scenario arise. However, we have to report that we experienced practical difficulties in treating geo-spatial information as virtual RDF graphs. As we described in Section 5, in accordance with principle 2, we tried to us tools like D2R [16] to expose the geo-spatial information without physically transforming all the information in RDF. We were forced by the lack of tools that directly wrap GIS systems, to convert information from GIS formats to plain relational data in order to use D2R. Like this we loose all the spacial computation facilities normally available in GIS environments and we had to replicate them using AllegroGraph geo-extention. Then, for performances reasons, for two of the three path finding strategies we discovered that it was more convenient to use physical RDF dumps instead of virtual RDF access. By doing so, we partially violate our principle 2. We believe that a mapping tool that natively wraps GIS can foster the use of Semantic Web technologies in combination with geo-spatial ones. Such a mapping tool will be for GIS what D2R [16] is for relational databases. It will allow spacial computation to be executed in a GIS

environment in accordance with principle 5 and allowing to fully comply with principle 2.

However, if we compare what we were able to realize with the requirements and challenges for the Semantic Web we defined in [27], 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. Moreover, 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 another possible advancements of our work. We intend to extend the current demonstrator to a world-wide scale by integrating the street topology of OpenStreetMap[25] made available by the LinkedGeoData project[26].

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)

[25] OpenStreetMap `http://www.openstreetmap.org/`.
[26] LinkedGeoData `http://linkedgeodata.org/`.

3. Visser, U.: Intelligent Information Integration For The Semantic Web (Lecture Notes in Computer Science). SpringerVerlag (2005)
4. 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)
5. Kiryakov, A.: Measurable targets for scalable reasoning (2007)
6. McGuinness, D.L., van Harmelen, F.: Owl web ontology language overview. W3c recommendation, W3C (2004)
7. Lab., O.: OWLIM - Pragmatic OWL Semantic Repository (2007)
8. Ma, L., Yang, Y., Qiu, Z., Xie, G.T., Pan, Y., Liu, S.: Towards a complete owl ontology benchmark. In: ESWC. (2006) 125–139
9. Brickley, D., Guha, R.: Resource description framework (RDF) schema specification. Technical report, W3C (1999) W3C Recommendation. http://www.w3.org/TR/PR-rdf-schema/.
10. Guo, Y., Pan, Z., Heflin, J.: Lubm: A benchmark for owl knowledge base systems. J. Web Sem. **3**(2-3) (2005) 158–182
11. Fensel, D., van Harmelen, F.: Unifying Reasoning and Search to Web Scale. IEEE Internet Computing **11**(2) (2007) 96–95
12. Manola, F., Miller, E.: RDF Primer - W3C Recommendation. Available on the Web at `http://www.w3.org/TR/rdf-primer/` (10 February 2004)
13. 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)
14. 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)
15. 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)
16. 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)
17. 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)
18. Uschold, M.: Where are the semantics in the semantic web? AI Magazine **24**(3) (2003) 25–36
19. Berners-Lee, T.: Putting Government Data online - W3C Design Issue. Available on the Web at `http://www.w3.org/DesignIssues/GovData.html` (June 2009)
20. 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)
21. Fielding, R.: Representational state transfer (REST). Ph. D. Thesis, University of California, Irvine, CA (2000)
22. Sanders, P., Schultes, D.: Engineering fast route planning algorithms. In Demetrescu, C., ed.: Experimental Algorithms, 6th International Workshop, WEA 2007, Rome, Italy, June 6-8, 2007, Proceedings. Volume 4525 of Lecture Notes in Computer Science., Springer (2007) 23–36

23. Sanders, P., Schultes, D.: Engineering highway hierarchies. In Azar, Y., Erlebach, T., eds.: Algorithms - ESA 2006, 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006, Proceedings. Volume 4168 of Lecture Notes in Computer Science., Springer (2006) 804–816
24. Dell'Aglio, D., Celino, I., Kim, E.D.K., Park, S., Steinke, F.: D6.6 - 2nd periodic report on data and performances (2009) Available from: http://www.larkc.eu/deliverables/.
25. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. IEEE Trans. Knowl. Data Eng. **19**(1) (2007) 1–16
26. Volz, R., Kleb, J., Mueller, W.: Towards ontology-based disambiguation of geographical identifiers. [28]
27. 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)
28. Bouquet, P., Stoermer, H., Tummarello, G., Halpin, H., eds.: Proceedings of the WWW2007 Workshop I^3: Identity, Identifiers, Identification, Entity-Centric Approaches to Information and Knowledge Management on the Web, Banff, Canada, May 8, 2007. [28]