



## **SEVENTH FRAMEWORK PROGRAMME**

### **Networked Media**

*Specific Targeted Research Project*

# **SMART**

(FP7-287583)

## **Search engine for Multimed*i*A environment generated conten*T***

### **D5.2 Report on Query Submission, Processing and Routing**

Due date of deliverable: 01-08-2013

Actual submission date: 12-09-2013

Start date of project: 01-11-2011

Duration: 36 months

### Summary of the document

<b>Code:</b>	<b>D5.2 Report on Query Submission, Processing and Routing</b>
<b>Last modification:</b>	12/09/2013
<b>State:</b>	Final
<b>Participant Partner(s):</b>	GLA, AIT, ATOS
<b>Author(s):</b>	Dyaa Albakour, Aristodemos Pnevmatikakis, María Angeles Sanguino
<b>Fragment:</b>	No
<b>Audience:</b>	<input checked="" type="checkbox"/> public <input type="checkbox"/> restricted <input type="checkbox"/> internal
<b>Abstract:</b>	A report on the techniques for formulating, processing and routing queries that are integrated within the SMART Search Layer. In particular, we build the required infrastructure for matching user queries against sensor updates and locations where SMART Edge Nodes are deployed.
<b>Keywords:</b>	<ul style="list-style-type: none"><li>• Processing user queries</li><li>• Matching user locations</li><li>• Geospatial indices</li><li>• Linked data</li><li>• Extracting location keyword descriptions</li></ul>
<b>References:</b>	DoW, see the References Section

---

## Table of Contents

1	Executive Summary .....	4
1.1	Scope .....	4
1.2	Audience .....	4
1.3	Summary.....	4
1.4	Structure.....	4
2	Introduction.....	6
2.1	Overview of the SMART Search Architecture.....	6
2.2	Interaction paradigms with the Search Engine .....	7
3	Matching user queries .....	10
4	Geospatial Indexing and Retrieval .....	12
4.1	Requirements .....	12
4.2	Indexing geospatial data .....	12
4.3	Retrieval of geospatial data .....	14
5	Metadata Description in SMART Edge Nodes .....	16
5.1	Overview of the feed descriptors in SMART edge nodes .....	16
5.2	Feed descriptions.....	16
5.2.1	Feed.....	16
5.2.2	Components.....	16
5.2.3	Outputs .....	17
5.3	Dictionary of descriptors.....	17
5.3.1	Example of use .....	18
6	Automatic Extraction of Keyword Descriptions .....	20
6.1	Overview of Linked Data .....	20
6.1.1	Linked Data Cloud .....	20
6.1.1	Datasets for SMART Consumption.....	21
6.2	The SMART Linked Data Manager.....	22
6.2.1	Collecting geospatial information.....	22
6.2.2	Implementation with SPARQL queries .....	23
6.3	Extracting location metadata.....	27
6.4	Evaluation .....	30
7	Conclusions .....	33
8	BIBLIOGRAPHY AND REFERENCES .....	34
	Appendix A List of used locations for evaluation.....	35

## 1 Executive Summary

### 1.1 Scope

SMART allows users and applications to ask for information stemming from the physical world in real-time. The Search Layer is a high level component that digests streams of information originating from sensors and social media in order to answer information needs about what is happening in the physical world. This document describes how information needs can be expressed to SMART by the users and/or applications and how they are matched against the metadata generated by sensors and the social media content. Furthermore, it describes the main components in the SMART Search Layer that supports processing and handling user queries.

### 1.2 Audience

The content of this deliverable can be of interest to a wide range of individuals within these groups:

- **The Information Retrieval (IR) community:** The challenges addressed in this deliverable are relevant to IR researchers and practitioners. Our solutions introduce novel approaches to derive localised textual descriptions associated to events and activities in certain areas.
- **SMART software developers:** The developers of the SMART project, notably those dealing with the open source software implementation of SMART components in the search layer and SMART applications/visualisation libraries.
- **SMART project members:** The deliverable gives insight to all project members involved in delivering the SMART concept. Notably, members involved in the development of WP3, WP4 and WP6 can better understand how the Search layer processes user queries and matches them against the available sensor metadata streams from different SMART Edge Nodes.
- **The Open source community:** The open source community, notably the community that we are building around the SMART results, will use the present deliverable as a guide towards understanding how users interact with the Search Layer and in return how we model these interactions and process the queries to find the best matched results.

### 1.3 Summary

This deliverable gives an overview of the query processing component of the SMART Search Layer and the required infrastructure we develop to address the challenges in this regard. We detail the interaction paradigms with the Search Layer and identify the type of queries that are supported by SMART. Since locality is an important aspect of SMART, we further describe how the SMART Search Layer supports the geospatial indexing and retrieval. Furthermore, we describe how the sensor metadata is handled by the Search layer. On the other hand, this approach can be automated by using the linked data cloud, which we leverage to deliver useful keywords describing activities that can be associated with certain locations.

### 1.4 Structure

The deliverable is structured as follows:

- Section 2 provides an introduction and an overview of the user interaction paradigms that SMART supports to motivate the query processing techniques we develop in this deliverable.
- In Section 3, we detail the query matching requirements in SMART that we address in this deliverable.

- In Section 4, we describe how we extend the open source Terrier search engine, the underlying search technology in SMART, to perform geospatial indexing and retrieval in order to match the user location against the search results.
- Section 5 describes the mechanisms provided at the Edge Node level for describing sensor metadata such that the user queries can be matched to the sensor feeds.
- Section 6 introduces our Linked Data approach for automatically extracting textual description of the location where SMART Edge Nodes are deployed.
- Finally, Section 7 summaries the conclusions and gives an outlook for the next versions of the related deliverables in WP5.

## 2 Introduction

SMART aims to deliver a framework for processing large streams of data about the environment originating from sensors in order to satisfy information needs about the physical world that are not usually served by existing search engines. In particular, SMART allows end-users and applications to answer their information needs about what is happening in the physical world in real-time.

The Search Layer is a higher level component in SMART that deals with the representation, storage, organization and access to the information provided by the lower-level SMART Edge Nodes, which connect to the physical sensors and produce metadata streams describing their output. More specifically, the Search Layer offers an efficient real-time *indexing* of sensor metadata and social streams in order to effectively *retrieve* and rank events that satisfy the information needs of a SMART end-user or an application. In this document, we describe how information needs can be expressed to SMART by the end-users and/or applications and how they are matched against the environment data generated by sensors and the social media content. The deliverable mainly discusses the techniques we have developed for handling user queries and matching them against the indexed sensor metadata and social streams. The efficient indexing infrastructure and the specialised retrieval models are described in other Search Layer related deliverables, namely [SMART-D5.1] and [SMART-D5.3].

In this section, we first give an overview of the overall SMART search architecture to identify where the processing and matching components fit within this architecture and outline their high level implementation details. Next, we give an overview of the user interaction paradigms with the search engine to motivate our approach in developing the required infrastructure for the query processing component.

### 2.1 Overview of the SMART Search Architecture

The Search Layer in the SMART framework is composed of the software components that provide services and applications with easy *real-time* access to information stemming from both the social and sensor media streams [SMART-D2.3]. The Search Layer deals with the representation, storage, organisation and access to the information provided by the entire population of SMART Edge Nodes and social media networks. Figure 1 illustrates the logical structure of the Search Layer, where the main software components are identified and the information flows between these components are also illustrated. In this deliverable we develop the core functions of two Search Layer components: (a) the *query processing component*; and (b) the *Matching, Retrieval and Ranking component*

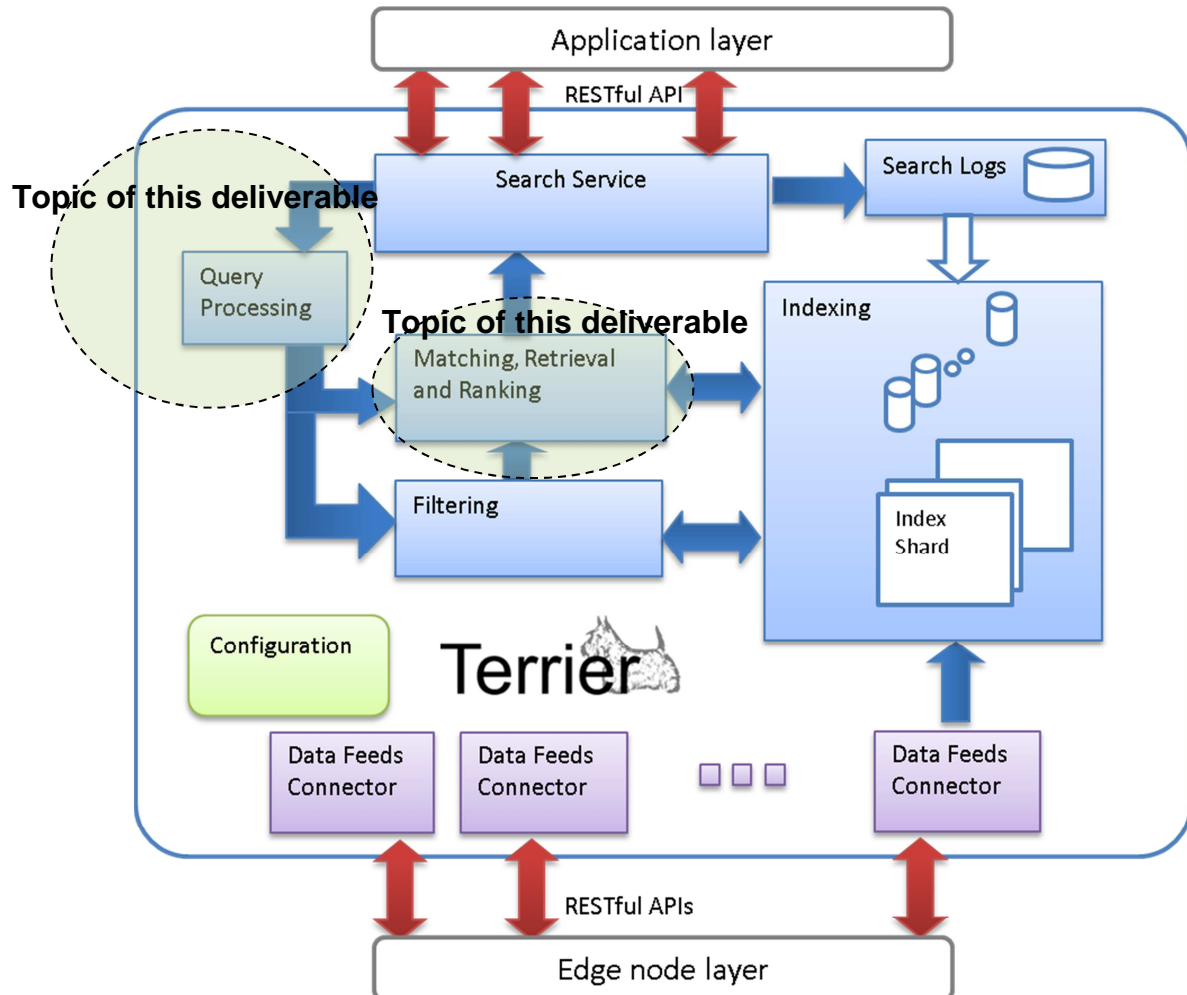
(a) The *Query Processing* component is responsible for identifying the user information needs through the user interaction with SMART applications. In particular, these queries can be explicitly submitted by the user or inferred from the user's context. The query processing component prepares the query to the matching component such that it is efficiently and effectively answered with the most relevant results from the index component.

(b) The *Matching, Retrieval and Ranking* component is responsible for matching a user query against the index to identify and rank events according to how they satisfy the user's information needs.

In a classical IR system, the matching process involves identifying a subset of all the indexed documents that partially match the user information needs such that they are scored and ranked for a user query to find the best matching ones. In SMART, the index stores real-time streams of updates from physical sensor feeds and social media feeds from Edge Nodes in various geographical locations. The matching process involves identifying a subset of locations and sensor feeds related to the query such that events in the physical world are detected and ranked to answer the user query (i.e. routing the query to right sensors and locations).

In the query scoring and anticipation subsystems deliverable [SMART-D5.3], we have developed new retrieval models that can identify local events across different locations covered by the SMART edge nodes from the sensor metadata streams, which forms the retrieval and ranking part of this

component. In this deliverable, we develop the infrastructure needed for the matching mechanisms in SMART to support these retrieval models.



**Figure 1: Search Layer Architecture [Smart-D5.1] (the components covered in this deliverable are highlighted: (i) Query Processing, (ii) Matching, Retrieval and Ranking)**

In order to understand the requirements for the query processing and matching components, we first describe the two main user interaction paradigms that are possible in SMART applications. This would allow us to define what the query looks like and what are the structures and the resources needed within the SMART Search Layer to match the query and rank the results.

## 2.2 Interaction paradigms with the Search Engine

SMART applications offer users and services access to the high level information inferred by correlating a large number of sensor and social streams. Regardless of the interfaces or the particular use cases of an end user application/service, the interaction with the Search Layer can be classified into one of three main paradigms

- **Explicit queries**

In this case an explicit keyword query is submitted to the search engine at a certain time. Figure 2 demonstrates an example of this paradigm. It shows an example of a user interaction with a



possible map interface. Upon submitting the query 'music', the Search Layer has retrieved the music events from the sensor and social observations that are collected in real-time and displayed them as on the map.



Figure 2 Illustration of explicit keyword query interaction

- **Implicit queries**

In this case no explicit keyword query may be submitted to the search engine. Instead the search engine relies on the context to find relevant results. The context is mainly characterised by the location of the user or the location specified by the service/application.

Figure 3 sketches an example of this paradigm. It shows an example of a user interaction with a possible map interface that can be displayed on a mobile device with positioning capabilities. The location of the user in this case specifies the query of the user and the Search Layer ranks events closer to the user and displays them on the map.

Although the context may involve other factors such as the profile of the user on social media, but the location remains a unique requirement for matching these types of queries as it involves representing geospatial information within the data structures of the index.

- **Hybrid queries**

In this case, the query is specified by both the keyword query provided by the user and the context of the user which includes the location. For example, in Figure 3, in addition to the location, the user may explicitly enter a keyword query (e.g. 'music') and in this case the Search Layer should use both the location of the user and the keyword query to rank higher the results that match the topic of the keyword query and that are closer to the user's location.

In the next section, we describe our retrieval framework in SMART in order to identify the main requirements in matching user queries. This description would also help us in identifying the data structures and the resources required to build the infrastructure for matching queries in SMART.



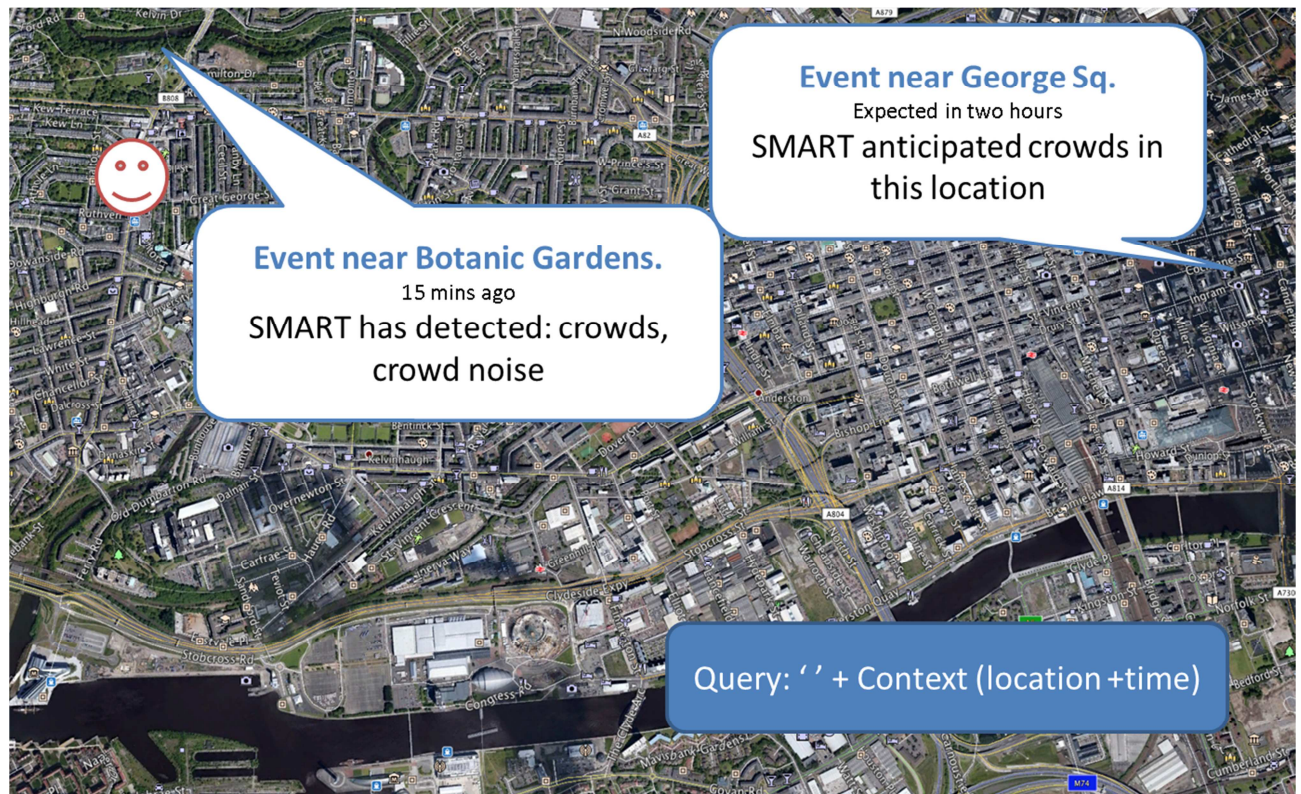


Figure 3 Illustration of implicit query interaction

### 3 Matching user queries

In this section, we describe our event retrieval framework in the SMART search layer that we have developed and evaluated extensively in a previous deliverable [SMART-D5.3]. In this description we will identify the requirements for query processing and matching in SMART and the infrastructure we are building in this deliverable to support these requirements.

Our retrieval framework aims to define an effective ranking function that scores pairs of time and location according to how likely they represent the starting time and the location of a relevant event for the user's information needs. More formally, and referring back to the interaction paradigms described in Section 2.2, the information needs of a user  $u$  are specified by a keyword query  $q$  submitted at a certain time  $t_q$  and/or a context of a user identified at least by the location of the user  $l_q$ . We consider a finite set of locations  $L = \{l_1, l_2, \dots, l_n\}$ , where we observe the physical world from the sensor feeds and the social media posts in SMART Edge Nodes. Each location  $l_i$  at a certain time  $t_j$  is denoted by the tuple  $\langle l_i, t_j \rangle$ . The event retrieval framework aims to score tuples  $\langle l_i, t_j \rangle$  according to how likely  $t_j$  represents a starting time of an event within the location  $l_i$  that matches the user's information needs. An event is considered relevant if it matches the explicit keyword query  $q$  of the user and/or the implicit context of the user (the location of the user  $l_q$  and/or her profile). In other words, the event retrieval framework defines a ranking function that gives a score  $R(q, u, \langle l_i, t_j \rangle)$  for each tuple  $\langle l_i, t_j \rangle$ . Our event retrieval framework developed in [SMART-D5.3] uses the sensor feeds and the social streams collected from SMART Edge Nodes as the main source of evidence to score the tuples. In particular, two components are built on this evidence, namely topicality and change as explained below.

#### 1. The Topical Component:

This component is based on the intuition that the description of metadata generated from the conceptual processing components about the sensor observations in a certain location are topically related to the user query. For example, low-level events detected in a feed of crowd metadata from the video analysis of a camera observing vehicles in a main road are topically related to queries like 'traffic', whereas those detected in crowd feed of main square may be topically related to queries like 'protests', 'festival' or other keywords describing activities that may occur in that particular square.

Moreover, and with regards to social media observations, the topical component is also based on the intuition that social media may reflect real world events, hence when an event occurs somewhere we expect to find topically related social posts about it originating from the location where it occurs [Zubiaga2011]. In this case, to measure the topical component, for each location at a given time, i.e. for each tuple of location and time  $\langle l_i, t_j \rangle$  we quantify how much the social media posts corresponding to the tuple (e.g. tweets  $T_{i,j}$  originated from location  $l_i$  at time  $t_j$ ) are topically related to the query  $q$ .

While in [SMART-D5.3], we proposed an approach for making the quantification of the topical component in social media post by borrowing ideas from the IR problem of expert search [Macdonald2006], in this deliverable, we aim to measure this component from mainly the physical sensor feeds. In particular, we devise methods for extracting keyword descriptions of the sensor feeds and their location. More specifically, Section 5 describes the mechanisms developed at the Edge Node level to describe the metadata feeds, while in Section 6 we develop an auto-

matic approach to derive keyword descriptions of the sensors feeds and the location where they are deployed.

## 2. The Change Component:

The second component is based on the intuition that events trigger an *increasing* activity measured by the various sensors (both physical and social sensors) that is also *unusual*. For example, a sudden increase in the crowd level observed in a certain area that is not anticipated or an increasing tweeting activity causing peaks of tweeting rates during the event (bursts) [Zubiaga2011]. For this component, we developed methods to quantify the *unusual change* in the sensor observations [SMART-D5.3].

Following this, and considering only an explicit keyword query  $q$ , the ranking function can be defined as a linear combination as follows:

$$R(q, \langle l_i, t_j \rangle) \propto (1 - \lambda) \cdot S(q, \langle l_i, t_j \rangle) + \lambda E(q, \langle l_i, t_j \rangle) \quad (1)$$

where:

- $S(q, \langle l_i, t_j \rangle)$  quantifies how much the sensor observations at  $\langle l_i, t_j \rangle$  are topically related to the explicit query  $q$ ; and
- $E(q, \langle l_i, t_j \rangle)$  is a score proportional to the probability that an event is about to start in the location  $l_i$  at time  $t_j$  as can be indicated from the sensor observations, and  $0 \leq \lambda \leq 1$  is a parameter to control the contribution for each component in the linear combination in Equation (1).

The ranking function can be extended to take into account the context of the user:

$$R(q, u, \langle l_i, t_j \rangle) \propto (1 - \lambda) \cdot S(q, u, \langle l_i, t_j \rangle) + \lambda E(q, \langle l_i, t_j \rangle) \quad (2)$$

Where  $S(q, u, \langle l_i, t_j \rangle)$  quantifies how much the sensor observations at  $\langle l_i, t_j \rangle$  are topically related to the explicit query  $q$  and to the context of the user  $u$  (the location of the user  $l_q$  and/or his profile). For example, in this case, locations closer to the user should be ranked higher.

Since the location is an important aspect of the user context, it is important to build efficient index structures that support representing the location of the sensor observations and enable us to effectively quantify the score  $S(q, u, \langle l_i, t_j \rangle)$ . In Section 4, we describe how we extend the traditional index of Terrier in order to support geospatial indexing and retrieval to enable representing locations at the Search Layer and allow matching the location of the user to the results.

To summarise, there two important aspects during processing and matching user  $q$ , which are both incorporated in quantifying the first component of the ranking function. Namely, (i) matching the location of the user against nearby locations and (ii) matching the query against a keyword description of the sensor metadata in a certain location and the social media posts. Section 4 deals with the building the geospatial infrastructure required for the first aspect. Sections 5 and 6 deal with second aspect where we introduce two different approaches for extracting keyword descriptions about the sensor metadata in a certain location.

## 4 Geospatial Indexing and Retrieval

As discussed above in Section 3, the user location is an important aspect of the user context that needs to be matched against geographically relevant (e.g. of a close distance) to SMART Edge Node locations in our event retrieval model. Hence, representing the geographical location of the sensor updates and the information needs of the user is an important issue that needs to be addressed at the Search Layer. Therefore, we have extended the Terrier<sup>1</sup> open source search engine (the underlying technology that the SMART Search Layer relies upon) to support geo-spatial indexing and retrieval.

In this section, we first identify the concrete requirements for geo-spatial indexing and retrieval in SMART. Furthermore, we explain how we have extended Terrier to support indexing geo-spatial information (Section 4.2) and retrieving geospatial information (Section 4.3).

### 4.1 Requirements

The requirements for geospatial indexing and retrieval in SMART stems from the interaction paradigms described in Section 2.2, which were then compiled into the matching process and the retrieval model described in Section 3. In particular, in the following, we identify the concrete requirements for supporting geospatial search in SMART:

- The Search Layer and its API interface should allow users and applications to specify the location of interest (i.e. location of the user) using the traditional geographical coordinate system. In this system, each location on the earth is specified by a pair of real numbers representing the latitude (a real value between -90 and +90) and longitude (a real value between -180 and +180) of the location.
- The Search Layer and its API interface should allow users and applications to also specify a region of interest to find results within. This region should be specified using the traditional geographical coordinate system. In particular, the region can be specified using a certain radius around the user location or by specifying a polygon bounding the area.

In both cases, during retrieval, the Search Layer should be able to either filter the search results to include only within the specified geographical region, and/or boost those which are closer to user.

To meet these requirements, the underlying indexing infrastructure which relies on open source Terrier [SMART-D5.1] needs to be extended to represent the location of both updates from SMART Edge Nodes and the location of the user (the implicit query). Moreover, the matching and retrieval process involves additional steps in order to match the user location and score the results accordingly. Next, we show how we achieve these objectives by extending our open source Terrier platform with the required data structures and processes for geospatial indexing and retrieval. In our implementation, we follow a modular approach where we make use of the available interfaces in Terrier, which allows us to make full potential of its current index data structures and retrieval processes.

### 4.2 Indexing geospatial data

In [SMART-D5.1], we discussed the anatomy of the Terrier indices used in SMART highlighting its main data structures. These include a lexicon and the postings lists of the inverted index. The lexicon (vocabulary or dictionary) contains a list of all the terms in the collection sorted alphabetically. A posting list for a given term within the inverted index contains a posting for each document (updates from SMART Edge Node feeds) in which that term occurs, with a posting giving the document identifier in which that term appears and how often (term frequency). Thus, the classical inverted index structure allows the documents that contain a term to be quickly identified. This inverted index is used during the retrieval process to match the user explicit keyword query and in particular estimate the topical component in Equations (1) and (2).

In order to represent the location of the documents (the Edge Node updates) within our index, we need

---

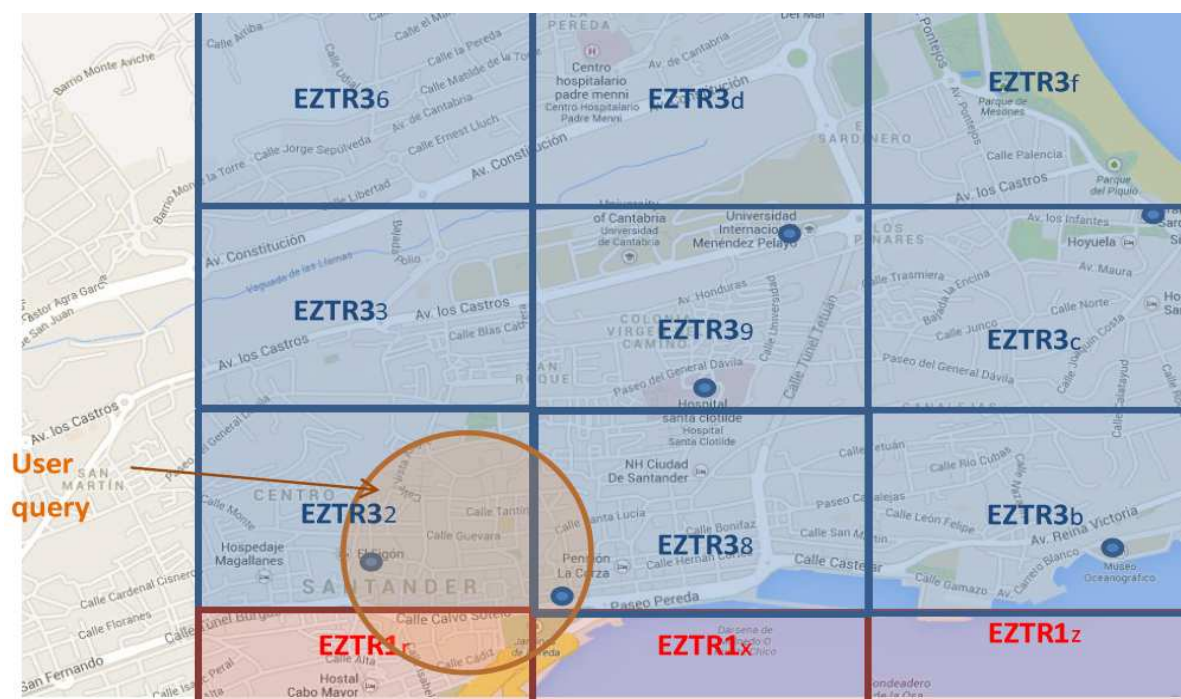
<sup>1</sup> <http://terrier.org>



to build additional data structures that store the location of the document identified by the geographical coordinates of the feeds (a pair of latitude and longitude), such that it can be looked up later and efficiently matched to a user location. In particular, both coordinates (latitude and longitude) need to be combined to create a value that represents both dimensions.

Our approach is to employ a popular function that has been used extensively to build geo-spatial indices in database systems (e.g. MongoDB<sup>2</sup>) or search engines (e.g. Apache Lucene<sup>3</sup>). In particular, we use the “geohashing function” which was built as a geocoding system for the web service (<http://geohash.org>) and offers unique URLs for positions on earth.

A geohash is a function that turns a pair of coordinate values into a single string value (a code) [Geohash]. Geohashes are released in the public domain and do not require any license. Geohashes offer a hierarchical geo-spatial structure, which divides space into a grid of areas recursively sharing the same prefix of geohash code. In this hierarchical spatial structure, places close to each other will often but not always exhibit similar prefixes. Conversely, the longer a shared prefix is, the closer the two places are. More precisely, when calculating a geohash, a precision is needed. The highest precision can be up to 12 characters, which is capable of representing a location with a distance error of only 2 meters. By truncating characters from the end of the geohash code, we get a less precise geohash and a correspondingly less precise (larger) selection of the map. In other words, while full precision effectively matches a point, partial precision represents a bounding box around an area in the geo-spatial space. Figure 4 illustrates the geohashes of precision 6 for the city of Santander. As we can observe, with the decreasing precision of a geohash, we obtain a smaller selection of the map. Areas in blue have the same prefix which corresponds to the lower precision geohash of 5 characters. Similarly areas in red have the same prefix.



**Figure 4: Example of geohashes of precision 6 for the city of Santander (Note that the blue areas share the same prefix ‘eztr3’, while the red ones share the prefix ‘eztr1’)**

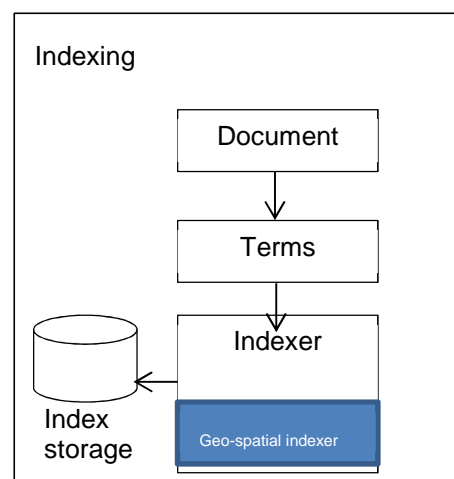
<sup>2</sup> <http://www.mongodb.org/>

<sup>3</sup> <http://lucene.apache.org/core/>

Geohashes are the basis for the geospatial inverted index that we aim to build and extend our Terrier indexing with. In this inverted index, documents can be mapped to a geohash they belong to in the exact same way an inverted index maps terms to a list of documents. As we will discuss later, in retrieval, the implicit query can contain a geohash which encodes the user's location and therefore we need to retrieve results from locations that match that geohash. It also means there is a possibility of querying large areas by using small geohashes and comparing it to the prefixes rather than exact matches to just one geohash.

We follow a modular approach for creating this geospatial inverted index in Terrier. In particular, we create a new type of posting lists, a geospatial posting list, where we extend the traditional Terrier "Posting List" used in the traditional inverted index. A geospatial posting list stores for each geohash prefix a list of document identifiers that correspond to the location of the geohash prefix.

In order to build the geospatial index, the indexing process is also extended as shown in Figure 5. In addition to building the inverted term index, the indexer has an additional process for building the geospatial inverted index. This process involves calculating the geohash for the location of each incoming document as specified in the metadata of the Edge Node feed, and building/updating the geospatial posting lists with the identifiers of those documents.



**Figure 5: Extended Terrier Indexing Process**

Next we show how the data structures we have built supports the retrieval process in matching and ranking locations geographically relevant (close) to the user location.

### 4.3 Retrieval of geospatial data

During retrieval, there needs to be a way of gathering the user's location. As described in the requirements (Section 4.1), the location could be gathered from the exact coordinates of the user together with either a distance or a query shape that will be specified by the end user application using some mapping GUI. More precisely, and with reference to Equation (2), the geo-spatial index would make it easier to match the user location and decide whether a location is relevant to the user, which would help in quantifying the final ranking score.

One possible avenue for geospatial matching is to encode the user location in a geohash and only consider those documents (updates from locations), which share the same geohash prefix at a certain precision. However, although it is true that points which share the same long geohash prefix are close to each other, the opposite is not always true. See for example Figure 4 (the blue areas share the same prefix 'eztr3' and some of them share edges with the red ones, which have the prefix 'eztr1'). Edges ex-

ist at all levels of geohash precisions and as a result a nearby location to the user may have a different geohash.

However, this can still be achieved with the geohash-based index. Matching and retrieval using geospatial posting lists follows that of a typical Document-at-a-time (DAAT) retrieval [Moffat1996] used by search engines. In particular, if events close to the query location are to be boosted, then the geospatial posting list can be processed like an additional query term. If events out with the query location are not to be retrieved, where the query location defines a filter, then we can use the geospatial posting list to more aggressively prune retrieval, such that only events within the query shape are considered.

We describe our methods for matching the user location with the example in Figure 4 in which a user query area is specified. Considering a 6 character geohash, and geospatial index with the posting lists of geohashes shown in Figure 6, the procedure of matching the user location can be summarised as follows (it is implemented in a modular way using Terrier interfaces for retrieval processes):

- We first identify candidate grids that overlap with the query shape. The result will be the geohashes highlighted in the table (also see Figure 4 for more clarification on how these can be identified; we calculate the possible geo-hashtes that intersect with the query shape).
- For all the items in the posting list of each considered grid, we decode the location of the document (which is provided in the meta index), and intersect it with the query shape (either filter results or boost those closer to exact user location).

Finally, DAAT Dynamic pruning strategies such as WAND [Broder2006] can be used to filter results within the query shape. Future version of deliverable D5.1 will consider efficiency considerations of geospatial posting lists.

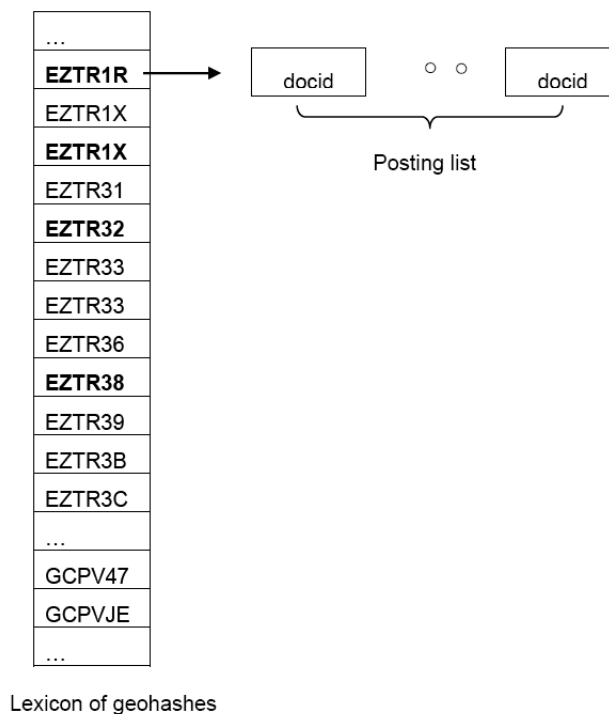


Figure 6 Illustration of the geospatial retrieval process



## 5 Metadata Description in SMART Edge Nodes

SMART Edge Nodes produces metadata feeds about the environment that are mainly perceived from the physical surroundings. Any feed gives metadata describing something. For example:

- A temperature feed describes the measured temperature.
- A visual density feed describes the density of moving objects in the field of the view of the camera.

All this information is useful, but it needs to be put in some context:

- Where is the temperature measured? Certainly somewhere in the vicinity of the edge node. But indoors or outdoors? At which location?
- Where are the moving objects whose density is measured? If outdoors, are they most likely pedestrians or vehicles?

To answer these questions that put the measurements carried by the feeds into useful context, every feed needs to be accompanied by descriptors that help the search engine understand if it is useful for a particular query.

This section details the content and the formatting of the feed descriptors. We mainly aim to show how the Edge Node supports the manual description of the feeds it produces in order to make it possible for the Search Layer to match explicit keyword queries against these metadata feeds.

### 5.1 Overview of the feed descriptors in SMART edge nodes

Every Edge Node feed is generated using an XML describing the feed in general (what it is about), its components (the physical and/or virtual devices that take the measurements) and its outputs (the different measurements), as described in [SMART D3.1]. The feed description can carry general geographic information. The component description can carry specific localisation information. This is particularly useful for virtual components, that imply some sort of processing can be very specific, as e.g. only a portion of a camera feed corresponding to a particular place can be employed in the underlying measurement.

Finally, the output description carries information about what exactly is measured.

### 5.2 Feed descriptions

This section considers the three levels of feed description in detail.

#### 5.2.1 Feed

The feed can receive information from multiple components, either physical (sensors) or virtual. Although in principle it can contain diverse metadata, this is not recommended. Its physical components are also somewhat distributed in space, but not by much. Its virtual components can vary a lot, since although they can stem from a single sensor, different processing can extract information from different locations. A typical example is a camera, where the signal is the sequence of frames. The locations depicted in the frames can vary a lot, especially in outdoors cameras where the field of view is large in order for people's faces not to be recognizable. For these reasons only crude geographical information can be found in the `<Feed>` section.

The location information for the feed is stored in the `<DescriptionTags>` section, either in human-readable lists of terms (`<TextTags>`), or more formally using `<URITags>`.

#### 5.2.2 Components

Every sensor is mapped into the feed XML description either as a physical component, or as one or more virtual components. The components refer to a very limited space and can be described using the following properties, found in the `<Physical>` or `<Virtual>` sections of the `<Components>`:

- **<Geolocation>**: (**<Longitude>**, **<Latitude>**) holds the GPS coordinates.
- **<Exposure>**: Outdoor or indoor.
- **<DescriptionTags>**: (usually **<TextTags>**, but also **<URITags>** are supported) contain human readable names for refined location description.

### 5.2.3 Outputs

Each component drives one or more numerical outputs, the measurements themselves. The descriptors for the outputs should provide information about what the measurements are about or what they refer to, e.g. “visual density attributed to pedestrians”.

This information for each output is found in the relevant **<Output>** section in the **<DescriptionTags>** section, either in human-readable lists of terms (**<TextTags>**), or more formally using **<URITags>**.

## 5.3 Dictionary of descriptors

Consider a feed reporting visual density numbers. These numbers correspond to outdoors pedestrian density at the square in front of the Santander city hall. From the metadata (output values) we know that the place is dense with foreground objects or not, but we have no information where this place is and what is causing the density (or at least who are the expected actors causing the density). In order for this information to be more useful, we add descriptors from a dictionary of terms that will convey the information needed to the search engine. The Search Layer can decide if the metadata are relevant to a particular query and can use them as described in Section 3 to quantify the topical component score. (e.g. a query ‘shopping’ will match those feeds which have ‘shopping’ in their text descriptors).

The dictionary is expected to be expanding continuously, as more measurement types are made available. The current version is given in Table 1. The descriptors are given in the third column, and are organised based on what they describe (location, measurement type, or expected cause) and also on the section of the XML description they are to be found (feed, component or output).

**Table 1: Dictionary of descriptors (third column), organised per what they describe (first column) and where in the XML description they are to be found (second column).**

Descriptor for	Found in	Descriptor/ term
Location	Feed or component	Human readable names for: Street, neighbourhood, building
		Indoors: Mall, education, entertainment, food, drink, transportation
		Outdoors: Square, street, park, transportation, shopping
	Component	GPS coordinates
		Human readable names for refined location description: Zebra crossing, particular store, etc.

Measurement type	Output	Audio intensity, audio event, visual density, temperature, etc.
Expected cause	Output	What is causing the output reading: Vehicle, people, etc.

### 5.3.1 Example of use

Consider a feed that is about visual density at the area of the town hall at Santander. This location is expressed in the `<DescriptionTags>` of the `<Feed>` section.

The visual density is produced by processing a physical component, the camera (described by the `<Physical>` section) that is positioned outdoors (expressed using the `<Exposure>` property) at the coordinates specified in the `<Geolocation>` section of the `<Physical>` description.

The camera view is split into two areas, the square in front of the town hall and the road (Calle Monasterio) further ahead. The split is expressed in the feed description by introducing two `<Virtual>` sections. Each virtual component is of `<Type> Visual_Processing`. The component related to the street is described as a shopping street of the particular name utilizing its `<DescriptionTags>` section. The component related to the square is similarly described as a square of the particular name utilizing its `<DescriptionTags>` section.

Each virtual component has one `<Output>` associated to it. Both outputs are about visual density, but the one referring to the street has the density caused by both people and vehicles, while the other referring to the square has the density caused by only people. These are expressed using the `<DescriptionTags>` section of each `<Output>`.

The complete feed description XML, with all the properties and sections related to the descriptors mentioned above is given in Listing 1.

**Listing 1: The XML feed description for the example in section 5.3.1.**

```
<?xml version="1.0" encoding="UTF-8"?>
<Feed xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SMART_Datafeed_Schema_v0.3.xsd">
  ...
  <DescriptionTags>
    <TextTags>Europe Mediterranean Spain Cantabria Santander Town_Hall</TextTags>
  </DescriptionTags>
  <Components>
    <Physical>
      <Name>city_hall_camera</Name>
      ...
      <Geolocation>
        <Longitude>43.462165</Longitude>
        <Latitude>-3.810204</Latitude>
      </Geolocation>
      <Type>Camera</Type>
      <Exposure>Outdoor</Exposure>
      <Disposition>Fixed</Disposition>
    </Physical>
    <Virtual>
      <Name>calle_monasterio</Name>
      <Description>...</Description>
      <DescriptionTags>
        <TextTags>Street Shopping Calle_Monasterio</TextTags>
      </DescriptionTags>
      <Type>Visual_Processing</Type>
```

```
</Virtual>
<Virtual>
  <Name>plaza_ayuntamiento</Name>
  <Description>...</Description>
  <DescriptionTags>
    <TextTags>Square Plaza_Ayuntamiento</TextTags>
  </DescriptionTags>
  <Type>Visual_Processing</Type>
</Virtual>
</Components>
<Outputs>
  <Output>
    <Name>calle_monasterio_visual_density</Name>
    <ProducedBy>calle_monasterio</ProducedBy>
    <Description>...</Description>
    <DescriptionTags>
      <TextTags>Visual_Density People Vehicles</TextTags>
    </DescriptionTags>
    ...
  </Output>
  <Output>
    <Name>plaza_ayuntamiento_visual_density</Name>
    <ProducedBy>plaza_ayuntamiento</ProducedBy>
    <Description>...</Description>
    <DescriptionTags>
      <TextTags>Visual_Density People</TextTags>
    </DescriptionTags>
    ...
  </Output>
</Outputs>
</Feed>
```

## 6 Automatic Extraction of Keyword Descriptions

In this section, we propose a novel approach that uses the Linked Data (LD) cloud for the automatic extraction of keyword metadata that describe the locations, in smart cities, where SMART edge nodes may be operating. This approach can complement the previous manual one or mitigate the problem of having no keyword descriptions provided by Edge Node maintainers. The keywords extracted can guide the retrieval models to match events to user search keywords as described in Section 3. We first give the reader an overview of concept of Linked Data. Then we describe our approach and the evaluation we conducted to assess the quality of the keywords extracted with our approach.

### 6.1 Overview of Linked Data

As discussed in [SMART-D4.1], the **Web of Data** is a relatively recent effort derived from research on the Semantic Web, whose main objective is precisely to generate a Web exposing and interlinking data in a way such that it is directly amenable to automated processing. The Web of Data is based upon four simple principles, known as the LD principles<sup>4</sup>:

- The use of URIs to identify resources
- The use http protocol to dereference URIs
- Resources are represented in RDF and queries expressed using SPARQL
- Using RDF links to represent relationships

The basic processes of the LD life-cycle are:

- Publishing data refers to the activity of making data available into a global data space in RDF (Resource Framework Description) format. The main aim of this publication in a structured format is enabling applications to discover and to consume data.
- Linking data can be defined as the process of interlinking elements from different datasets. The idea is not publishing RDF-format data only from one dataset but also establishing mappings or referring data from other datasets.
- Consuming data is the activity of making use of the data available by applications or other data consumer like humans. This is possible through several mechanisms such as browsing the linked open data cloud<sup>5</sup> or querying the SPARQL end-point services.

SMART takes advantage of the consuming activity by exploiting information from existing LD datasets. Access of the entire dataset is made possible via RDF crawling, via an RDF dump, or via a SPARQL endpoint. In the following, we describe the concept of the LD cloud and the datasets we are proposing to use in SMART.

#### 6.1.1 Linked Data Cloud

Since the LD principles were outlined in 2006, there has been a large uptake; the latest analysis carried out in September 2011 indicates that there are over 295 datasets hosting nearly 31 billion RDF statements. The LD cloud is constantly growing with more datasets being contributed and made available; according to Datahub<sup>6</sup>, 337 dataset are currently found (as opposed to 295 in September 2011). Figure 7 shows a view of the current state of the LD cloud. More information can be found on the web page of the state of Linking Open Data cloud.<sup>7</sup>

---

<sup>4</sup> <http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/LinkedDataTutorial/#principles>

<sup>5</sup> <http://richard.cyganiak.de/2007/10/lod/>

<sup>6</sup> <http://datahub.io/>

<sup>7</sup> <http://lod-cloud.net/state/>



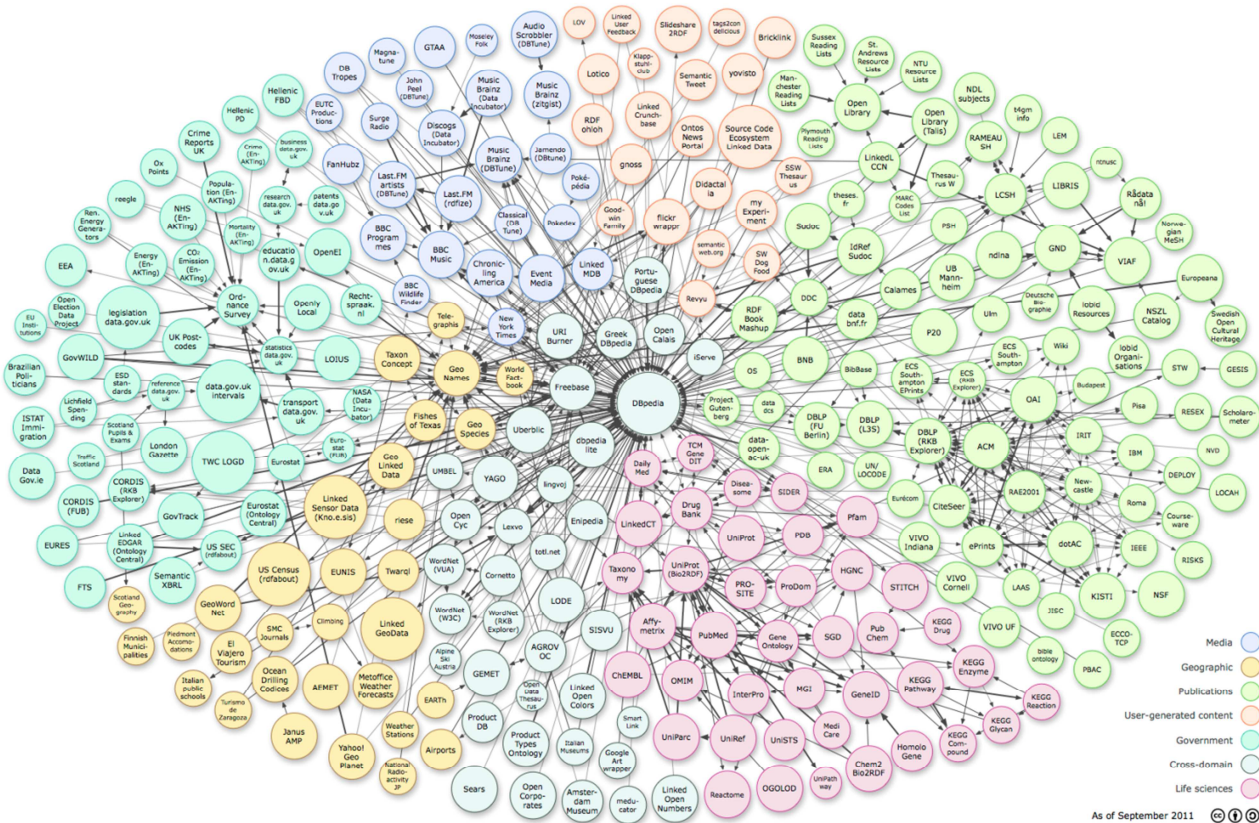


Figure 7: The Linking Open Data cloud diagram <http://lod-cloud.net/>

### 6.1.1 Datasets for SMART Consumption

We have identified the following datasets in the LD cloud that can be useful for the different components of SMART in general and for the metadata extraction in the Search Layer in particular:

- **DBpedia.**<sup>8</sup> DBpedia provides an RDF view of the Wikipedia content. It is located in the centre of the Linked Data cloud, as it provides links to almost all the rest of available datasets. DBpedia can be used via SPARQL using several existing end points, and is also available for download (<http://datahub.io/dataset/dbpedia>).
- **Geonames.**<sup>9</sup> Geonames is an open-license geographical database that publishes Linked Data about 8 million locations (<http://www.geonames.org>). SPARQL endpoints for this data are made available by third parties, e.g. FactForge.<sup>10</sup>
- **FactForge.**<sup>11</sup> FactForge represents a reasonable view to the web of data. It includes several of the most central datasets of LD and aims to allow users to find resources and facts based on the semantics of the data, like web search engines index. It is probably the largest and most heterogeneous body of general factual knowledge that was ever used for logical inference.

<sup>8</sup> <http://es.dbpedia.org>

<sup>9</sup> <http://www.geonames.org/>

<sup>10</sup> <http://factforge.net/>

<sup>11</sup> <http://factforge.net/>

FactForge includes the following dataset: DBpedia, New York Times<sup>12</sup>, MusicBrainz<sup>13</sup>, Lingvoj<sup>14</sup>, Lexvo<sup>15</sup>, CIA world Factbook<sup>16</sup>, Wordnet<sup>17</sup>, Geonames and Freebase<sup>18</sup>. The SPARQL endpoint is available at <http://factforge.net/sparql>. OWLIM<sup>19</sup> semantic repository is used to load the data and materialise the facts that could be inferred from it.

- **Geosparql**. Geosparql is a geographic query language for RDF data developed as standard by the Open Geospatial Consortium (OGC). Basically, it is a set of SPARQL extension functions for geographic information. The endpoint is available at <http://geosparql.org/>.

Next, we give a description of the Linked Data Manager in the SMART Edge Node and we describe the data that it can extract from the LD cloud to support the retrieval models of the Search Layer.

## 6.2 The SMART Linked Data Manager

The Linked Data Manager (LDM) is the software component in the SMART Edge Nodes that is responsible for collecting information from the LD cloud and make it available to the SMART Edge Node and the Search Layer. Figure 8 shows a high level view of the LDM architecture. The data tier contains the needed terminology to build the queries in order to collect data from the LD cloud datasets. It consists of concepts and properties used to annotate the dataset, as well as any particular special query constructions and extension functions for efficiently processing the geospatial information attached to the entities contained in the dataset. The logic tier contains the components in charge of processing the terminological information and builds the specialised LD queries, using SPARQL, to collect information from the desired datasets. The interface layer contains the REST API implementation that is responsible for exposing the provided functionality.

### 6.2.1 Collecting geospatial information

The LDM makes it possible to automatically extract spatial entities published across the web that are relevant to the location covered by a SMART Edge Node. In particular, it uses the available endpoints of the LD cloud to query billions of data items published around the web.

In the following, we discuss how the LDM can extract useful geospatial entities for a certain location covered by a SMART Edge Node that can be collected from the LD cloud. In particular, we identify two types of entities that can be useful for searching the physical world within the SMART framework:

- (1) Places/venues: these entities refer to actual venues, where the feeds from sensors in SMART Edge Nodes are observed and where actual events take place. This includes city squares, theatres, stadia, shopping centres, streets, and others.
- (2) Activities/events: these entities refer to concrete human activities, such as concerts and football matches, which happened in the past or planned at a certain time in the future.

To extract these entities, the LDM offers two possible avenues:

- Searching by location name: Finding venues and activities placed in a geographical area identified by a certain name. In particular, with this approach the LDM allows the following:
  - Finding venues located or events occurred in a certain area identified by a certain name. In

---

<sup>12</sup> <http://data.nytimes.com/>

<sup>13</sup> <http://datahub.io/dataset/data-incubator-musicbrainz>

<sup>14</sup> <http://datahub.io/es/dataset/lingvoj>

<sup>15</sup> <http://datahub.io/es/dataset/lexvo>

<sup>16</sup> <https://www.cia.gov/library/publications/the-world-factbook/>

<sup>17</sup> <http://wordnet.princeton.edu/>, <http://datahub.io/dataset/w3c-wordnet>

<sup>18</sup> <http://datahub.io/es/dataset/freebase>

<sup>19</sup> <http://www.ontotext.com/owlim>



this case, we can for example find all the venues/the activities in Covent Garden (a popular district in London) or we can restrict the search to a desired type of venues or activities, e.g. finding the restaurants in Covent Garden, or the concerts in Santander.

- Searching by geographical coordinates: Finding venues and activities located in a certain area defined by the exact geographical boundaries. In particular, with this approach the LDM allows the following:
  - Finding entities (venues/activities) located within a geospatial circle identified by the geographical coordinates (a pair of latitude and longitude) of its centre and its radius.
  - Finding entities within a geo-spatial rectangle identified by the exact geographical coordinates of its lower-left corner and its upper-right corner.

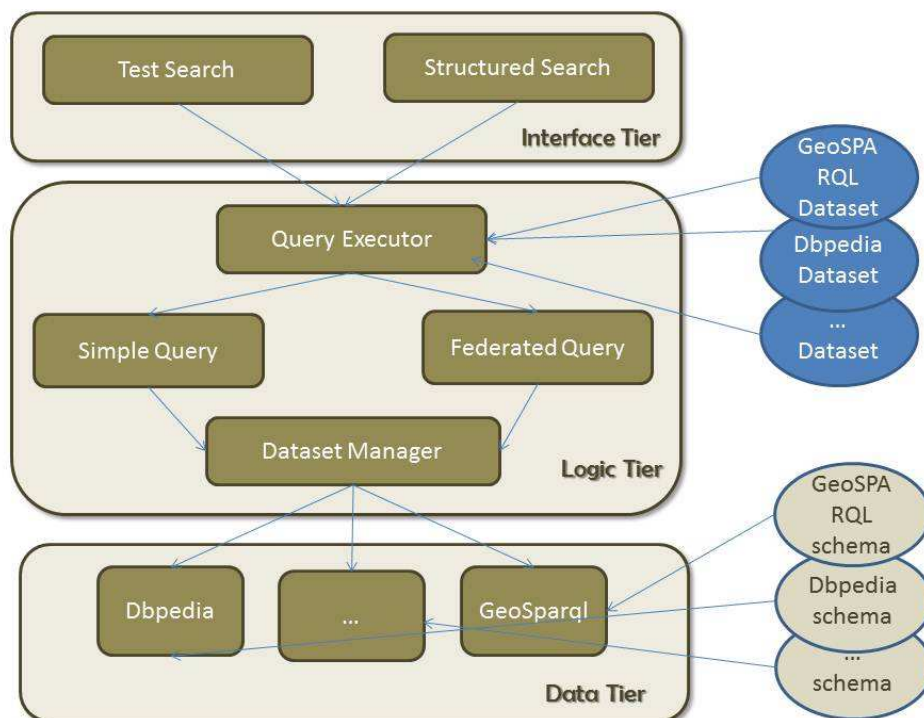


Figure 87: A high level architecture of the Linked Data Manager

### 6.2.2 Implementation with SPARQL queries

In this section, we detail the implementation of the structured queries that the LDM constructs and submits to the LD endpoints to collect geospatial entities described in Section 6.2.1. Like the tables of a relational database are queried using SQL, the triples of RDF data are queried using SPARQL. The majority of datasets in the LD cloud provide SPARQL endpoint to query and consume the exposed data. The LDM retrieves geospatial entities through these SPARQL endpoints. In the following, we outline the SPARQL queries used by the LDM to extract geospatial entities.

- **Searching by Location Name:**

DBpedia can be used as an endpoint for this purpose. In this case, the URI of the entity for the location in question, i.e. the one around which we are searching for venues, should be specified. Figure 9 shows an example of a SPARQL query constructed by the LDM to find all the venues in London. The URI of

London is specified in this query, and the type of the entities is given as (dbpedia-owl:Place).<sup>20</sup> Figure 10 shows part of the results returned by the DBpedia endpoint, which is the list of the URI of all venues in London.

Figure 11 shows another example of a SPARQL query where we restrict the type of results to restaurants only. In this case, the type of the entities is specified as (dbp-ont:Restaurant). Figure 12 shows the retrieved results.

It should be noted that in both cases, the results are not guaranteed to be located in London. These entities are only potentially located in London as the query looks for a relation of any type between London and the entity in question.

```
select ?c ?lat ?long where {
  ?c a dbpedia-owl:Place.
  ?c ?d dbpedia:London;
  geo:lat ?lat;
  geo:long ?long.
}
```

Figure 9: SPARQL query for retrieving all places in London

← → ↺ dbpedia.org/sparql?default-graph-uri=http%3A%2F%2Fdbpedia.org&query=select

c	lat	long
<a href="http://dbpedia.org/resource/Shepherd%20%80%99s_Bush_Pavilion">http://dbpedia.org/resource/Shepherd%20%80%99s_Bush_Pavilion</a>	51.504	-0.2244
<a href="http://dbpedia.org/resource/Subrata_Roy_Sahara_Stadium">http://dbpedia.org/resource/Subrata_Roy_Sahara_Stadium</a>	18.6744	73.7064
<a href="http://dbpedia.org/resource/Riverbank_Arena">http://dbpedia.org/resource/Riverbank_Arena</a>	51.5494	-0.0205556
<a href="http://dbpedia.org/resource/United_Kingdom">http://dbpedia.org/resource/United_Kingdom</a>	51.5	-0.116667
<a href="http://dbpedia.org/resource/England">http://dbpedia.org/resource/England</a>	51.5	-0.116667
<a href="http://dbpedia.org/resource/British_Antarctic_Territory">http://dbpedia.org/resource/British_Antarctic_Territory</a>	-75.0	-50.0
<a href="http://dbpedia.org/resource/United_Kingdom_of_Great_Britain_and_Ireland">http://dbpedia.org/resource/United_Kingdom_of_Great_Britain_and_Ireland</a>	51.5	-0.116667
<a href="http://dbpedia.org/resource/Croydon_Airport">http://dbpedia.org/resource/Croydon_Airport</a>	51.3564	-0.117222
<a href="http://dbpedia.org/resource/Gatwick_Airport">http://dbpedia.org/resource/Gatwick_Airport</a>	51.1481	-0.190278
<a href="http://dbpedia.org/resource/London_City_Airport">http://dbpedia.org/resource/London_City_Airport</a>	51.5053	0.0552778
<a href="http://dbpedia.org/resource/London_Southend_Airport">http://dbpedia.org/resource/London_Southend_Airport</a>	51.5714	0.695556
<a href="http://dbpedia.org/resource/Battersea_Bridge">http://dbpedia.org/resource/Battersea_Bridge</a>	51.4811	-0.1725
<a href="http://dbpedia.org/resource/Cannon_Street_Railway_Bridge">http://dbpedia.org/resource/Cannon_Street_Railway_Bridge</a>	51.5083	-0.0919444
<a href="http://dbpedia.org/resource/Craven_Cottage">http://dbpedia.org/resource/Craven_Cottage</a>	51.475	-0.221667
<a href="http://dbpedia.org/resource/Geffrye_Museum">http://dbpedia.org/resource/Geffrye_Museum</a>	51.5317	-0.07663
<a href="http://dbpedia.org/resource/Geffrye_Museum">http://dbpedia.org/resource/Geffrye_Museum</a>	51.5317	-0.0762194
<a href="http://dbpedia.org/resource/Grosvenor_Bridge">http://dbpedia.org/resource/Grosvenor_Bridge</a>	51.4847	-0.1475

Figure 10: Results retrieved; list of URIs representing all the venues in London

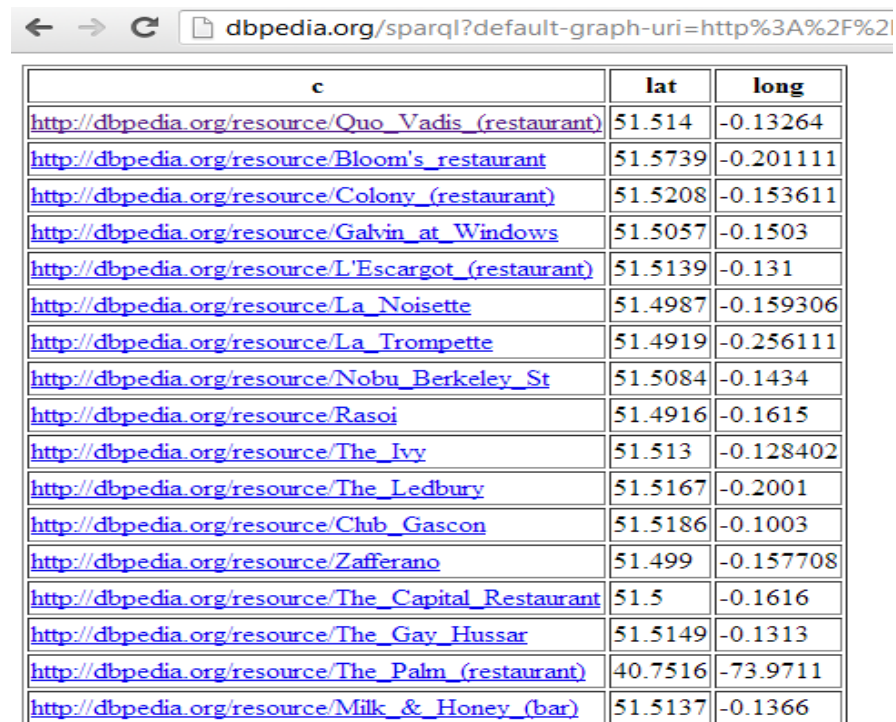
PREFIX dbp-ont: <http://dbpedia.org/ontology/>

SELECT ?c ?lat ?long where {

<sup>20</sup> dbpedia-owl corresponds to the prefix of the DBpedia ontology: <http://dbpedia.org/ontology/>.

```
?c a dbp-ont:Restaurant.
?c a dbp-ont:Place.
?c ?d dbpedia:London;
geo:lat ?lat;
geo:long ?long.
}
```

Figure 11: SPARQL query for retrieving all restaurants in London



c	lat	long
<a href="http://dbpedia.org/resource/Quo_Vadis_(restaurant)">http://dbpedia.org/resource/Quo_Vadis_(restaurant)</a>	51.514	-0.13264
<a href="http://dbpedia.org/resource/Bloom's_restaurant">http://dbpedia.org/resource/Bloom's_restaurant</a>	51.5739	-0.201111
<a href="http://dbpedia.org/resource/Colony_(restaurant)">http://dbpedia.org/resource/Colony_(restaurant)</a>	51.5208	-0.153611
<a href="http://dbpedia.org/resource/Galvin_at_Windows">http://dbpedia.org/resource/Galvin_at_Windows</a>	51.5057	-0.1503
<a href="http://dbpedia.org/resource/L'Escargot_(restaurant)">http://dbpedia.org/resource/L'Escargot_(restaurant)</a>	51.5139	-0.131
<a href="http://dbpedia.org/resource/La_Noisette">http://dbpedia.org/resource/La_Noisette</a>	51.4987	-0.159306
<a href="http://dbpedia.org/resource/La_Trompette">http://dbpedia.org/resource/La_Trompette</a>	51.4919	-0.256111
<a href="http://dbpedia.org/resource/Nobu_Berkeley_St">http://dbpedia.org/resource/Nobu_Berkeley_St</a>	51.5084	-0.1434
<a href="http://dbpedia.org/resource/Rasoi">http://dbpedia.org/resource/Rasoi</a>	51.4916	-0.1615
<a href="http://dbpedia.org/resource/The_Ivy">http://dbpedia.org/resource/The_Ivy</a>	51.513	-0.128402
<a href="http://dbpedia.org/resource/The_Ledbury">http://dbpedia.org/resource/The_Ledbury</a>	51.5167	-0.2001
<a href="http://dbpedia.org/resource/Club_Gascon">http://dbpedia.org/resource/Club_Gascon</a>	51.5186	-0.1003
<a href="http://dbpedia.org/resource/Zafferano">http://dbpedia.org/resource/Zafferano</a>	51.499	-0.157708
<a href="http://dbpedia.org/resource/The_Capital_Restaurant">http://dbpedia.org/resource/The_Capital_Restaurant</a>	51.5	-0.1616
<a href="http://dbpedia.org/resource/The_Gay_Hussar">http://dbpedia.org/resource/The_Gay_Hussar</a>	51.5149	-0.1313
<a href="http://dbpedia.org/resource/The_Palm_(restaurant)">http://dbpedia.org/resource/The_Palm_(restaurant)</a>	40.7516	-73.9711
<a href="http://dbpedia.org/resource/Milk_&amp;_Honey_(bar)">http://dbpedia.org/resource/Milk_&amp;_Honey_(bar)</a>	51.5137	-0.1366

Figure 12: Results retrieved; list of URIs representing all the restaurants in London

- **Searching by geographical coordinates:**

As discussed earlier, GeoNames is a valuable source of freely available geographical database. The position information in GeoNames is provided using standard RDF and can be queried in the normal way using SPARQL. However, without special extensions, using this data with geospatial constraints (such as searching for geospatial entities located between certain coordinates or within a certain distance from a reference point) can be computationally expensive and therefore extremely slow. For this reason, OWLIM includes special support for 2-Dimensional geospatial data. Geospatial constraints can be expressed in SPARQL queries using specialised functions, e.g. "omgeo:nearby (<lat> <long> <distance>)". Particular datasets, such as FactForge stored in OWLIM, are used for this type of queries. Therefore, the LDM uses FactForge as an end point to collect geospatial entities using exact coordinates, which offers a more elegant way than collecting such entities using the previously described DBPedia queries.

Figure 13 shows the SPARQL queries used to find all places within a circle of 0.05 mile (around 80 meters) radius around Trafalgar square in London specified by its exact coordinates (Latitude: 51.507198, Longitude: -0.127500), and Figure 14 shows the results retrieved in the form of a list of URIs of the entities representing the venues.

Figure 15 shows another example of a SPARQL query, where we restrict the results to airports and the results retrieved are shown in Figure 16.

```
PREFIX omgeo: <http://www.ontotext.com/owlim/geo#>
PREFIX fb: <http://rdf.freebase.com/ns/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX geo-ont: <http://www.geonames.org/ontology#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
select ?place ?name WHERE {
    ?place omgeo:nearby("51.507198" "-0.127500" "0.05mi") .
    {{?place rdf:type <http://www.opengis.net/gml/_Feature>. ?place
fb:type.object.type fb:location.location . ?place rdfs:label ?name} UNION
    {?place rdf:type geo-ont:Feature. ?place geo-ont:name ?name}}
}
```

**Figure 13: SPARQL query: All the places within 80 meters of Trafalgar square**

<div> FactForge RDF Search and Explore SPARQL RelFinder About Contact </div> <div>View as ExhibitDownload SI</div>	
place	name
dbpedia:Nelson's_Column	Nelson's Column@en
dbpedia:Trafalgar_Studios	Trafalgar Studios@en
dbpedia:Northumberland_House	Northumberland House@en
dbpedia:Greater_London_Urban_Area	Greater London Urban Area@en
dbpedia:London	London@en
dbpedia:London	London (England)@en
dbpedia:T_Square_200	T Square 200@en
http://sws.geonames.org/6943628/	Charing Cross
dbpedia:London	London
http://sws.geonames.org/6467287/	Trafalgar Hilton
http://sws.geonames.org/6491787/	The Grand at Trafalgar Square

**Figure 14: Results retrieved; all the places within 80 meters of Trafalgar square**

```
PREFIX co: <http://www.geonames.org/countries/#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
SELECT ?link ?name ?lat ?lon
WHERE {
    ?link gs:within(51.139725 -0.895386 51.833232 0.645447) .
    ?link gn:name ?name .
    ?link gn:featureCode gn:S.AIRP .
    ?link geo:lat ?lat .
```

```
?link geo:long ?lon
```

```
}
```

Figure 15: SPARQL query: All airports enclosed in an area

link	name	lat	lon
<http://sws.geonames.org/6296597/>	"Biggin Hill "	"51.330833"	"0.0325"
<http://sws.geonames.org/6296599/>	"London City Airport"	"51.505278"	"0.055278"
<http://sws.geonames.org/6296598/>	"London / Gatwick Airport"	"51.148056"	"-0.190278"
<http://sws.geonames.org/6296600/>	"Farnborough"	"51.275833"	"-0.776333"
<http://sws.geonames.org/6691393/>	"Terminal 5"	"51.471559907885"	"-0.487926006317139"
<http://sws.geonames.org/2647216/>	"Heathrow"	"51.4711455584879"	"-0.456490516662598"
<http://sws.geonames.org/6691396/>	"Heathrow Terminal 1"	"51.4729365901483"	"-0.450611114501953"
<http://sws.geonames.org/6301524/>	"Northolt"	"51.553"	"-0.418167"
<http://sws.geonames.org/6691397/>	"Heathrow Terminal 2"	"51.469488123267"	"-0.451812744140625"
<http://sws.geonames.org/6691395/>	"Heathrow Terminal 3"	"51.4709450654931"	"-0.457327365875244"
<http://sws.geonames.org/6691394/>	"Heathrow Terminal 4"	"51.4596759429511"	"-0.446963310241699"

Figure 16: Results retrieved; all airports enclosed in an area

### 6.3 Extracting location metadata

As discussed earlier, the LDM is capable of identifying geospatial entities from the Linked Data cloud using specialised geospatial end points. In particular, the LDM makes it possible to identify venues and historical events around the location where sensor feeds of SMART Edge Nodes are observed. The venues located within the reach of sensor feeds indicate the type of the events that can be observed via these sensors. For example, if a crowd is detected near the 'Racing' football stadium in Santander, then these crowds are most likely there to attend a football match in the 'Spanish Liga'.

Therefore, we aim to use the information about these entities in order to aid the search engine in matching the low-level events, detected by the perceptual components of the Edge Nodes, to user keyword queries. As described in Section 3, the topical component of the retrieval function aims to score how similar the user query is to the topic of events in the location and time space of the sensor observations. Information about the venues in the given location can be therefore used to obtain these scores. For example, if we know that the 'Racing football stadium' is located nearest to a SMART crowd analysis feed, then user queries such as 'liga', 'football' and 'Racing' can be matched to low level events that are detected from the crowd level feeds. This is done by mining the structured information in the Linked Data cloud found about the entity 'Racing football stadium'.

Although there is a wealth of *structured* information about the venues and places that we may extract in a given location, our aim is only to *textually* match the user queries against the description of what may actually happen in these places. Therefore, we propose an approach for extracting keyword description (metadata) about the sensor observations where we use the textual description of the entities (venues) surrounding these observations. We perform a validation study where we extract key phrases from the textual description of those entities using a shallow Natural Language Processing (NLP) technique to validate our approach and assess its suitability for the retrieval task in hand.

The automatic extraction is done as follows.

Given a location  $L$ , identified by its exact geographical coordinates, where an Edge Node is operating, we extract the geospatial entities  $(e_1, e_2, \dots, e_n)$  surrounding the location using the LDM queries on the GeoNames dataset described in Section 6.2. We restrict the search to a radius of 80 meters, a granularity which should be ok for the outdoor applications. For each entity (a venue), we use a textual description that describes the place and the activities that may occur in it. In our implementation we limit this description to the Wikipedia article describing the place. We refer to these descriptions with  $(d_1, d_2, \dots, d_n)$ . Finally, we construct a virtual document  $d_L$  that concatenates the textual description of the individual documents  $(d_1, d_2, \dots, d_n)$ . This virtual document describes the location and time space of the sensor feeds and can be used by the retrieval model to obtain the topical score.



In our validation study, we use a shallow NLP approach to extract from the constructed virtual document key phrases describing the location and the activities that may occur in it. This approach is summarised as follows. After applying part of speech tagging, we look for particular patterns, i.e. sequences of part-of-speech tags based on the algorithm for the detection of terminological terms described in [Justeson1995]. Patterns of length two and three terms form the vast majority of terminological terms according to [Justeson1995].

There are two admissible patterns of length two and five of length three as can be seen in Table 2 (where A is an adjective, P is a preposition and N is a noun). Finally, we select the most frequent nouns and noun phrases of those identified and consider them as key phrases representing the location and the activities that occur in it.

**Table 2: Part-of-speech patterns for noun phrases candidates**

Pattern	Example
A N	Shopping Centre
N N	Floor Area
A A N	Substantial New Development
A N N	High-end Retail Area
N A N	Football Top League
N N N	London Transport Museum
N P N	Set of Halls

To give the reader a concrete example about this process, let us consider the location of the centre of Covent Garden, a popular touristic shopping district in Central London. Using the exact geo-coordinates of this location (51.51197, -0.1228), the LDM uses the FactForge endpoint and the SPARQL query described in Section 6.2 to collect geospatial entities around this location. In particular, the entities collected are shown in Table 3.

**Table 3: List of entities extracted for location (Covent Garden)**

Entity	URI
London Transport Museum	<a href="http://dbpedia.org/resource/London_Transport_Museum">http://dbpedia.org/resource/London_Transport_Museum</a>
Avenue of Stars, London	<a href="http://dbpedia.org/resource/Avenue_of_Stars,_London">http://dbpedia.org/resource/Avenue_of_Stars,_London</a>
Covent Garden	<a href="http://dbpedia.org/resource/Covent_Garden">http://dbpedia.org/resource/Covent_Garden</a>
Evans Music-and-Supper Rooms	<a href="http://dbpedia.org/resource/Evans_Music-and-Supper_Rooms">http://dbpedia.org/resource/Evans_Music-and-Supper_Rooms</a>

For each entity, the URL of its Wikipedia page can be actually obtained using its property (foaf:isPrimaryTopicOf). Upon downloading the HTML page of the Wikipedia article for each entity, we extract the raw text to create the virtual document representing the location. The described procedure above for extracting key phrases is then applied on the virtual document constructed. Table 4 shows the top 20 key phrases extracted with our approach for the centre of Covent Garden. With a quick glance at the extracted phrases, we observe that some of them do actually represent Covent Garden



and the activities that occur there (Flowers market, Jubilee Market, theatre Royale etc.). However, some of them are not relevant to describe the location and match it to a user keyword query. Next, we systematically evaluate with a user study the quality of the extracted phrases using a number of popular busy locations around London.



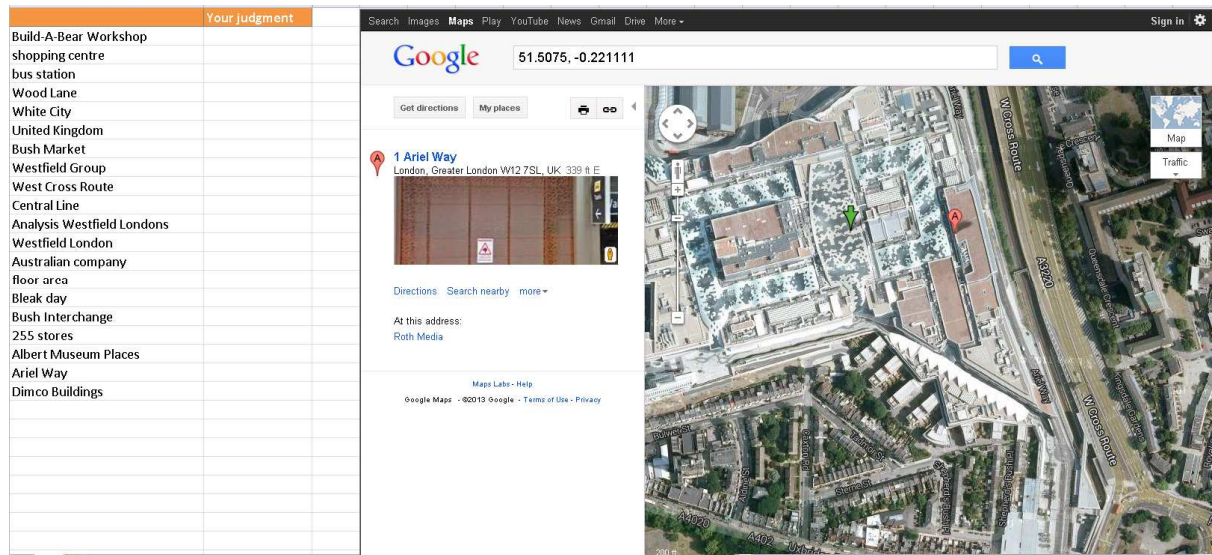
**Table 4: Top 20 phrases extracted for the location (Covent Garden)**

Flower Markets	St Martin
Royal Opera House	Frances Lincoln Publishers
London Encyclopaedia	Drury Lane
Francis Russell	Historical Research
St Paul	James Street
Long Acre	London Transport Museum
Covent Garden London	Covent Garden
Eighteenth Century	Jubilee Market
Inigo Jones	Theatre Royal
Cambridge University Press	Low Life

## 6.4 Evaluation

In this section, we validate our approach of using the textual descriptions of the geospatial entities extracted from the linked data cloud to represent a location covered by SMART sensors. In particular, we aim to assess the quality of the extracted key phrases and their suitability for their intended application, which is the matching of SMART user queries against low level events that may occur in those locations.

We performed a user study where we asked manual assessors in our lab, who can be typical users of the SMART search engine, to perform an evaluation of the quality of the extracted phrases. In this study, we selected a number of typical busy locations in London where SMART Edge Nodes can be deployed to process sensor feeds observing the physical environment in these locations. In particular, we selected 20 different locations surrounding three different types of venues: shopping streets, live music venues and football stadia. The list of locations is provided in Appendix A. For each of these locations, we ran our key phrase extraction described in Section 6.3. Three different assessors were then asked to judge the quality of the top 20 extracted key phrases for each of these locations. All the assessors were British citizens who are familiar with London. More specifically, each assessor had to fill in 20 different spreadsheets corresponding to the 20 selected locations. In each spreadsheet, a map is displayed highlighting the exact location considered together with the top 20 extracted phrases for that location in random order. Figure 17 shows an example of the spreadsheets used for assessment. Assessors were asked to judge whether each phrase describes the location and the activities that may occur in it. For each key phrase, they had to decide whether it is highly relevant, relevant or not relevant to the location and the activities that may occur in it. The exact instructions that were given to the assessors are provided in Figure 18.



**Figure 17: Example spreadsheet used by the assessors**

*In each spreadsheet, you can find a map in which a certain location in London is highlighted with a green arrow. Also, the map gives you the address of the location.*

*On the left hand side of the map, you can see some keywords that should describe the location and more importantly the activities that may occur in that location. For each keyword, we would like to judge whether they are indeed relevant to the location and the activities that may occur in it. Please put in the cell:*

*0: if you think it is not relevant*

*1: if you think it is relevant*

*2: if you think it is highly relevant*

*If you are not sure what the place is, you can google it. Note that in some cases there may be more than one venue (place).*

**Figure 18: Instructions given to assessors**

We then aggregated the assessment obtained from the three assessors. Statistics about the agreement between the three assessors are reported in Table 5. The overall percentage of agreement together with the Fleiss' Kappa [Fleiss1981] co-efficient of multi-assessor agreement are reported. We observe that overall, the agreement is acceptable as it is on par with agreement reported in relevance assessments for other NLP tasks and IR evaluation frameworks such as TREC [Soboroff2005], however this moderate level of agreement suggests that either the task is difficult (e.g. locating exactly what is around the location), or people have different understanding of what may describe a location.

**Table 5: Agreement statistics among the assessors**

Percentage of overall agreement	63.7 %
Fleiss Kappa	0.275

For each phrase, we aggregate the judgments provided by the assessors to obtain a binary decision: (i) Relevant: if at least 2 assessors judged the phrase as relevant or highly relevant; and (ii) Not relevant

otherwise.

For each location, we can then estimate the traditional precision of the key phrases at a certain rank ( $P@k$ ). The Mean Precision  $MP@20$  can be then estimated by averaging the precision obtained for each location. Figure 19 reports the results obtained for the different categories of locations considered. Overall, our approach seems to produce reasonable results as reflected in the precision obtained (45%), which suggests that using the LD cloud to extract keywords describing a location is a promising approach and can complement the manual approach described in Section 5. Moreover, the locations of live music venues obtained the highest precision, whilst the locations of sport venues had the lowest. It was noted by the assessors that people names comprise a large ratio of the phrases that they judged irrelevant which was probably amplified in the sport venues (occurrence of player names). In future implementations, we will consider a richer textual description of the geospatial entities retrieved, other than Wikipedia articles, to overcome these issues and extract relevant terms that are more specific to the activities that occur in those locations.

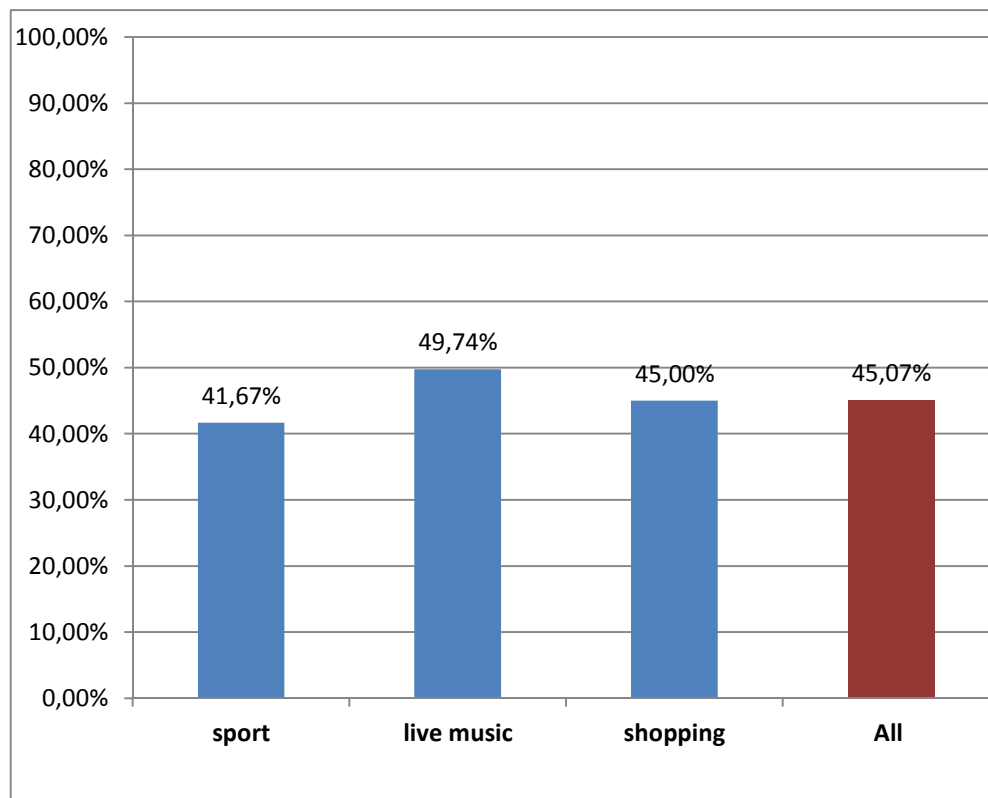


Figure 19:  $MP@20$  for different categories of locations

## 7 Conclusions

This deliverable has addressed an important aspect in the SMART Search Layer. In particular, we visited the process of understanding user information needs from the explicit keyword queries and/or implicit queries identified from the context of the user. In this deliverable, we have developed the required infrastructure at the Search Layer to make it possible to prepare user queries and effectively matched against the indexed information in SMART. Two major issues arose when it comes to matching user queries in SMART. Namely, (i) identifying keywords that describe the sensor metadata feeds at a certain location, and (ii) representing the location of the user and the feed updates within the index.

For the latter, we extended our open source Terrier platform with geospatial indices based on the geohashing function, which is capable of encoding locations in the geospatial space. We followed a modular approach to derive the data structures required for indexing and extend the processes needed during retrieval.

For extracting keyword descriptions about sensor feeds and locations, we presented two complementary approaches: (i) a manual one where we show how the Edge Node allow deployers to extensively describe their feeds, and (ii) an automatic where we use the Linked Data cloud to extract textual descriptions describing a location and the activities that may occur in it.

The deliverable contributed yet another application of the Linked Data cloud, where we exploit a number of databases, especially geospatial ones, in order to extract descriptions of an arbitrary location on earth. With our approach, we have shown, using a user validation study considering a number of locations in London, that indeed our approach produces useful key phrase describing those locations

Finally, we aim to capitalise on the geospatial indexing components we have developed in this deliverable in order to enhance our SmartReduce framework with more efficient distribution architecture for matching user implicit and explicit queries.

---

## 8 **BIBLIOGRAPHY AND REFERENCES**

[Broder2006] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. "Efficient query evaluation using a two-level retrieval process". In *Proceedings of CIKM 2006*.

[GeoHashes] <http://www.geohash.org>.

[Fleiss1981] J. L. Fleiss, B. Levin and M. C. Paik. "The measurement of interrater agreement." *Statistical methods for rates and proportions* 2 (1981): 212-236.

[Justeson1995] J. S. Justeson and S. M. Katz. "Technical terminology: some linguistic properties and an algorithm for identification in text". *Natural Language Engineering* 1.1(1995) : 9-27.

[Macdonald2006] C. Macdonald and I. Ounis. "Voting for candidates: adapting data fusion techniques for an expert search task". In *Proceedings of CIKM 2006*.

[Moffat1996] A. Moffat and J. ZOBEL. "Self-indexing inverted files for fast text retrieval". *Transactions on Information Systems (TOIS)* 14, 4, 349–379 (1996).

[SMART-D2.1] SMART FP7 consortium. Deliverable D2.1, "Detailed Report on Stakeholders Requirements", 2012.

[SMART-D2.3] SMART FP7 consortium. Deliverable D2.3 "Multimedia Search Framework Open Architecture and Technical Specifications", 2012.

[SMART D3.1] SMART FP7 consortium. "Sensors and multimedia data knowledge representation", SMART project, deliverable 3.1

[SMART D4.1] SMART FP7 consortium. "SMART Distributed Knowledge Base and Open Linked Data Mechanisms", SMART project, deliverable 4.1, 2012.

[SMART-D5.1] SMART FP7 consortium. Deliverable D5.1, "SmartReduce Engine", 2012.

[SMART-D5.3] SMART FP7 consortium. Deliverable D5.2, "Query Scoring and Anticipation Subsystem", 2013.

[Soboroff2005] I. Soboroff, and D. Harman. "Novelty detection: the TREC experience". In *Proceedings of EMNLP 2005*.

[Zubiaga2011] A. Zubiaga, D. Spina, V. Fresno, and R. Martinez. "Classifying trending topics: a typology of conversation triggers on Twitter". In *Proceedings of CIKM 2011*.

**Appendix A List of used locations for evaluation**

Location	Exact Location Coordinates
Emirates Stadium	51.554886, -0.108439
O2 London	51.501738,0.003104
Queen Elizabeth Hall	51.506667, -0.116361
Oxford circus	51.513611, -0.155556
Stamford Bridge Stadium	51.480580,-0.1911
Westfield	51.5075, -0.221111
Earls Court Exhibition Centre	51.488889,-0.197778
Royal Albert Hall	51.500944,-0.177436
Covent Garden	51.51197,-0.1228
Portobello Road	51.51425, -0.203889
ExCel London	51.5075, 0.029722
Bloomsbury Theatre, UCL	51.525278, -0.133056
White Hart Lane Stadium	51.603333, -0.065833
Boylan Ground Stadium	51.531944, 0.039444
London Bridge	51.505, -0.086
Barbican Centre	51.5202, -0.095
Carven Cottage Stadium	51.475, -0.221667
Round House	51.5432, -0.1519
Brixton Academy	51.465107, -0.114922
Wembley Stadium	51.555833, -0.279722