Extension of a Privacy-Aware Ride-Sharing App

Markus Braunbeck\textsuperscript{1}, Philipp Butz\textsuperscript{2} and Christoph Kleine\textsuperscript{3}

Abstract—Hitch-hiking is a means of transportation that is gained by asking people. Since almost everyone owns a car hitch-hiking faced a decrease of popularity. Lately, due to pollution and cost issues hitch-hiking gains population these days. As almost everybody owns a mobile device grouping up for a ride is easier than ever before. Ride-sharing is said to be the most environmental friendly motorized transport mode besides traveling by train. You can easily find lots of application for organized traveling such as "Mitfahrgelegenheit.de". Unfortunately, those applications do not support privacy issues of the owner very well.

We introduce a client server based Android application for hitch-hiking in this paper. To give the user a greater freedom we've integrated the Privacy Management Platform (PMP). vHike itself is not limited to cars only but grants the users the freedom to chose the means of transport. We further address with vHike the issue of trustworthiness of hitch-hiking by offering a permanent ride tracing and a star based rating system.

Index Terms—Ride-Sharing, Privacy, Android Permission, Privacy Management Platform, Hitch-Hiking, Android-based smartphones

1 INTRODUCTION

In times of rising energy costs and environmental awareness we are seeing a steady rise of hitch-hiking platforms. This is the place where a bunch of people can not only share a car but also share their petrol bill. The advantages are obvious: Car-sharing is petrol-saving, environmental friendly, prevents traffic jams, is often cheaper than calling for a taxi and further more.

Due to statistics the number of recorded traffic jams in Germany in 2013 was 415.000 with a total length of 830.000 kilometers \cite{1}. Almost everyone knows how grinding it is being involved in a traffic jam. After all we can all contribute in dissolving those traffic jams by sharing rides with colleagues on our way to work or by first checking if there are others, planning the same longterm trip as we do. Mobile devices are a great way to encourage potential users to use hitch-hiking platforms - a place where they can group up for a ride - more often as they allow more flexibility like short term ride planning and freedom to book it everywhere and not just in front of a computer.

Mobile devices have evolved technically over the years. Other than cellphones smart phones a shipped with high end hardware such as a strong processor or a touch screen. Thus, they are easier to handle and are capable to run sophisticated application called apps. The global smart phone market is mainly ruled by two competitors: Apples iOS and Googles Android. According to statistics over 37 Million people in Germany are using smart phones \cite{13}. On more then 60 % of the smart phones sold in Germany runs Android OS \cite{14}.

Smart phone users have access to a wide range of apps for almost every purpose. Among these one can already find many apps offering a hitch-hiking service e.g Mitfahrgelegenheit, flinc, blablacar. Sadly, lots of them following the trend to collect a vast amount of private data. Figure 1 shows the permissions requested by the mitfahrgelegenheit app. Nowadays, people are more aware of their privacy in times of the surveillances by secrets services. Hence, people care about which personal information is collected by the application and how it is used.

The goal of this work is to develop a hitch-hiking application respecting the privacy of its users. We divide hitch-hikers into two groups. The casual ones who are hicking just from time to time, using hitch-hiking as means of reaching their destination as cheap as possible. On the other hand are the ones seeing hitch-hiking as a way to meet new people and thus offer their periodical rides. With our approach we try to meet both groups. To meet this issue a hitch-hiking application has to approach the following: A clean User Interface which is easy to use, flexibility, seamless integration into everyday life, usage of state of the art technologies and options to increase the users trust in the application.

With this work we introduce vHike - a new hitch-hiking app. Our work is based on \cite{17} and \cite{16}. In our approach we followed our claim for privacy and simplicity. The users of vHike are supported in their flexibility and can selectively opt privacy interfering features such as GPS tracking out.

To address a wide range of users for our prototype we realized our app targeting Android. vHike is designed as a server - client system. vHike runs as a client on Android smart phones, from where the user can interact with a large ride database hosted on a Java server. The prototype offers beneath basic search and offer functionality some features aming to raise the users trust in hitch-hiking. Such an application requires many Permission which may have a strong deterrent effect on users. With the so called Permission System, Android regulates how applications can access private data and hardware features like GPS sensors and so.

There is no official service to opt specific Permissions out, you either grant them all by installing the app or refuse using it. To address this issue we make use of the Permission Management Platform (PMP). It works as a sublayer between the application and the Permission System. PMP empowers the user on the one hand to manually select the permissions to be used and on the other allows to pass user edited fake information to the requesting app.

The remainder of paper is structured as follows:

Related Work: In this section we give a brief overview of what is already on the market and how the competitors realize their goals. We’ll analyze their according to our own patterns and put a special focus on privacy and permissions. Privacy Management Platform: Section 4 gives a quick overview of the PMP. We describe the overall structure and how they components work together.

Concept: This Section describes the overall concept of vHike.

Implementation: This part is split into two section. On the one hand we describe how we build the server and which goals we were aiming and on the other hand we give a detailed insight how
we realized the client - our application.

2 RELATED WORK

In Android Play Store are many applications supporting hitch hiking. In this section we introduce some applications to carve out the basic functionality needed by the community.

In the Google Play Store are more than 700 K available applications. Some of them providing a platform to support hitch hiking. We wanted to pick applications, which differ in their popularity and in their provided functionalities. We choose four to have a closer look to: Mitfahrgelegenheit, Fahrgemeinschaft, Flinc, Blablacar.

First we elaborate the functionalities provided by every application, since this shows the basic functions needed by the community. To achieve this we dissect these services into basic and feature functionalities. Furthermore for our perspective a reasonable choice of suitable permissions is a key success factor. Knowing their basic and feature functionalities, we can sort the permissions into two groups:

- necessary for the basic functionalities
- necessary only for feature functionalities

Comparing these application we used the following aspects:

- Usability: With the aspect of usability we want to contribute the overall functionality of the applications. Therefore we list the provided functionalities and also the functionalities are separated into static and dynamic hitch hiking.

- Feature Functionality: The applications are not providing the same spread of functionalities. For that we distinguish the feature functionalities, which are not necessarily needed.

- Permission: The permissions needed for every application are listed and rated. For that we will discuss whether they are indispensable for the core functionalities or whether they should be able to be switched on and off individually by the user.

Mitfahrgelegenheit [10]: This is the official application for the online platform Mitfahrgelegenheit.de. It is the German adaption of carpooling.com which provides hitch hiking not only in Germany but in Europe. It enrolls a community of 5 Million users.

Everyone using this application creates a profile, which information such as age, whether the person is a smoker or not, optionally a picture of the person. The search functionality depends not only on a specific city, more easily every user can set a range within he wants to join or get off a ride. The position can be directly determined by GPS. Also this application offers hitch hiking by trains as an option in their search input routine, which can be seen in the Figure 2. Additionally it offers rides only for women, which makes the application more attractive for women as it provides more security. To improve the overall community well being, this application provides a rating system to evaluate the drivers. To offer rides, drivers have first of all to add their banking account to their profile, since the application is based on a booking system. For that every driver have to commit their sensible banking data. This data is needed, because Mitfahrgelegenheit is offering their service for a fee. The booking system makes the system more secure and creates confidence in traveling with strangers. However, it supplies a negative aspect since the drivers have to pay a percentage fee for every fellow passenger. In addition it offers a chat system, so that the driver can directly communicate with the fellow passengers, avoiding the indirect way by exchanging numbers with each other.

The basic functionalities offered by this application are the profile, the rating and the search functionality. Since offering is associated with committing sensible data to their servers, this functionality can not be clearly assigned to basic functionalities.
The feature functionalities are the offering of hitch hiking by train, the booking system and the chat system. The key factor of hitch hiking is car sharing, for that hitch hiking by train is assigned to the feature functionalities. Also the booking system can be seen as a feature function, since this goes with transferring sensible data. To widen the functionalities a chat system is a good choice to improve the community well being, but it is not necessarily needed as a basic functionality.

Knowing the subdivision of the functionalities in basic and feature functionalities, the permissions needed by this applications can be discussed. These are shown in Figure 1 and will for this application be discussed in detail. The meaning of permissions are listed at the developer website of android [9].

The first permission is needed to store data on the phone, so that they are not downloaded every time the application is started. With this permission this applications is able to read, modify and delete every content on the storage of the device.

The permission for the GPS service is needed by the input routine for the position. Since users are able to enter their position manually, this could be a optional permission. Also, because the GPS positioning contains sensible personal data.

The third permission in Figure 1 allows the application to find out, which other applications are currently or recently running. This is a inappropriate intrusion in the privacy of every user.

The permission to directly call phone numbers and makes it possible for the application to call numbers in the background. Since every applications can launch the phone screen and submit a number, so that the user have only to manually press the call or send button. Considering this, the permission can set as an optional permission or may completely be removed.

The permission group network communication is needed for sharing information with the server. For the basic functionalities like searching and offering rides, it is needed and therefore not able to be an optional permission.

The system tools permission allows the application to check whether the sd card is available to write files on it. Therefore, this permission is not affecting the privacy and can be used if necessary.

The vibrating functionality and preventing the phone from sleeping seems not to affect the privacy and can be used arbitrary. This permission is probably needed for the chat system, which is a feature functionality.

The permission run at start allows the application to start with android. Therefore it is probably needed to connect to the server on start up to receive requests by fellow passenger and to start the chat system.

The last permission allows to find accounts on the device and to read Google service configuration. First of all this permission allows the application to find every account of the user. The Google services providing Google Play Services like Google Cloud Messaging. This could be needed for the chat system.

This applications provides not only the basic functionalities but also feature functionalities. However there are permissions, which are not necessarily needed or even injure the users privacy.

**fahrgemeinschaft.de** [5]: Is a German community with more than 140 K users. It is based on the idea to provide free ride sharing service to its users.

**Flinic** [7]: The Flinc service offers hitch hiking since 2011. Their community size is specified with 200 K users.

It not only provides shared rides but also dynamic hitch hiking, which means that it is possible to hop on and off on parts of the drivers route. The users can enter their start and destination position into the search routine, the applications starts searching for an appropriate ride. The user can send a request by pressing one button, which have to be accepted by the driver. The service provides dynamic hitch hiking, therefore the application is searching for rides, that contain the start and destination position of the possible fellow passenger. This functionality is shown in Figure 4. Drivers therefore select their start and destination position, but furthermore intermediate positions, so that the applications knows the route. This is necessary for the dynamic hitch hiking service. The community is
one of the main aspects of their application. Their concept is based on trust, since every ride might be with complete strangers. The profile functionality provides personal information, the rating and also a profile picture. Users can add friends and with that build a community. Passengers can see whether someone they know has already driven with a possible driver. This provides confidence in using their application and builds an interconnection between users.

The basic functionalities offered by this application is the dynamic search and offer functionality.

For a feature functionality, they provide services to build social connections between users. Along with the rating system the application creates trust and therefore improves the confidence in sharing rides.

The applications need many permissions to provide their services. These permissions are nearly the same needed by the application Mitfahrgelegenheit. Therefore the permission to use GPS and network communication, to access to Storage, to read account information and to test the access to the storage that is required. Furthermore Flinc needs the permission to prevent the device from sleeping and the control of the vibration functionality. The privacy invasion caused by this permissions were discussed at the subsection Mitfahrgelegenheit. Moreover, the application needs the permission to take pictures and videos. This is needed to add a picture to the profile. Since users could add a picture out of the pictures on the device, this permission should also be an optional.

Flinc provides a more dynamic and community based hitch hiking service than Mitfahrgelegenheit or Fahrgemeinschaft. However, it needs permissions, which injure the users privacy.

Blablacar [3]: The 2006 founded company manages rides not only in Germany but in about ten European countries. Thereby they provide a overarching community network in Europe. With their community of over 6 million people, they are the key player on the ride sharer market together with Mitfahrgelegenheit.

This application provides detailed profiles of every user with information like the type of the car, some personal interest and the rating of other users. Additionally every user can add a picture and rate how communicative they are. Every ride offer starts with entering the start and destination position as well as the date and the time. Furthermore, driver can select options, that will be shown with figures in the final offer. These options show the main interest of the driver, whether he/she likes music or conversation while driving or whether he/she allows smoking in her/his car or not. The results of a search are listed according the closest to the search request, which can be seen in Figure 5. Afterwards every ride can be rated, what creates a trustworthy community.

The basic functionalities provide by this application are the search, offer and profile functionality.

The rating system is a feature functionality to provide more confidence in sharing rides with strangers.

This application needs the same permissions like Mitfahrgelegenheit, which were discussed in the subsection of Mitfahrgelegenheit. So that Blablacar containing the same privacy damaging permissions. Also these two applications are market leader measured at the size of the community.

The mentioned aspects are summarized in Table 1. Analyzing these applications we identified the needs of the community.

### 3 REQUIREMENT CATALOG

By analyzing the derived applications we created a requirement catalog for a hitch hiking application. Therefore we defined functional and none functional requirements. At first we need to define the basis functionality for such a hitch hiking application. The search functionality should provide various opportunities letting the users define their desirable ride. For that it should basically allow to not only enter the date and time. Furthermore the passengers should be able to define time tolerance by choosing a time slot instead of a specific time. More options improve the usability, therefore preferences should be selectable. Considering that a search could be defined particularly by letting the users state whether they want to hitchhike with smokers. In addition for some personal advantages it would be an asset to provide the opportunity to choose the option for a women only ride.

The create functionality: As counterpart of the search functionality, the create functionality have to support the same aspects. For that a driver should be able to modulate the place and time he wants to offer a ride. Furthermore, the discussed restrictions must be able to choose. These are the women only ride and to offer a ride where it is possible to smoke in the car or at least to plan stops for smoking.

The profile: The profile functionality is mainly to improve the community aspect. Everyone giving or joining a ride anticipates to see who the fellow passenger or the driver is. To take this as a basis the profile should at least supply the name, birthday or age and the
<table>
<thead>
<tr>
<th>Permissions</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>Mitfahrgelegenheit</td>
</tr>
<tr>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Network-based</td>
<td>X</td>
</tr>
<tr>
<td>Network Communication</td>
<td>X</td>
</tr>
<tr>
<td>Storage</td>
<td>modify or delete the contents of the USB storage</td>
</tr>
<tr>
<td>Phone Calls</td>
<td>directly call phone numbers</td>
</tr>
<tr>
<td>Affects Battery</td>
<td>prevent device from sleeping</td>
</tr>
<tr>
<td>Accounts</td>
<td>find accounts on the device</td>
</tr>
<tr>
<td>Camera</td>
<td>take pictures and videos</td>
</tr>
<tr>
<td>System tools</td>
<td>Test access to protected Storage</td>
</tr>
<tr>
<td>Features</td>
<td>Mitfahrgelegenheit</td>
</tr>
<tr>
<td></td>
<td>• chat system</td>
</tr>
<tr>
<td></td>
<td>• booking system</td>
</tr>
<tr>
<td></td>
<td>• rating system</td>
</tr>
<tr>
<td></td>
<td>• rating system</td>
</tr>
<tr>
<td>Type of hitch hiking</td>
<td>static rides</td>
</tr>
<tr>
<td>Costs</td>
<td>free to use</td>
</tr>
<tr>
<td>Community Size</td>
<td>5 million</td>
</tr>
</tbody>
</table>

Table 1: Overview of the analyzed related applications

Rating system: The rating system is another community building aspect. This system allows the fellow passengers to see whether the driver of an offered ride is a safe and trustworthy driver. For that reason a rating system improves the safety of hitch hiking.

Security: The analyzed applications verifying their users by sending confirmation sms or emails. Noted that most rides are with strangers, there should be a system to observe the rides. As a prevention of crime acts a safety system could inhibit crimes or support the crime clarification. As a first step a safety button could make the access to an emergency call easier. Not only by directly dialing the nearest police station, but by sending geo-location and a predefined emergency message.

A second step could be the automatically sending of geo-location and timestamps during the rides. This locations can be used to reconstruct rides in cases of crime for a more easy crime investigation. With that we discussed the functional aspects our application should be able to handle. The second part is the security aspect. Hereinafter the none functional aspects are listed and described.

Comfort: None functional usability aspects should be minded as well. The application should provide comfort in using the offered functionalities. Therefore the application should satisfy simple usability aspects: intuitively and easy to handle functionalities, automatic complete of entered addresses, easy selection of date and time. Furthermore users should not be forced to enter telephone numbers manually. They should be able to call the members of the ride directly out of the application.

Safety: Therefore, the user should be able to have the full control over the used permissions even to choose which he wants to approve to the application. This leads to better safety by protecting the phone from damage by applications.

For Sending sensible personal data such as geo-locations and profiles for every person, a secured communication is needed. To provide a platform for hitch hiking, many sensible data have to be exchanged between the server and the users. Therefore the Android Permissions System is not sufficient to provide privacy. The PMP as an alternative and more dynamic system satisfies the requirements and will be introduced in the following section.

4 Privacy Management Platform

The PMP is a new Privacy Manager for Android. It empowers the user to cloak private data and allows him or her to change permission access at runtime. With the PMP installed third party application with no PMP integration can still run on the device with no further restrictions. The PMP is build up of six main components shown in Figure 7. Namely Resources, Privacy Settings, PMP-Compatible Apps, Service Features, PMP Management and Presets [18].

We will describe those components and their interaction in the following.

The PMP encapsulates the device’s function in so-called Resources each must be registered to the PMP. Resources with similar functionalities can be grouped in Resources Groups. Resource
Groups are designed to be generic which means that they can be added or updated whenever needed. Each of these groups define their very own access policies called Privacy Settings. These can be viewed as an equivalent to the Android Permissions. However, there are two major differences. On the one hand: Privacy Settings are unlike Android Permissions not linked to the application but to a Resource Group. And on the other hand unlike the Android Permissions, privacy settings can be adjusted very fine-granulated just be granted or denied. With this the user has much more control over his or her data and apps.

Service Features have no equivalent in Android Terms nor is there anything like that implemented. Service Features determine which app feature requires access to which resource and how the privacy settings are set.

Each function can be summed up to Privacy Policy. Privacy Policies contain Service Feature, a name and a short description. A Privacy Policies not only defines how those can be accessed but can also be associated with temporal and local conditions. Privacy Policies can also determine how the requested data are return back to the application. Currently there are three different modes:

- Normal
- Mock
- Cloak

Normal mode works as expected, it returns the requested data unmodified back to the requesting application. With Mock and Cloak Mode the user gets the possibility to return modified data to the request. The difference here is whether the app perceives these data as modified or not.

The PMP Management runs the PMP in the background and displays the UI to the users.

With Presets the user has the ability to import and export his or her setting to share it afterwards with others.

In the following we will discuss PMP from a users point of view:

Aiming at a wide range of people PMP offers two execution modes: Simple Mode provides basic functionality for the unexperienced user. They only have do deal with the resources they want to grant or not.

Expert Mode is suitable for the advanced users. Its not recommended to use this mode unless the user is in charge of his actions.

5 Concept

The idea behind vHike is to merge long term planned rides with spontaneous organization of rides into one. vHike is based on the client server concept with an integration of the PMP. Thereby, it
consists of two components a server which implements all the back end functionality and runs time and performance consuming computations. On the other hand is the client which makes data accessible for the users in a nice and intuitive way. The concept of vHike is described in the following.

Since it’s a tough decision to make if you like to get into a car with a completely stranger, as you don’t know anything about his driving skills nor about his reliability, we added some trust raising features.

Following our claim for security each user is mapped to a unique profile. To be able to access the service of vHike s/he first needs to login with his or her profile into the system. After a ride each participant gets the opportunity to give his/her rating of the driver based on the executed ride. These ratings are then mapped to the drivers profile.

With a tracking service we want to maximize security for both driver and hikers. The idea behind the tracking service is to log the position of the current ride in discrete time intervals. On the one hand to prevent crime and on the other to support the investigation in case of one.

The user can chose whether s/he wants to host his or her profile and ride specific data on the official vHike servers where everyone can get his or her ride offers as a search result or if s/he wants to run his or her own personal server. These personal server can then only be accessed by initiates such as friends or family.

Initially a user selects whether s/he wants to place a offer or search for a ride. Once a appropriate ride is found the user contacts the driver and asks for acceptance and further information (like stopovers or where to pickup) about the ride. If they come up with an agreement the driver can then add the selected hikers to his ride offer. Drivers can chose the way how they can be contacted by anyone interested. You either make your phone number accessible for direct calls or make use of the chat system.

vHikes supports the user in different ways in the input process. Some of this requires services offered by the Google Play Api. Since some of these are limited in the amount of query’s we decided to offer two version of vHike. A free version with limited access to those features (1000 per day) and a payed one with unlimited access.

In the following we will outline how we implemented the concepts described in this section.

6 IMPLEMENTATION

Since every user is linked to his or her unique profile the first launch of vHike show a set up dialog for the users profile which is stored on the server. The profile is mapped to the unique phone number and gives thereby the opportunity to switch the smart phone with no effort. The user can inspect his or her profile with a read-only access by clicking profile in the Navigation Drawer Menu shown in Figure 6c. Due to security reasons we removed the ability to change the profile once it has been set-up.

Initially a user selects whether s/he wants to place a offer or search for a ride. Following our claim for simplicity we merged the set up and search filter dialog into one shown in Figure 6a. With this we gave the user the ability to set up the basic (structured) ride information such as start and destination coordinates together with time and date. The user can decide on the fly whether s/he wants to set up a new ride offer or to look for an existing in the server database without switching to another interface. To do so the user simply has to check the appropriate radio button and some further content specific unstructured data input fields appear.

The search action specific input-fields are:

- Time offset
- Radius

To get a wide range of possible rides the users can set a time and place offset. With those offsets set the search query also returns rides with a start date/place deviation.

The action specific input-fields for placing an offer are here:

- transport mode
- comment
- free seats

We have implemented a free text field for transport mode as we wanted vHike not only to work for cars but also for Trains or Buses (public transit systems often have special offers for groups).

The comment section is the place where the driver can add some specific informations or constraints to the participants for example: Driving Skill Level, costs and further more.

The driver can specify who many seats s/he wants to offer with the ride. This is done via the free seats field. With Smokers and Women-only flags the user can further restricted the search result.

Once this is done the complete search query is transmitted to the server. As shown in Figure 8 the app communicates with the Server through the Java socket mechanism and exchanges messages via the JSON format which we choose because it is more lightweight than the verbose XML standard.

The search process works as follows:

We decided to map each destination to its GPS coordinates before sending them to the server and thus to make them filterable. To prevent typos and make it easier to geocode strings we integrated auto complete from the Google Place Api. By clicking the from or to input field a new activity with a single input field is launched. Its only purpose is to handle and geocode the input.

Another way how the user is supported in the input process is by an appropriate date and time picker. By clicking on the day/time input field the application launches a date/time picker to keep it easy and simple for the user to settle the start date.

Once a appropriate ride is found the user calls the driver.
we constructed a database 10 to manage user profiles and at least the return of the database queries is transformed to
we can filter all rides and arrange them in proper order. for our database ride filter system, we decided to integrate this
the distance between two GPS-coordinates. With that information
Filtering the rides for a given input is done in the database. For that purpose we implemented a database function that computes the distance between two GPS-coordinates. With that information we can filter all rides and arrange them in proper order.
At least the return of the database queries is transformed to JSON messages and send back to the application.

6.1 Server Implementation
To make the application smooth and simple, we wanted to outsource as much functionality as possible from application to the server. Considering that the connection to the server is encrypted and the data is stored save, gives us the possibility to guarantee the users privacy and to offer fast data processing. Thus we decided to lay the validation, storing, searching process to the server (Figure 9). Application-side validation is only met by specific input-types. Therefore, we constructed a control mechanism for the server. Thus, each message from the server will be valid and complete. If a message from the application is not valid or complete the server generates an informative error message. The server provides extra functionality: The Google Distance Matrix API [8] offers a service that provides travel distance and duration for a matrix of origins and destinations. To get additional information for our database ride filter system, we decided to integrate this service.

We constructed a database 10 to manage user profiles and ride data. The database comprises four tables: One for the profiles, one for the car passengers, one for the rides and one for GPS data.

6.2 PMP Integration
As privacy is the impulse of this work we included the PMP to grant the users of our app a greater freedom when it comes to private data. To make an app PMP-compatible one needs a few components, described in the earlier section.

We designed our app to have the following Service Features:

Internet The user choses which server vHike connects to. The official vHike servers are set as default. If the user wants to use a private server s/he just needs to change the access data. From then on vHike will automatically connects to the specified server instead of the default.

Location Apps can access the location of the phone in two ways. On the one hand Google Play Service offer a network based location which is not as precise as GPS but much faster and less energy consuming. On the other hand is the GPS based location which is very precise as it is gained by accessing GPS Satellites. By sharing his or her location the user gets some additional benefits such as ride tracking and route calculation. With location access disabled security features like constant ride tracking will be disabled. As the ride tracking can also be seen as a strong interference with the privacy the user has control whether s/he wants to use this feature or opt it out. This can be done either temporarily or permanent. The location will then neither be tracked nor committed to the server.

Phone State As each user profile is mapped to the users phone number vHike needs access to it. This can be approached by reading the Phone State.

To make the application PMP aware we used the resource groups from previous works and created a singleton object within the app to check whether or not the access for a specific request are granted.

6.3 Application Scenario for a user to look for a ride:
Our prototype is based on a Android driven smart phone with minimum 4.1 Jelly Bean, required resource groups, pmp installed, up-to-date Play Services and a running Server instance.

Step 1
Once a user came up with the decision to look for a suitable hitch- hiking offer. S/he launches the application by clicking on the vHike icon, mobile network / WiFi is taken for granted from now on! Furthermore PMP and the required resource groups must be installed and running.

Step 2
The user navigates to the search interface and fills in the structured data for the search query. After checking the hike radio button the unstructured data input fields appear. The user can decide whether or not to fill them and pushes his search query to the server via the search button.

Step 3
Browsing the results and selection of a appropriate offer if there is no fitting offer the user can post his or her own offer or come back later.

Step 4
Once an appropriate offer is found the user contacts the driver by either calling or using the chat functionality to settle some further issues s/he might has.
6.4 Application Scenario for a user to place an offer:

Step 1
If a user wants to place an offer for his or her trip whether it’s periodically or not s/he launches vHike via the icon to place an offer, mobile network / WiFi is taken for granted from now on!

Step 2
The user navigates to the search filter and fills in all data required for his or her placement. After s/he pushes the offer to the server s/he has to wait till s/he gets contacted by anyone who is interested.

Step 3
The driver negotiates the overall setups with the participants.

Step 4
After they have made up an agreement the driver adds all participants to his offer via vHike.

6.5 Activity Diagram
In figure 11 we show an activity diagram that gives an overview of the basic steps to use our hitch hiking system:

Step 1
The driver and the passengers need to create a profile.

Step 2
The driver places an offer and the passenger searches for a ride.

Step 3
After matching the driver and the passenger together, they need to communicate and discuss the ride.

Step 4
During the ride the application tracks the GPS position of the users for security reason.

Step 5
Finally the passengers and the driver rate each other.

7 Conclusion
The Android Playstore contains several applications providing a platform for hitch-hiking. Unfortunately these applications need many permissions not all of them required for functionality. Users should decide which permission to grant for such applications, even when disabling some of the permissions means lack of functionality.

With vHike we created an application to combine the power of PMP and its privacy-awareness with the feasibility of an online ride-sharing platform.

It is tough to decision whether to get into a car with a complete stranger. You do not know anything how he drives and whether he is reliable. It does not get easier in times of media wide kidnapping. To raise the users trust in this vHike application we added neat features like continuous GPS tracking and user rated profiles. We wanted to give the user the choice to opt out those features as they can also be seen to interfere with private data. With respect to GPS we wanted to create more than a mere yes or no decision, so we came up with an idea that the user enters his position manually.

Some additional functionality needs to be implemented to make the application useful. One such feature is the full implementation of the PMP resources network and GPS.

Other functionality will add new features to the application.
The current application comprises a calling function. The implementation of a chat function will allow the users further possibilities to communicate with the driver or with the other passengers. For this reason a social media integration like Facebook or Google+ could be interesting.

Up to now the application allows users to search best matches for rides depending on the start and end position. For future work this could be optimized by extending the algorithm to find subsection routes.

Another possibility to provide additional features is to integrate a navigation system for the driver. So a route can be computed depending on all sub routes and exit points.

The server accepts JSON message with parameters for periodic rides. The option to create and participate in periodic rides needs to be fully implemented in the application.

The application and the server both depends on external Google services: auto-complete for locations and route duration computation. These services have a daily limit. Thus the limitations can either be passed to the users or, alternatively, a premium service with monthly payment and unlimited access to specific features can be implemented.

Finally an emergency button, as shown in paper [17], could be implemented in the application. This will increase the security for the users. The button could force the server to check the situation by calling the driver or passengers and than take necessary actions.

REFERENCES


