Pervasive Ad hoc Location Sharing To Enhance Dynamic Group Tours

Markus Duchon, Corina Schindhelm
Siemens AG, Corporate Technology
CT T DE IT 1
Otto-Hahn-Ring 6, 80200 Munich
{markus.duchon.ext, corina.schindhelm}@siemens.com

Julian Köpke, Michael Dürr, Florian Gschwandtner
Ludwig-Maximilian-University Munich
Mobile and Distributed Systems
Oettingenstrasse 67, 80538 Munich
julian@die-informatiker.eu
{michael.duerr, florian.gschwandtner}@ifi.lmu.de

***Abstract*—Transportation and mobility are important factors for economics as well as social life. In this context, dynamic carsharing systems gain increasing importance, as they address traffic related problems, like increasing the occupancy rate, and therefore have a positive effect on reducing congestion. In this work, we propose a decentralized architecture and our approach to enhance these services as well as their accessibility with ad hoc location sharing. The system works based on ad hoc communication, various positioning technologies and the users' preferences. Finally, a discussion of the initial results of our prototypical implementation is presented.**

***Keywords*-Distributed information systems; Mobile ad hoc networks; Context-aware services; Transportation systems**

I. Introduction

During the past decade, the importance of Location-based Services (LBS) has increased immensely. The vast number of scenarios creates diverse services, such as simple location dependent advertisements, a finder of restaurants or other points of interests (PoI), multi-party buddy or child tracking systems, etc. Most of these systems apply a central or client server architecture like e.g. the TraX framework [1], where the client mainly determines its own location (GPS, WLAN RSSI, Cell-ID, etc.) and other service related information and sends it to a central server. The server is responsible for the computation process, the comparison of the user's location and his current interest as well as the distribution of the results. Additionally, mobility is an important factor for social life and business management and is a key element for prosperity, economic growth and the quality of life. As a result, transportation related applications enjoy great popularity. These services range from simple traveler information systems [2] to services for finding available parking spots [3] to dynamic carsharing systems [4].

In our work, we are particularly interested in addressing dynamic carsharing (also known as ad hoc ride sharing) as well as the sharing of group tickets for public and rail transport. Thus, the occupancy rate can be increased while the travel costs decreased. However, in the case of ticketsharing, it can become difficult and time consuming to search for a spare seat or offer one. Furthermore, it is quite challenging to actually find the location of a person who is willing to share a ride or ticket especially at crowded locations.

Imagine the following scenario: A group of three persons share a group ticket for up to five persons, leaving two seats unused. At the train station they decide to offer these seats. They use one of their smartphones to establish an ad hoc network and offer the remaining seats. Another person arrives at the station and before buying a single ticket he starts the TicketShare application and enters his destination. When both devices are in communication range of each other, the application searches for group offering that match the single person destination. If a match is discovered, the current location of the devices with the matches is displayed on each others device.

Therefore, with the current position transferred, it makes it possible to locate each other at the train station even though they have not met before.

By knowing the exact location of the person on the roadside, the discovery process of fellow passengers also enhances carsharing with the knowledge of whom to pick up.

A disaster management operation represent another field where ad hoc location sharing could be helpful by allowing requests for special tools or task force units to be broadcasted without the need for an existing infrastructure.

The remainder is structured as follows. Section II gives an overview of the related work. The main requirements and the resulting system architecture, including the communication and matching processes, are outlined in Section III. The prototypical implementation and preliminary results follow in Section IV. Section V concludes the paper.

II. Related Work

During the past years a lot of work has been published related to the provision of location information especially in the area of transportation and carsharing. Due to their vast number only a small overview on the most recent projects can be provided in the following.

Fu et al. [5] propose a conceptual framework for the dynamic ride sharing community on traffic grid. Thereby, users announce their travel demands as well as planed trips including the current location to a central server. The server allows for plenty additional functionalities and the route matching

 ISBN: 978-1-61208-162-5

also takes the current traffic flow and predictions into account to provide more reliable timing information for rendezvous points. However, the system utilizes a centralized approach which does not support the idea of our locally limited and distributed approach.

OpenRide [6] is a system developed by the Fraunhofer Institute which allows for spontaneous carsharing even if the users are already on the move. The transport request or carsharing offers and requests are sent to a central server which tries to find an appropriate combination of routes and passengers. In case fellow passengers are found the service responds in real-time. Although, the system supports dynamic shared rides even when the users are in good distance a central server is utilized which also does not correspond to our idea.

Piorkowski [7] sketches the idea of an application called SmartRide which utilizes short-range communication technologies and aims mostly at urban, opportunistic trips. Thereby, the author outlines the potential benefits achieved by carsharing in general. Whereas, the main challenges and requirements are discussed neither an implementation nor a specification can be found on either communication or the transfered information yet.

Winter et al. propose a peer-to-peer based shared ride trip planning system in [8]. The system operates in a distributed manner and locally exchanges relevant information. In [9] they evaluate different communication strategies and the results received by a simplified simulation indicate that the single-hop approach dramatically decreases the message overhead. But, by using a mid- or long range strategy better results regarding the overall travel time can be achieved. As the focus of their work is on the communication process including booking and revocation of trips it only addresses a part of a complete system.

Rudnicki et al. [10] propose a concept for local shared ride trip planning. They utilize a distributed approach whereby requesting clients periodically send a query within their communication range. Hosts which are coupled to vehicles answer incoming queries in case the requested route fits with the own one. The focus of this work is on the rendezvous problem in case several transfers need to be considered. However, the applied communication range of at least 1000 meters seems to be unrealistic, it could be shown by simulation that no better results can be achieved when it is further increased.

Banerjee et. at [11] present their approach for spatially restricted location exchange and implemented a Friendmeter application which calculates the users relative position by the received signal strength indication using several radio based technologies. The locations, distances and names are also displayed in the 2D euclidean space. Besides, the exchange of location data no service information or any local matching process is considered so far.

In summary, carsharing is said to reduce the load on the overall transportation system and provides several benefits. Besides some central approaches also decentralized gain more and more importance, but none of the presented approaches is able to cover the whole spectrum yet. In regard to privacy and according to Ghelawat et al. [12] people are mostly concerned about how their personal information is used by such a system and how it might be disclosed to others especially when a central server is involved.

III. System Overview

To improve carsharing or group tours by using local information exchange, we must consider some fundamental requirements. Subsequently, we present our architecture including the most important components of a distributed location sharing application in the area of group tours. This architecture can also be adapted to other scenarios, where certain service information as well as position data needs to be exchanged.

A. Requirements

With regard to privacy concerns, the responsibility for personal profiles, route matching, and the coordination of information exchanges should not be carried out by a central component. And the users themselves should have full control to what gets disclosed. Also, the dissemination within a spatially restricted area could decrease reservations, because the provided information can only be used at the current location and for a limited duration. In order to guarantee these aspects a decentralized service provisioning and application is essential. However, it should be noted that the secure transmission of information is not in the focus of this work, but will be addressed in the near future.

In order to provide an appropriate communication infrastructure with as less resources as possible the system should apply wireless communication technologies to satisfy the requirement of ad hoc communication. Therefore, generic mobile phones with integrated WLAN hardware enabling the establishment of a mobile ad hoc network seem to be sufficient.

To determine the position of each device, different technologies can be applied. Considering our use case, indoor positioning techniques should be regarded as e.g. subway stations are also targeted. Given the insufficient precision and accuracy, the solitary use of cellular positioning systems is no option. In outdoor environments with a direct line of sight, the position can be easily obtained by satellite systems like GPS. In general, the system should also allow for a decentralized location determination.

Furthermore, a decentralized and locally executable matching process is necessary to compare the transfered information in order to discover overlapping routes and profiles respectively. To match the combinability of routes, according information about the stops and their order is required too. Since our system should support several and concurrently different fellow passenger groups a scalable approach is vital. But due to the spontaneous and locally limited character we believe that a number of 200 simultaneous users is deemed to be adequate.

B. System Architecture

In favor for an autonomous ad hoc interaction between mobile devices we waive a central component and propose a

 ISBN: 978-1-61208-162-5

fully decentralized system architecture. Therefore, we decided to use modern smartphones which allow among others for ad hoc WLAN communication and offer certain positioning capabilities via GPS and other radio based hardware.

The proposed architecture is illustrated in Figure 1. The user interface handles the users input in terms of the destination and whether seats are offered or demanded. It also illustrates the applications output, e.g. if a match occurred. The application logic is responsible for the execution of some minor functions and the delegation of incoming tasks to the according subcomponents, namely `Positioning`, `Ad hoc Communication` and `Matching`. The latter two will be discussed in more detail. The positioning component detects

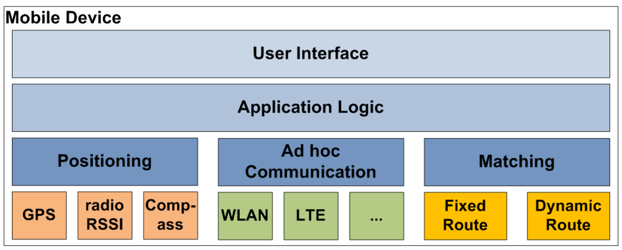
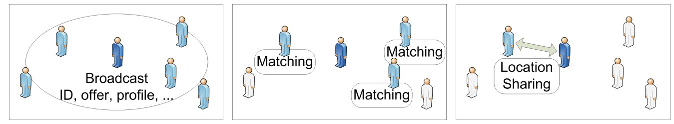


Fig. 1. System architecture for Ad hoc Location Sharing

the users exact location which can be accomplished by an integrated GPS receiver or the relative location based on the received signal strength indication using Bluetooth or WLAN, as proposed by Banerjee et al. in [11]. Also, additional sensors like a compass can be utilized as well, which in total allow for a decentralized position detection and distance measurements. The communication component has several tasks to fulfill. First, the network device needs to be configured to establish an ad hoc network functionality. Second, packets broadcasted by other clients need to be received and optionally forwarded. And third, packets which are generated by the application logic must be broadcasted. For that reason, several technologies can be applied, e.g. WLAN or LTE. The matching component compares user profiles and possible overlaps of routes. The routes can be either static like in a public transportation system or dynamic with respect to carsharing.

A simplified overview of the negotiation process is illustrated in Figure 2. In the ticketsharing scenario, a user announces a certain number of available seats, the position and orientation, the destination as well as some personal preferences within the communication range of the mobile device (Fig. 2a). Other devices and users, which receive this information locally, check if the offer fits their own needs (Fig. 2b). If so, the location and orientation of the supplier and purchaser are periodically exchanged in order to actually find each other (Fig. 2c). Especially the process of finding each other is important, so that the journey can be continued together as a group. In case that more than one seat is offered and one match has been found already, the announcement will be broadcasted again but with a decreased number of available seats. In this way we utilize the first come first serve principle.



(a) Announcement (b) Matching (c) Location Sharing

Fig. 2. Ad hoc Location Sharing Process

C. Ad hoc Communication and Protocol

To enable this kind of communication and actual data transfer using either TCP or UDP an IP address has to be assigned first. This address has to be configured by the involved devices themselves and also possible conflicts need to be resolved. A suitable protocol called APIPA [13] is utilized for this task which works as follows:

1) The client generates a random IP address within the 169.254.0.0/16 address space.
2) An ARP request for the generated address is broadcasted throughout the network.
3) If an answer is received, and the address is already in use, the process is restarted at the first step.
4) If there is no answer after a previously defined timeout of for example one second, the network adapter is configured to use the address.

For a small number of clients this protocol works good, but the full address range of $254^2 = 64516$ cannot be used efficiently, because the free addresses always have to be found by random. In our case, the chance to get a free address at the first try is still above 99% with 200 addresses in use. The communication

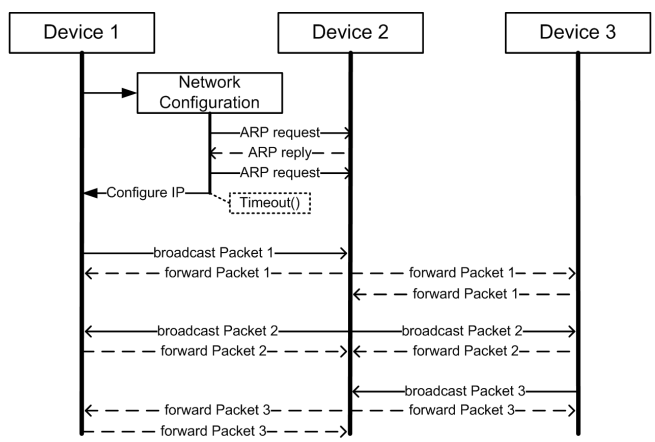


Fig. 3. Sequence diagram: network configuration and an example transmission of three packets (multihop)

between more than two devices including a multihop approach is illustrated in Figure 3. Device 2 is in the range of Device 1 and Device 3, but Device 1 is not in range of Device 3. In this example Devices 2 and 3 are already configured and the diagram starts with the configuration of Device 1 as explained earlier. By doing so, a decentralized service provisioning and application can be realized using the standard internet protocol. The service itself can be implemented by an additional protocol describing the packets to be transfered.

By utilizing the XML standard, as illustrated in Listing 1, the protocol is human readable, extensible, and new features can easily be added in future versions. To identify a packet,

```
<?xml version="1.0" encoding="utf-8"?>
<packet>
  <profile id="00:12:34:56:78:9A">
    <name session="1300738151790" counter="1">
            Julian
    </name>
    <settings>[...]</settings>
    <preferences>[...]</preferences>
  </profile>
  <confirmed>
    00:1F:3B:27:56:AB,00:AB:13:71:34:A3
  </confirmed>
  <origin>
    <station>Odeonsplatz</station>
    <position>
      <gps lat="48.150122" lon="11.581163"/>
      <heading>73</heading>
    </position>
  </origin>
  <destination line="U6">
    <station>Garching-Forschungszentrum</station>
    <adress/>
  </destination>
  <seats>
    <status>available</status>
    <count>3</count>
  </seats>
  [...]
</packet>
```

Listing 1. XML packet example

in addition to the device identifier, each packet will possess a basic counting ID which will start at zero and will be incremented with each new packet that is sent by that sender. Furthermore, a session ID is included, which consists of the time when the program was started. Each new packet will invalidate all previous packets by that device. This way, every device only has to store a set of device IDs together with their latest session ID and highest counting ID of that session to know if a packet has already been received or still has to be forwarded.

Within the confirmation section the device IDs are stored, which are matching with the offer. To prevent double reservations it can only be set by the supplier. This creates a certain matching overhead, but guarantees a proper distributed negotiation process. The first and vital payload of the protocol will be the position of the user in the origin section, which either consists of a pair of latitude and longitude when GPS is used, or the received signal strength indication (RSSI) of all devices in communication range in case of WLAN (<wlan rssi1="00:1F:3B:27:56:AB,-29dBm" rssi2="..."/>). In addition, the compass heading is included. The point of origin and the destination in form of a railway station has to be set and will be used to match the routes. Optionally, a railway line and a departure time could be selected, so different trains serving the same routes can be distinguished. Furthermore, there has to be the number of available or required seats which is vital for the proposed scenario. Other useful fields not listed are e.g. price and comment, whereby the latter one can be used to add something to the offer or request not covered yet.

The protocol also features a profile section, in which at least the user name is mandatory because it will be used to show up on other devices. The profile can be extended by certain settings and preferences like the gender, age, or habits which are used during the matching process to find suitable fellow passengers not only on the basis of the route.

D. Matching Process

First it must be guaranteed, that every user shares the same information about the network which is either a public transportation system or a general street network according to the applied use case. That information can be directly stored and handled by the mobile device as demonstrated in previous work [14]. The required information of the respective network can be easily extracted from the OpenStreetMap project [15] and preprocessed using Osmosis [16]. The map tiles provided by this project can be also utilized for the visualization and can either downloaded on demand or cached on the device beforehand.

In order to prevent double reservations the matching process first checks if the device ID is in the list of confirmed candidates and in a positive case only the name and location information is further processed. The confirmation can only be set by the supplying user to guarantee a reliable negotiation process. If not in this list, the algorithm compares the status for a supplier or a purchaser, and if the number of seats is sufficient. In case a supplier and a purchaser exchange information the process continues. Otherwise, the packet is ignored and depending on the dissemination strategy optionally forwarded. Afterwards, the profile information or preferences of the received packet are compared to the own settings. The process continues if, e.g. the age of the settings is within the interval of the desired age in the preferences, the process continues. As an alternative to the automatic comparison it would be also possible to let the users decide themselves in order to further reduce privacy concerns. By using an appropriate namespace for the settings and preferences respectively the same vocabulary is used which can be simply checked against each other. Matching and non-matching device IDs are remembered by the application and therefore are not considered by the matching process when the next packet is received.

To compare two routes, it has to be determined if both are using the same train. Thereby, and if no timetables are available, it is assumed that both take the next available train traveling to the destination. With the knowledge of the station lists which has been stored on the device the origin, destination and the railway line number can be selected by the user. The line number can also be omitted, if all stations of all lines on the route are pairwise disjoint. During the matching process it just needs to be determined if one or another destination is along the way of the other one. When a match finally occurs and is accepted by the user, the number of seats is decreased by the requested ones and the broadcast is continued. In case no seat is left the number will be zero and therefore no more match will occur, but the position data is still necessary in order to find each other.

The length of the shared route is another factor which can be accounted for. A list of matching candidates could be generated and these can be sorted by the length of the common route, whereby the seats are filled up starting with the longest one. This way, the application could not only show which clients could be picked up at all, but also can make a good suggestion for whole groups to share a ride.

In the case of carsharing the exact match of a route would lead to very few results. So, instead of the route, the detour that had to be made to deliver the person to its destination is considered. Thereby, the candidates' destination is integrated as an intermediate stop and the length of the resulting route is compared with the length of the own one. If the detour is below a certain threshold, which can be configured within the settings a match occurs. This requires several route calculations and therefore a relatively fast algorithm is required, e.g. the mobile contraction hierarchy approach proposed by Sanders et al. [17]. It calculates the length of a route in the European road network in less than 60 ms and a complete routing graph in less than 100 ms.

Depending on the use case (ticket- or carsharing) one or another matching strategy is selected, whereby both allow for a local and decentralized matching process in order to support the idea of a system which works without any central component.

IV. Implementation

We implemented the presented approach using the Android platform. To enable a spontaneous interaction between devices, we utilize the WLAN ad hoc capabilities with the location being determined by the integrated GPS receiver. For testing purposes, we decided to use the ticketsharing scenario and obtain the required train network, including the order of the stations, from the OpenStreetMap project.

A. Ad hoc Communication

By the time of implementation, it was not possible with the provided Android standard tools to initiate an ad hoc WLAN connection nor possible to utilize an arping command required for our implementation of the APIPA protocol illustrated in Listing 2. Therefore, we applied some opensource modifications to a common device in order to enable the mentioned functionalities. A broadcast with an update of the location and other data is sent every second, and as a result, the GPS location is detected once per second and shared with the matching participants. First, a random IP address is generated with the

```
private String generateIP() {
   Random rand = new Random();
   return ip = "169.254." + (rand.nextInt(254) + 1)
             + "." + (rand.nextInt(254) + 1);
}

private void checkIP() {
   try {
      su.writeBytes("busybox arping "
      + "-D -f -c 1 -w 1 -I ‘getprop wifi.interface‘ "
      + ip + "\n");
   } catch (IOException e) { e.printStackTrace(); }
}

private void setIP() {
   try {
      su.writeBytes("export brncl_lan_gw=" + ip + "\n");
      su.writeBytes(TicketShare.getInstance()
      .getFilesDir() + "/wifi config\n");
   } catch (IOException e) { e.printStackTrace(); }
}
```

Listing 2. Implementation of the APIPA example

`generateIP` method, then it is checked using the `checkIP` method and if no response is received after a timeout of 1 second, the interface is configured by the `setIP` method. To create an actual network connection without any centralized configuration, there have to be some fixed variables shared by all participating devices: (a) The SSID of the wireless network was predefined to `TicketShare`, but regarding the use case also other names can be used. (b) The wireless channel defines the frequency range on which the radio device sends and receives packets and is set to 1. (c) The UDP port is a virtual identifier for the service implemented by the protocol. Since port 31337 is not used by any known application, it was selected for the whole communication process.

To increase the coverage, we also implemented the proposed multihop broadcasting, whereby the number of hops could be limited by the time to life (TTL). Every unique packet received is sent out again by the device until the TTL value is zero. To identify a packet and to decide which one needs to be retransmitted the device ID, the session ID, and the counting ID were used. This can dramatically increase the coverage, especially when there is a high density of devices like e.g. at train stations.

B. Matching

For testing purpose, we extracted the subway network of Munich, Germany from the OpenStreetMap project and stored the resulting, ordered stations as XML files (see Listing 3) on every device, because for an according matching process all clients have to share the same network information. Otherwise, there could be inconsistencies between the matching results shown on different devices that must be avoided.

```
<?xml version="1.0" encoding="utf-8"?>
<routes>
   <stations route="U1">
      <station name="Olympia-Einkaufszentrum" />
      <station name="Georg-Brauchle-Ring" />
      <station name="Westfriedhof" />
      <station name="Gern" />
      <station name="Rotkreuzplatz" />
      [...]
      <station name="Mangfallplatz" />
   </stations>
   <stations route="U2">
      [...]
   </stations>
   [...]
</routes>
}
```

Listing 3. Example of an XML station list

First, the device ID is looked up in a list of already matching candidates. If not in the list, the algorithm checks the availability of the ticket, otherwise the name and location information is used to display the candidate on a map.

The cases where both users provide a ticket or neither offers a ticket represent a non-match. When only one of them has a ticket, the number of available seats is compared with the required number. If more or equal seats are available than needed, the algorithm continues, otherwise, they do not match.

Then the preferences of the profile are matched with the own settings and in the positive case the algorithm continues, otherwise no match was found and the device ID is stored in another list of non-matching candidates. In the implemented use case the next field being matched is the line. If this is not equal, the users will travel in different trains and therefore will not be able to share a ticket. Finally, the destinations are matched and if they are on the same route the actual destination of the supplier is irrelevant in this context, because it is assumed that the person with the longest route takes the ticket. If the route can be traveled together a match is found and the according device ID is remembered and the according information is used to display the name and location of the candidate. Otherwise no match could be found at all but the device ID is also remembered in order the matching process will not be repeated when following packets are received.

C. Discussion

We tested our implementation with four devices communicating simultaneously and equal origins heading to different destinations. The decentralized service provisioning and application worked and matching candidates were displayed including their current locations and distances of every involved device. Due to the interval of one second per broadcast the location updates were sufficient and the participants were able to find each other easily. Only the orientation information was subject to high deviations depending on the integrated hardware. The matching process for the implemented use case was also below the time of a location update and without appreciable delays. The maximum distance that could be achieved using a single hop mechanism covered about 150 meters with a direct line of sight and could be extended by the multihop broadcasting approach.

With our limited number of devices we were not able to test the maximum capacity of ad hoc networks, but by calculation and according to the analysis of Li et al. in [18] the network will be clogged with 130 devices broadcasting simultaneously when using an unlimited forwarding strategy and a packet size of around 400 bytes, which is about the size of the example packet in Listing 1.

Because of the very fast route length calculation from the work of Sanders et al. [17] also the carsharing use case seems feasible. When riding towards a group of potential fellow passengers at a speed of 30 km/h and assuming a reduced communication range of 75 meters about 150 route length calculations are possible in theory until the car passes the group which is largely enough.

In summary our approach of a decentralized ad hoc location sharing could definitely enhance group tours and their accessibility. Certainly, ulterior and comprehensive tests are necessary and will be conducted in future work.

V. Conclusion

In this paper we presented a decentralized approach to enhance alternative modes of transportation, such as ticket- and carsharing, as well as their accessibility. By utilizing ad hoc communication, a distributed matching approach, and terminal based positioning technologies, we were able to demonstrate the feasibility of our service based on commonly used smartphones. Additional improvements regarding social aspects like the ranking of fellow passengers and price negotiations are also planned. Furthermore, different position approaches need to be evaluated in more detail. Given that the focus of this work was the proof of concept for a decentralized location sharing approach, future work will concentrate on a secure authentication and transmission of necessary service information. The use of IPv6 as communication protocol instead of IPv4 could leverage the network configuration process because each device can be addressed directly. Finally, the ad hoc capabilities of LTE network and the possibility to integrate this technology into our prototype remain interesting topics.

References

[1] A. Küpper, G. Treu, and C. Linnhoff-Popien, “TraX: a device-centric middleware framework for location-based services,” *IEEE Communications Magazine*, vol. 44, no. 9, pp. 114–120, 2006.

[2] B. Ferris, K. Watkins, and A. Borning, “Location-Aware Tools for Improving Public Transit Usability,” *IEEE Pervasive Computing*, vol. 9, pp. 13–19, 2010.

[3] C. J. Rodier and S. A. Shaheen, “Transit-based smart parking: An evaluation of the San Francisco Bay area field test,” *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 2, pp. 225 – 233, 2010.

[4] J. Shao and C. Greenhalgh, “DC2S: a dynamic car sharing system,” in *Proceedings of the 2nd ACM SIGSPATIAL*, ser. LBSN ’10. New York, NY, USA: ACM, 2010, pp. 51–59.

[5] Y. Fu, Y. Fang, C. Jiang, and J. Cheng, “Dynamic Ride Sharing Community Service on Traffic Information Grid,” *Intelligent Computation Technology and Automation, International Conference on*, vol. 2, pp. 348–352, 2008.

[6] Fraunhofer FOKUS, “OpenRide,” 2009, last accessed 27. July 2011. [Online]. Available: http://www.open-ride.com/english/index.php

[7] M. Piorkowski, “Collaborative Transportation Systems,” in *WCNC, 2010 IEEE*, 2010, pp. 1–6.

[8] S. Winter, S. Nittel, A. Nural, and T. Cao, “Shared Ride Trip Planning with Geosensor Networks,” in *Institute of Geoinformatics, University of Muenster*, 2005, pp. 135–146.

[9] S. Winter and S. Nittel, “Ad hoc shared ride trip planning by mobile geosensor networks,” *International Journal of Geographical Information Science*, vol. 20, no. 8, pp. 899–916, 2006.

[10] R. Rudnicki, K.-H. Anders, and M. Sester, “Rendezvous-Problem in Local Shared-Ride Trip Planning,” International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. XXXVIII, 2008.

[11] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner, “Virtual Compass: Relative Positioning to Sense Mobile Social Interactions,” in *Pervasive Computing*. Springer Berlin / Heidelberg, 2010, vol. 6030, pp. 1–21.

[12] S. Ghelawat, K. Radke, and M. Brereton, “Interaction, privacy and profiling considerations in local mobile social software: a prototype agile ride share system,” in *Proceedings of the 22nd Conference of the Computer-Human Interaction*, ser. OZCHI ’10. New York, NY, USA: ACM, 2010, pp. 376–379.

[13] C. M. Kozierok. (2003) APIPA - Automatic Private IP Addressing. [Online]. Available: http://www.tcpipguide.com/free/t_DHCPAutoconfigurationAutomaticPrivateIPAddressingA-2.htm

[14] M. Duchon, A. Paulus, and M. Werner. Mobile Anwendung zur Routenplanung mit öffentlichen Verkehrsmitteln basierend auf OpenStreetMap-Daten. FOSSGIS 2011.

[15] S. Coast. (2004) OpenStreetMap - The Free Wiki World Map. Last accessed 27. July 2011. [Online]. Available: http://www.openstreetmap.org

[16] Osmosis. Processing OSM data. Last accessed 27. July 2011. [Online]. Available: http://wiki.openstreetmap.org/index.php/Osmosis

[17] P. Sanders, D. Schultes, and C. Vetter, “Mobile Route Planning,” in *Algorithms - ESA 2008*, ser. Lecture Notes in Computer Science, D. Halperin and K. Mehlhorn, Eds. Springer Berlin, Heidelberg, 2008, vol. 5193, pp. 732–743.

[18] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris, “Capacity of Ad Hoc Wireless Networks,” in *7th ACM International Conference on Mobile Computing and Networking*, 2001.