PlanetData

Network of Excellence

FP7 – 257641

D13.1 ParkMe Requirements and Data Source Analysis

Coordinator: Jacek Kopecký

1st Quality Reviewer: Oscar Corcho
2nd Quality Reviewer: Andreas Harth

Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	M14
Actual delivery date:	M14
Version:	1.0
Total number of pages:	18
Keywords:	linked data, geographical data, crowdsourcing, mobile

Abstract

ParkMe is a mobile and Web application that combines geographic data and parking space information with user location, social networks and other data sources in order to let its users conveniently find parking, and related value-add services, when coming to work or driving into town. ParkMe has a particular focus on gathering space availability data about car parks through crowdsourcing from the inputs of its users. In this deliverable, we describe the envisioned functionality of the application, from which we extract the requirements on the mobile app, on the Web application, and on the back-end system. We also analyze the data sources that will fuel the first prototype of the application.

Executive Summary

ParkMe is a mobile and Web application that combines geographic data and parking space information with user location, social networks and other data sources in order to let its users conveniently find parking, and related value-add services, when coming to work or driving into town.

ParkMe has a particular focus on gathering space availability data about car parks from the inputs of its users. The application will let its users add information about car parks and their up-to-date status, effectively crowdsourcing the creation of the parking data. Naturally, the application will publish the aggregate results as linked open data, to enable other third-party mashups and applications.

In this deliverable, we describe the envisioned functionality of the application, from which we extract the requirements on the mobile app, on the Web application, and on the back-end system. We also analyze the data sources that will fuel the first prototype of the application.

The list of requirements presented in this deliverable is based on our envisioned functionality of the application, and on a survey of similar mobile apps that already exist. We split the requirements into three categories: 1) requirements on the mobile app, 2) requirements on the Web application, and 3) requirements on the back-end services that will support the apps. Further, we characterize the requirements as *must-have requirements* that describe the core capabilities of the system, *nice-to-have features for incentives* that will bolster the adoption of ParkMe, and *nice-to-have features for usability* that are will make the system easier to use. The result of the ParkMe project must meet all the *must-have requirements*, and it should meet as many of the *nice-to-have features* as possible, without undue bias towards either incentives or usability because both must be well-represented for the app to have a chance of success.

Among the data sources analyzed in this document, three stand out: **LinkedGeoData** which is an RDF view on the data of the OpenStreetMap — at least initially, ParkMe will take car park locations primarily from this data source, and contribute back to it car park information submitted by our users. **SFpark** is a real-time parking availability data source for several parking garages and for some on-street parking in San Francisco, and it will be a showcase for integrating third-party parking data in ParkMe. Finally, **iServe** is a registry of services, which will be the main source of data on car park services, and on local services useful for drivers.

Document Information

IST Project Number	FP7 – 257641	Acronym	PlanetData
Full Title	PlanetData		
Project URL	http://www.planet-data.eu/		
Document URL	http://wiki.planet-data.eu/web/File:D13.1.pdf		
EU Project Officer	Leonhard Maqua		

Deliverable	Number	D13.1	Title	ParkMe Requirements and Data Source Analysis
Work Package	**Number**	WP13	**Title**	Call1: ParkMe RTD

Date of Delivery	Contractual	M14	Actual	M14
Status	version 1.0		final ☒	
Nature	Report (R) ☒ Prototype (P) ☐ Demonstrator (D) ☐ Other (O) ☐			
Dissemination Level	Public (PU) ☒ Restricted to group (RE) ☐ Restricted to programme (PP) ☐ Consortium (CO) ☐			

Authors (Partner)	Jacek Kopecký (OU)			
Responsible Author	**Name**	Jacek Kopecký	**E-mail**	j.kopecky@open.ac.uk
	Partner	OU	**Phone**	+447946721458

Abstract (for dissemination)	ParkMe is a mobile and Web application that combines geographic data and parking space information with user location, social networks and other data sources in order to let its users conveniently find parking, and related value-add services, when coming to work or driving into town. ParkMe has a particular focus on gathering space availability data about car parks through crowdsourcing from the inputs of its users. In this deliverable, we describe the envisioned functionality of the application, from which we extract the requirements on the mobile app, on the Web application, and on the back-end system. We also analyze the data sources that will fuel the first prototype of the application.
Keywords	linked data, geographical data, crowdsourcing, mobile

Version Log			
Issue Date	**Rev. No.**	**Author**	**Change**
6/11/2011	0.1	Jacek Kopecký	First version for review
21/11/2011	1.0	Jacek Kopecký	After review comments

Table of Contents

1 Introduction

ParkMe is a mobile and Web application that combines geographic data and parking space information with user location, social networks and other data sources in order to let its users conveniently find parking, and related value-add services, when coming to work or driving into town. ParkMe has a particular focus on gathering space availability data about car parks from the inputs of its users.

Managing parking in congested areas is a well-recognized problem, increasingly addressed by electronic means. In the modern car-oriented world, many will experience difficulties finding parking places after driving to work or into a congested city. It is generally expensive to build more parking spaces, and difficult to lower the number of cars that need the spaces (e.g. by supporting public transport or car-pooling). The first efforts therefore always aim to improve the efficiency of the use of existing spaces, by informing drivers about available spaces, and by guiding them to alternate car parks. In some cases this is done with manually-placed "Car Park Full" signs; in better-equipped areas there are electronic systems in place.

The figure on the right shows a typical electronic display showing the status of the car parks around a town center. The data displayed on such signs can easily be published as linked open data, which a user can then conveniently check in a mobile application or on the Web. Still, only a minority of car parks are monitored by an electronic system. With the growing popularity and affordability of internet-enabled smartphones, and with the increasing wealth of data available online, especially in the Linked Data cloud [1], we can now take a step to address the parking problem in an inexpensive and efficient manner.

Figure 1.1: A typical electronic parking sign

ParkMe will be a simple application that will help drivers locate car parks with available parking spaces. Much of the needed location information is already available, for instance in the LinkedGeoData[1] project. While there is little publicly accessible online data on the up-to-date availability status of car parks (whether a car park has available spaces or is full), a mobile application can make it effortless for users to contribute items of information (car park A is full, B isn't). In particular, our application will let its users add information about car parks and their up-to-date status, effectively crowdsourcing the creation of the parking data. Naturally, the application will publish the aggregate results as linked open data, to enable other third-party mashups and applications.

As a crowdsourcing application, ParkMe must be able to capture the interest of users, and to convince them to submit data about car parks and their availability. The core value of the application is clear (users will spend less time looking for a parking space) but it depends on the quality and quantity of the user-submitted data. The project addresses user incentives along two axes: 1) the simplicity and efficiency of the user interface, which makes it effortless for users to submit the data that they know is valuable; and 2) the added value of services and businesses related to car parks. ParkMe can integrate a number of data sources with relation to parking, such as the locations of businesses in the area, traffic and weather conditions, and even statistical information on car-related crime. By combining parking location data with business and service directories, ParkMe can for example help the users select a car park that is near a desired business or other place of interest.

In addition to such rich data sources, ParkMe will further incorporate dynamically discovered online third-party services available at or around car parks, for example for booking parking spaces or amenities such as hand car washing, or for restaurant reservations. The application can also draw on online services, e.g. the APIs of Facebook and `Revyu.com`, for social features.

To work with the multitude of heterogeneous data sources and services, and to generate and publish the crowdsourced parking data, we will use state-of-the-art lightweight Semantic Web Service (SWS) technologies that are aimed at the integration of online services and Web APIs in distributed applications. Lightweight SWS technologies are a natural fit with the kinds of data sources and services used by ParkMe.

This deliverable contains a detailed analysis of requirements (Section 2) for ParkMe, including user-based requirements and functionality requirements. Further, it identifies relevant data sources and services (Section 3) and determines their potential to be used within the application.

[1] `http://linkedgeodata.org/`

2 Requirements for ParkMe

This section analyzes the requirements of ParkMe. We start in Section 2.1 by presenting the features of the application, then in Section 2.2 we discuss some existing apps in the area, to complete our view of the features that the application should have in order to be competitive. Finally, we present all the requirements in a form of a list in Section 2.3.

2.1 User Requirements, Application Features

In this section, we detail the features of the application, in order to analyze its requirements.

Firstly, the main part of ParkMe will be a mobile app for smartphones and tablets. We have chosen the Android platform over the Apple iOS, because Android is an open platform without restrictions and approval processes for the apps. Additionally, according to market research reports such as [4], the Android platform begins to dominate the smartphone market.

Another key part of ParkMe will be the Web application that will also serve as a data store and as a back-end service for the mobile devices.

The core features of ParkMe are as follows:

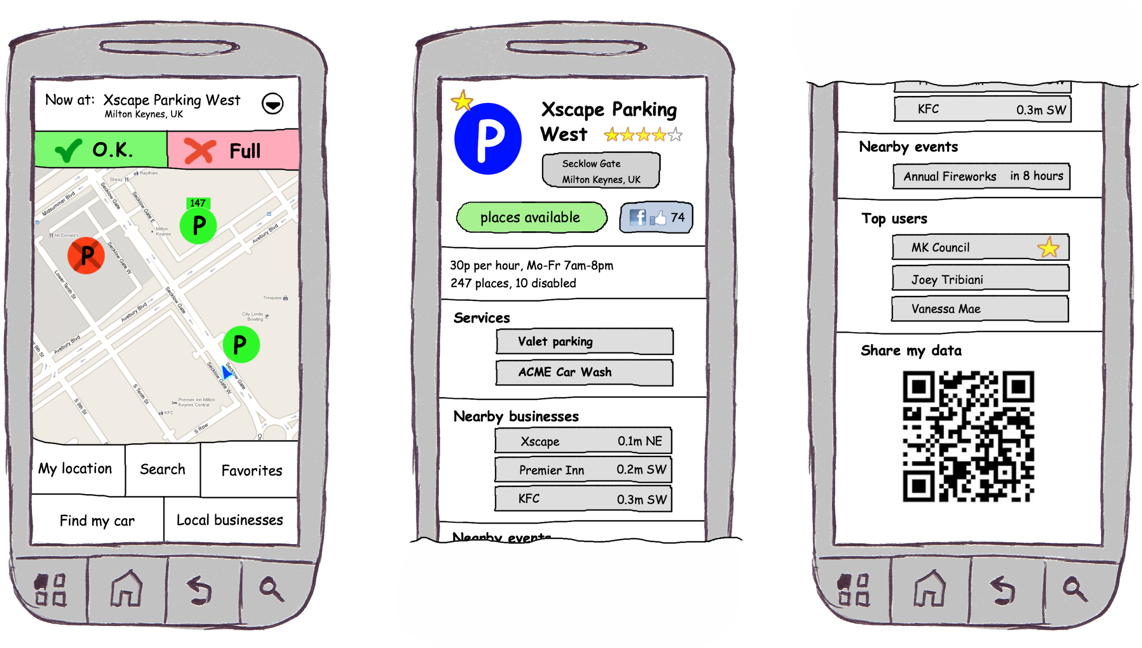
- Users finding car parks around the user's current location or in an area selected on a map, especially with information on whether there are any available spaces.
- Users submitting information about car parks, whether about the up-to-date availability of spaces, or even about the location and other characteristics of car parks previously unknown to the application.

As shown in the sketch in Figure 2.1(a), the central element of the envisioned user interface is a map, by default showing the user's current location (a blue triangle near the bottom-right of the map). The user may move the map manually, or go directly to a saved favorite area, or search for locations by address and/or key words. The map shows a number of known car parks, highlighting the current availability status. The main screen also names the car park closest to the current location of the user, allowing it to be quickly marked as full (and the user presumably drives on) or as having places available (and the user presumably parks there). If the user's location information is imprecise and the application places the user at the wrong car park, the user can select a different nearby car park; or if the system does not know any car park at the current location, the user may submit a car park description as new data.

The location where the user selected *O.K., I'm parking here* can also be used to guide the user back to the parked car, and the time when the user parked there can be combined with time limit information (e.g. 30-minute free parking in a town center, or 2-hour parking limit at a dining hub, or how much the driver has pre-paid for at the parking meter) to warn the user when the time limit is close. Finding the parked car and watching the time limits are such useful features that there are a number of apps available (cf. Section 2.2) that solely do these simple functions. As part of ParkMe, these features act as a further incentive for the users to mark the car parks they use.

When a user marks a car park as full or as available, this data is submitted to the back-end system and combined with other inputs to provide up-to-date information to other ParkMe users. The aggregate results are also published in real time as open linked data, including a stream of data updates.

The map may show further information selected by the user. The available linked data sources make it possible for ParkMe to provide a wealth of extra information relevant to parking without having to invest in building and maintaining a large proprietary database. For example, the map can display a set of chosen locations (e.g. a store, a restaurant or a business where the user wants to go) in order to facilitate finding a car park that is in a convenient position with respect to those locations. Another category of useful information is statistical government data on crime, especially when a user is driving to an unfamiliar area. Finally, up-to-date traffic information, or severe weather warnings issued by the authorities and available as linked data, can also be displayed on the map and helpful when looking for convenient car parks.



(a) The main screen with a map of car parks

(b) Detailed information about a car park, including interactive elements

Figure 2.1: A sketch of the user interface of the ParkMe mobile app

If the user does enter a desired location (or a set of locations), along with preferences on aspects such as price, distance, the existence of disabled places, or favoring/avoiding parking garages, the application can perform complex ranking to recommend the best-matching available parking options in the area.

In addition to the information displayed on the screen and to interaction through touch input, the app should also have voice capabilities. First, it should be able to inform the user through voice about relevant events such as what car park the user is approaching (with some useful details), or that the target car park where the user is headed has now filled up. Second, the app should accept voice input with commands to the effect of "I'm parking here," "this car park is full" and "where's the next/nearest available parking," followed by appropriate voice responses.

Beside the map view, the other main part of the app is an information page sketched in Figure 2.1(b). It contains an interactive information package available for each car park. The main information about a car park consists of its address (which leads to driving directions from the user's current location), the last-known availability of places (which leads to another way to submit explicitly whether the car park has places now), and details such as pricing, timing restrictions, the presence of disabled parking places or places for parents with children etc. All of this information can be edited by registered ParkMe users. As the application refreshes the car park's status, this page can alert the user if the car park becomes full as the user is driving towards it.

Using the standard format of a five-star field, the page displays the rating of the car park (through services such as `Revyu.com` and Linked User Feedback[2]). Selecting this field will allow the user to rate and review the car park, and to see the reviews of other users. Further, the single star above the ParkMe logo is a way for the user to add and remove the car park from the user's private list of favorites. Finally, the user may *like* the car park on Facebook — the application will show the number of other people who like it.

Car parks may provide services to their clients. In the sketch, we list valet parking and a hand car wash; these fields can be selected by the user to go to the online interface of the selected service. This way, a hand car wash can be pre-booked, or the user can let the valet know that the car should be made ready for departure.

[2] A service for user-generated reviews, comments and tags, `http://technologies.kmi.open.ac.uk/soa4all-studio/consumption-platform/luf/`

The application can also list a number of businesses located around a car park. Along with basic information, the businesses, too, can provide convenient online services. For example, a hotel can have a booking interface for accommodation, and a restaurant can have one for ordering take-out food. While this feature is not directly related to parking (it can be seen as a generic location-aware application such as Siri, see Section 2.2), it is easy to implement with the technologies used in ParkMe, and can act as an incentive for users to submit information about parking availability to the app.

ParkMe can also display a list of local events because they may heavily affect the parking patterns in the area. For example, an annual fireworks display in a city center is likely to fill up many car parks.

Finally, the application can also list users and other data sources that are judged as providing reliable information about this car park. As an open system, ParkMe cannot avoid erroneous and even malicious data entries, therefore users are allowed to give preference to certain data sources. For example, car park owners may register as ParkMe users and give up-to-date information about space availability; then local users can mark their data as reliable. This would allow authoritative sources to be recognized as such without any approval process. Additionally, using the social network of a user, the app can favor data submitted by the user's friends, or data explicitly marked by a friend as reliable.

Practically, every user of ParkMe who marks car parks as full or available is a data source. In the sketch in Figure 2.1, we show a QR Code[3] that represents a URI of the user's data source for easy sharing. Car park owners, mentioned earlier, may display these QR Codes around their car parks to point ParkMe users to their authoritative data source; printed QR Codes serve here as hyperlinks from the real world into the virtual world of data sources and services.

Easy data sharing fosters the formation of trust networks which are robust against data attacks such as advertising spam, so the application does not need to have a mechanism for the registration (and verification) of authoritative data sources. In effect, as the providers of the ParkMe system, we do not have to build and maintain explicit relationships with vast numbers of local car park owners to provide reliable data. Instead, the authorities are established through a combination of features in the system (data source recommendations, social network data) and in the real world (such as the cheap-to-print official QR Codes on parking meters or car park gates).

The ParkMe Web application will primarily allow registered users to manage their data (favorite places, car parks, data sources) from the comfort of a large screen and keyboard input. Further, the Web application may have comparable front-end functionality to the mobile app sketched in Figure 2.1, especially on laptop or tablet computers with accurate location information.

In the back-end, the Web application aggregates all the inputs from its users and publishes the up-to-date information about car parks and their availability status. In addition to publishing the linked parking data, the application will also document its API for accessing and updating the data from the mobile app; using this API, it would for example be possible to integrate ParkMe with existing electronic parking systems.

2.2 Related Existing Applications

The basic functionality of finding car parks near a given location is not new. Before smartphones, GPS navigation systems included basic knowledge of parking options in their built-in maps. Local authorities, mapping providers, automobile associations and other companies build and maintain their databases of car parks. This data can often be licensed for commercial use, but the cost reflects the database building and maintenance effort.

A prime example of an application built around such licensed data is AA Parking[4] — the official parking app of the UK Automobile Association, currently sold for £1.99 in the Apple App Store. It can locate car parks and provide rich information about them, including real-time space availability for some car parks.

A related application, Parkopedia,[5] extends AA Parking with coverage outside the UK and Ireland, and allows its users to submit information about car parks, but with a manual review process for submitted informa-

[3] *Quick Response Code*, a two-dimensional barcode designed for high-speed machine recognition; QR Codes can hold arbitrary data and are often used to encode URIs, in effect serving as convenient links from the real world into the World Wide Web. The QR Code in the figure contains the example URI `http://parkme.example/user123/park456/data`

[4] `http://itunes.apple.com/gb/app/aa-parking/id365260102`

[5] `http://itunes.apple.com/us/app/parkopedia-parking/id409340361`

tion. Parkopedia also includes a direct booking interface for some car parks, which again requires an explicit partnership with the car park owners, which is hard-wired as the only type of service possible. In effect, while the app is described as "think Wikipedia... but for parking!", it is in fact a closed data silo — Parkopedia does not make the user-submitted data freely and openly available, except through the proprietary mobile app, and it does not integrate third-party data sources or services.

Another notable parking-oriented app is SFpark, built around sensors in on-street parking spaces and in parking garages in selected areas of San Francisco. The system tracks in real time when and where parking is available. The app is provided directly by the operator of the parking places, therefore the data (which is also available to third-party developers, and discussed in Section 3) provides rich and highly-accurate information about the parking spaces and their prices, and it highlights the areas with available space, but it only contains data about a limited set of locations. SFpark does not only benefit the drivers — the data gathered by this system also allows the parking operator to vary its prices based on patterns of demand.

In contrast to AA Parking and Parkopedia, ParkMe will include collaborative parking information from its users, which becomes freely available as Linked Data, without an expensive process of manual submission review by ParkMe maintainers. Especially novel is the crowdsourcing of parking space availability data from the users, which contrasts with the use of sensors in SFpark; and integration of parking data from external Linked Data sources, with the inclusion of nearby businesses and services, particularly including direct access to service invocation.

Along with such rich parking apps as Parkopedia, there are more limited apps that do not need extensive data sources. For instance, ParkingMyCar[6] and Where did I park?[7] both combine the simple "find my car" functionality with parking time reminders; there are many similar free and paid apps available for a wide range of smartphone platforms.

Beside parking-specific apps, there are also more general *personal assistant* applications such as Siri[8] that can integrate some of the parking-related functionalities. Siri has a very broad scope, being able to find and book restaurants, events, taxis and more. Such generic apps are built on partnerships with third-party data sources and services rather than on Linked Open Data; the functionality of listing nearby car parks is natural for them. On the other hand, these broad apps cannot be used to generate specific new data, such as the parking information and up-to-date space availability data that are crowdsourced by ParkMe.

Finally, the SOA4All project has recently produced an iPhone app called Real Estate Finder,[9] using "information coming from multiple resources including semantically enriched data storages such as ones coming from the UK government open data initiative." It is a rich search application that "visualizes data about houses available, train stations, bus stations, schools on one map view," to "allow users to make educated decisions about property they would like to buy or to rent." While it uses several sources of Linked Data, the actual application is a tightly-coupled system with private formats and APIs for data exchange, therefore it can be seen as a step towards applications that consume Linked Data, whereas ParkMe uses Linked Data throughout, along with Linked Services (public semantic descriptions of Web services and APIs); furthermore, ParkMe will publish new Linked Data and the associated open access API.

2.3 Summary of Extracted ParkMe Requirements

The requirements on ParkMe can be put in three categories: 1) requirements on the mobile app, 2) requirements on the Web application, and 3) requirements on the back-end services that will support the apps. Further, they can be characterized as *must-have requirements (marked as R.x.y below)* that describe the core capabilities of the system, *nice-to-have features for incentives (FI.x.y)* that will bolster the adoption of ParkMe, and *nice-to-have features for usability (FU.x.y)* that are will make the system easier to use. The result of the ParkMe project must meet all the *must-have requirements*, and it should meet as many of the *nice-to-have features* as possible,

[6] `https://market.android.com/details?id=ukzzang.android.app.parking`
[7] `https://market.android.com/details?id=es.android.carfinder.activities`
[8] `http://siri.com`
[9] `http://itunes.apple.com/gb/app/soa4all-real-estate-finder/id418132948`

without undue bias towards either incentives or usability because both must be well-represented for the app to have a chance of success.

2.3.1 Requirements on the Mobile App

Below, we summarize the requirements and nice-to-have features for the mobile app.

Requirement R.1.1 Implementation in Android: until the HTML5 effort makes it possible to provide comparative user experience on mobile devices, especially in map-based apps, native implementations are preferable; the choice of the Android platform is driven by its openness and market dominance.

Requirement R.1.2 Map-based UI: as a geolocation app, ParkMe must naturally incorporate a map as its central user interface element, able to show the user's current location; fortunately, mapping is a capability bundled by default with Android libraries.

Requirement R.1.3 Car park information in the map: the map must clearly display car parks and their availability status, with the current number of available spaces, where known. Whenever displaying availability status, the app should include the *freshness* of the estimated status.

Usability feature FU.1.1 Highlighting nearest car park: the car park nearest to the user's location, especially if the user is actually in the car park, should be highlighted in the user interface, making it possible for the user to mark it as full, or as having spaces available.

Requirement R.1.4 Interpreting user input: the user may select "I'm parking here", from which the ParkMe app can deduce that finding a spot at this car park was indeed possible, as it is unlikely the user found the last available space.

Incentive feature FI.1.1 Where did I park? as an incentive for the user to mark where they parked, the app can remember the location and lead the driver back to their car.

Incentive feature FI.1.2 Parking time reminder: similarly as above, the app can remember the time of parking; combined with time restrictions in the car park, or the user's indicated pre-paid parking meter time, ParkMe can remind the driver before their parking expires.

Incentive feature FI.1.3 Additional information in the map: the map should be able to display locations of interest to the user, for instance selected target locations (businesses, meeting venue, workplace etc.) in the vicinity of which the user wants to park. Further extra types of data to show may include traffic information, governmental statistics on crime or severe weather warnings.

Incentive feature FI.1.4 Rich information about car parks: the app should gather, display and use rich information about car parks such as the address of the entrance, the opening hours, existence of disabled spaces or spaces for parents with children, etc.

Usability feature FU.1.2 Ranking of found car parks: the app should be able to find car parks near a selected location and rank them not only by distance, but also by other criteria such as price, existence of disabled places, or favoring/avoiding parking garages.

Incentive feature FI.1.5 Ratings for car parks: the app should incorporate rating and review data about car parks, or even features such as Facebook "liking" of car parks.

Incentive feature FI.1.6 Related services: the app should discover and give access to relevant local services, such as parking reservations, valet services, and hand car washes. Further, the app can find and give access to other third-party local services, e.g. restaurant reservations and cinema bookings.

Incentive feature FI.1.7 Related data: the app can display further related local data, such as events in the area that may affect parking availability.

Usability feature FU.1.3 Voice information: the user interface should be able to say relevant information in voice, so the user gets the information even if they are not looking at the mobile device (especially as they should keep their attention on the road when driving).

Usability feature FU.1.4 Voice interaction: the user interface should be able to receive commands said by the user, so the driver need not manipulate the device to perform common ParkMe tasks while driving.

Requirement R.1.5 Users as data sources: the data submitted by a user can be shared with other users; for instance, the owner of a set of car parks can register as a ParkMe user and submit authoritative up-to-date information about their car parks, which other users can then mark as reliable.

Usability feature FU.1.5 External discoverability of user data sources: the app should support easy placement of links both on the Web and in the real world for drivers to find out about the local authoritative data sources. In the real world, links can be in the form of QR Codes with prominent and privileged placement on parking meters or at car park entrances — such links imply the authoritativeness of a data source and side-step the issues of online anonymity and trust.

Usability feature FU.1.6 Syncing of data: a user who registers with ParkMe can have their data (favorite car parks, places, data sources etc.) synced with the ParkMe server so that if they switch devices, their settings can automatically follow them.

Incentive feature FI.1.8 Unregistered use: the app should provide its functionality also to users who choose not to register a user account. One functionality can readily be exempt from this requirement is sharing of data — a user cannot share their concrete submissions about car park availability without registering an account.

2.3.2 Requirements on the Web Application

The ParkMe Web application should primarily offer data management functionality for registered ParkMe users; the requirements below are oriented towards this functionality. Optionally, the Web application may also include functionality comparable to the mobile app for map-based navigation and car park information; here, the requirements from the previous section apply.

Requirement R.2.1 Registering user accounts: a user may register with ParkMe to be able to share their car park availability information and save their favorites and settings on the server.

Usability feature FU.2.1 Data management: since for some tasks, a large screen and keyboard input are beneficial, the Web application should give users rich access to all their stored information, for review and editing.

Requirement R.2.2 Deleting user accounts: a user may choose to delete their user account together will all data that the application stored about them, especially their favorite car parks, places, data sources, and any location data that the system stores about the user. The Web application must make clear and strong guarantees about when the data is really deleted, and it must clarify what kinds of data are retained after an account is deleted — none of the retained data should be tied in any way to the user's identity.

2.3.3 Requirements on the Back-end Services

The ParkMe server must be able to process all the known car park location and availability information, relevant data sources, and user accounts. Here we summarize the requirements on the services provided by the server to the mobile and Web app.

Requirement R.3.1 Accessing and integrating external data sources: the server must be able to access and refresh data from external sources, such as location of car parks, events, business directories and so on, and to integrate all the data into a common schema used by ParkMe.

Usability feature FU.3.1 Optimizing access to real-time external data sources: if ParkMe includes real-time external data sources (especially such as from SFpark), it should optimize the refresh behavior for balance between information freshness and network communication which causes load on external servers; for instance, some online data providers are known to limit the rate of access from clients, and SFpark itself recommends not calling the service more often than once a minute. Such limitations need to be taken into account adequately.

Requirement R.3.2 Publishing static car park information: the server will publish static information about car parks, submitted by the users. It may also republish car park data gathered from external sources, with the appropriate provenance information. The data should especially be accessible through a query interface for use by the ParkMe apps.

Requirement R.3.3 Publishing car park availability information: the server will publish dynamic car park availability information, especially in the form of a query and streaming API (e.g. according to [2]) optimized for use by the ParkMe apps.

Requirement R.3.4 Publishing user data sources about car park availability: the server will publish extra user data sources (following the Linked Data principles) about car park availability, to support sharing.

Usability feature FU.3.2 Provenance and information quality assessment: with the individual data sources above, users will be able to select specific trusted data sources (for example, data from the local city council or from other car park owners) whose data will then have precedence. Such explicit trust data can then be combined with social graphs to form a network of trust that is resilient to attacks.

Requirement R.3.5 API for user data management: the server will handle user registration and their personal data.

Requirement R.3.6 API for static car park data submission: the server will be able to receive new data about car parks, whether updates of known data, or information about car parks that the system was not aware of.

Requirement R.3.7 API for car park availability data submission: this API will enable submissions about car park availability, and process the submissions into aggregate information and individual user data sources. Parking status information *decays* with time, that is, the actual estimate about the status of a particular car park depends on when and how many users have submitted data, and the server must implement appropriate decay patterns.

Requirement R.3.8 Management interface: the system must have an administration interface for general management, including (de)registering interesting external data sources such as geographical data, relevant business directories, event lists, statistical data about local areas etc.

Usability feature FU.3.3 Processing on server: the server should handle as much processing as it can on behalf of the mobile clients, to support smooth user experience and power efficiency of the mobile app.

Usability feature FU.3.4 Scalability: ParkMe will primarily deal with large geospatial data sets, along with many user indications of the availability of spaces at car parks, therefore it must be able to scale both to large data and to many concurrent transactions.

3 Data Source Analysis

ParkMe puts together a number of data sources and services: geographical data for the locations of car parks and other businesses, governmental data for information such as local crime rates and severe weather warnings, social networks for friend-of-a-friend networks, review sites for car park ratings and reviews, and service registries for online services.

As shown in Figure 3.1, most of the data sources come from the Linked Data cloud, but data can also be provided by third parties such as parking owners, and the ParkMe application gathers and generates new parking data that, in turn, enriches the Linked Data cloud. The figure also shows how the mobile application can directly access third-party data sources selected by the user, and even invoke third-party services (to book a hand car wash or to order take-out food, for example).

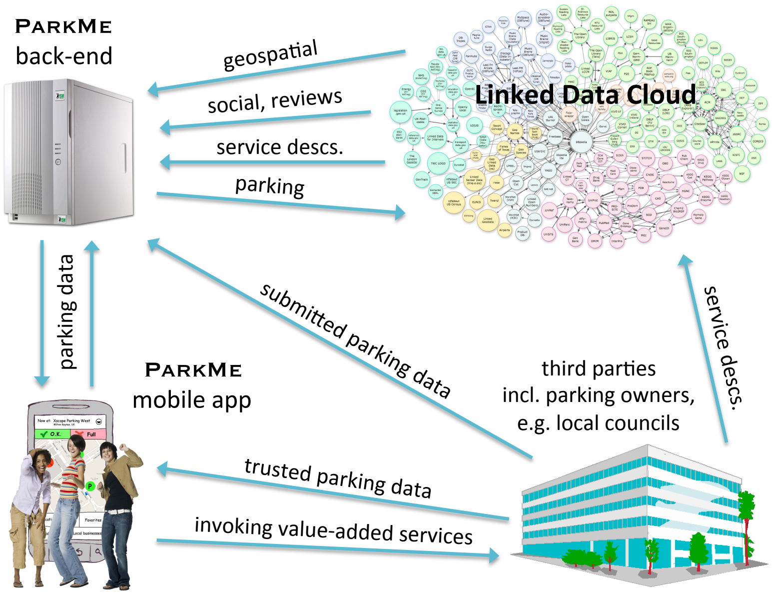


Figure 3.1: High-level data source architecture of ParkMe

The list below names the data sources that will fuel the first iteration of ParkMe development:

- Geographical:
  - LinkedGeoData (`http://linkedgeodata.org/`) contains the information collected by the OpenStreetMap project, a free collaborative map of the world that includes the locations of car parks. Considering that OpenStreetMap is the currently most prominent open source of geographical information, we can contribute back to it the information about car park locations submitted by ParkMe users.
  - GeoVocab (`http://geovocab.org/`) defines RDF vocabularies for geometry and spatial features (under the name NeoGeo [3]), and it provides two datasets, both of which can be useful for ParkMe: GADM is an RDF spatial representation of all the administrative regions in the world, and EU NUTS (Nomenclature of Territorial Units for Statistics) is a classification defined by the Eurostat office of the European Union.
  - GeoNames (`http://www.geonames.org/ontology/`) will be useful for address and area name searches;
  - DBpedia (`http://dbpedia.org/`), a data view of Wikipedia, may also contain some information about notable car parks but especially about landmarks and other points of interest.
  - SFpark (`http://sfpark.org/how-it-works/developerresources/`) provides up-to-date information on some on-street parking areas and parking garages in San Francisco; this information

is not currently available in RDF and therefore would need to be mapped appropriately. Below this list, we discuss how real-time information sources could be integrated in ParkMe.

- Governmental:
  - `data.gov.uk` provides statistical data such as local crime rates for various types of crimes, out of which car-related crime rates are especially relevant to ParkMe;
  - `metoffice.gov.uk` can provide severe weather warning data, as part of the UK National Severe Weather Warning Service.
- Social:
  - Facebook is a prime example of a service that provides social data and services; while Facebook friends lists are not part of Linked Data (partly for reasons of privacy), with the user's explicit approval they can nevertheless be imported into ParkMe.
- Reviews:
  - `Revyu.com` is a website where users can review and rate anything; the reviews and ratings are published as Linked Data;
  - Linked User Feedback[10] is a service from the SOA4All project for user-generated reviews, comments and tags, also published as Linked Data.
- Services:
  - iServe (`http://iserve.kmi.open.ac.uk`, developed in SOA4All) is a semantic service registry and a source of Linked Services information, and can provide data about services available at and near car parks; iServe also allows the ParkMe app to let the user invoke those third-party services.

The generated data will be hosted by the ParkMe Web application, but car park owners (and possibly other users) may also easily host their data in their own Web space, in the spirit of the decentralized Web and Linked Data, making it clear that it is *linked open parking data*, not proprietary ParkMe data. ParkMe can automatically integrate such data sources, discovered for example by users through QR Codes placed in the actual car parks.

Third-party data, such as that from SFpark, can only be republished within the ParkMe data set if the licence permits that. For real-time data sources, we will investigate the best strategies for incorporating up-to-date information in ParkMe user queries; this can be done partly by delegating parts of the queries to the API of the relevant real-time data sources, and partly by periodic polling and caching of the API.

In the diverse selection of data sources and services above, there is much heterogeneity. First, even RDF data from the Linked Data cloud may use overlapping but different ontologies, which may need to be aligned. Going beyond Linked Data, information may be provided in various data formats (e.g. XML, JSON, CSV) and schemas; for the purpose of the project, such data sources will be mapped into RDF.

[10] `http://technologies.kmi.open.ac.uk/soa4all-studio/consumption-platform/luf/`

4 Conclusions

A successful application like ParkMe would reduce stress, save fuel and benefit the environment, as witnessed by the motto of the SFpark application: "circle less, live more." It can further provide valuable aggregate data for traffic planning, also illustrated by the SFpark effort where parking prices vary dynamically based on patterns of actual demand.

In context of the PlanetData project, the application can demonstrate the wealth of linked data available that describes the real world, and emphasize that even when some desired data is missing (such as the authoritative and up-to-date status of all car parks), a social application can step in and provide it.

In this deliverable, we have described the envisioned functionality of the application, from which we have extracted the requirements on the mobile app, on the Web application, and on the back-end system. We have also analyzed here the data sources that will fuel the first prototype of the application.

References

[1] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *Intl Journal on Semantic Web and Information Systems (IJSWIS), Special Issue on Linked Data*, 2009.

[2] Andreas Harth and Roland Stühmer. Publishing Event Streams as Linked Data. Public draft, 2011. Available at `http://km.aifb.kit.edu/sites/lodstream/`.

[3] Juan Martín Salas, Andreas Harth, Barry Norton, Luis M. Vilches, Alexander De León, John Goodwin, Claus Stadler, Suchith Anand, and Dominic Harries. NeoGeo Vocabulary: Defining a shared RDF representation for GeoData. Public draft, May 2011. Available at `http://geovocab.org/doc/neogeo.html`.

[4] Troy Wolverton. Apple and Google have turned smartphone market on its head. *Mercury News*, 2011. Available at `http://www.mercurynews.com/business/ci_17253530`.