FAKULTÄT FÜR !NFORMATIK

Airwriting: A Platform for Private, Mobile, Spatial Group Messaging

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

Wirtschaftsinformatik

ausgeführt von

Alexander Sommer
Matrikelnummer 0047388

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung:
Betreuer/Betreuerin: Prof. Dipl. Ing. Dr. Rene Mayrhofer

Wien, 30.04.2009

(Unterschrift Verfasser/in)

(Unterschrift Betreuer/in)

Table of Contents

1 Introduction

Communication is a basic human need. Hence, it is not surprising messaging services have become very popular [3]. They promptly connect people wherever they are with moderate or at least decreasing costs. Range from rather simple [1] messaging services like textual chat to complex ones like real time, secure video chats for international business conferences.

As time goes by, human needs change. And if they do, services will change as well and often quickly; if people need to communicate anonymously, services for secure communications are offered. If people need to communicate instantly, instant messaging services occur. And if people want to share their moments online within their social community, twitter services are quickly available. In marketing language, this is called *pull* strategy [4]. The demand of customers leads to new services.

Instead of responding to customers' needs, new messaging services can be offered in advance, too. In marketing language, this is named *push* strategy. New services are leading to or stimulating new needs and opportunities, which have not been perceived before.

Both strategies have different advantages and drawbacks [5]. On the one hand, focusing on pull strategies maximizes the probability that offered services will be accepted by customers. On the other hand, focusing on push strategies maximizes the probability for inventing unique new services. All projects, which have serious attempts for gaining a certain market share, should take marketing strategies into account. For that reason, the project *airwriting*, which will be presented in this thesis, aims to apply both [2] strategies but will use the push strategy exclusively in the very first steps for several simple reasons. Firstly, programmatically applying the pull strategy after applying the push strategy is much easier than the other way around [3]. Secondly, exemplarily pushed services may explain others the new concept and potential of airwriting. And thirdly, even useless (but funny) services can gain crucial attention in the world of marketing.

In the world of business, fierce competition among (mobile) phone manufacturers and telecommunications service providers generates vast amounts of different phone models with more and more features, few but complex and rich software platforms (e.g.: J2ME, Android, Objective C), different services, and many different service contracts as well. But the original services enabling *voice messaging* and *textual messaging* didn't change significantly, although communication and the language itself (linguistic) is a very complex thing to grasp.

We believe that messaging services can be *significantly* improved, if their underlaying implementation is based upon models and concepts and demonstrate a conceptual design of a specific platform which is implementing an interpretation of a *model of attributes*. In this context, attributes of a message are all

[1] From a technical point of view.

[2] Both strategies are expected to make important contributions for the success of this project.

[3] Chapter 3 explains why.

components it is existing of, like its text, date, location, receiver, recipient or group-membership [6]. Several previous prototypes have already shown encouraging results if messaging services are enhanced with special functionalities or even with *unique attributes*. One particular attribute is the location. Spacial messaging services are unrealizable without the location per definition. And enable various services in gaming [7], meeting scenarios, and grass-roots motorized traffic control [8]. For that reason, the location attribute (and therefore spacial messaging) is discussed and demonstrated as the prime example of attribute candidates in this thesis.

This project airwriting is described in three different dissertations. The paper of Mayrhofer et al [1] presents the idea, implementation and privacy issues of this platform shortly. Thus, it gives a compact overview of all important topics concerning this project. Saral [2] covers the privacy aspect and server implementation of the airwriting architecture in depth. This master-thesis contributes the specific platform design (chapter 3) and implementation (chapter 5) for *attribute based messaging* using mobile devices in detail.

The implementation is tested (chapter 6) with a game example "Scavenger Hunt" on three different mobile device platforms; The J2ME- with the help of the J2ME Polish framework, the iPhone- and google's android platform. These client implementations will follow general design guidelines which will be presented in chapter 4. Gained insights concerning these platforms will be presented and compared to each other in table 7. A short evaluation and presentation (chapter 2) of various other location based services [9] in terms of attributes will help in classifying or even creating future services more methodologically. The tables 2 and 3 will help in giving such an attribute based view a first impression. Some attributes suitable for various applications like games, information-services and scientific exploration are listed and suggested in these tables as well. Chapter 7 will conclude this master-thesis in providing suggestions for further research and summarizing the results of the usability study.

The short-term goal of this project is demonstrating and verifying that even a simple implementation of a model of attributes is able to work successfully. The long-term goal is twofold; Firstly, this project aims to provide new services and applications or at least better ones in terms of utility and enjoyability. It should (ideally) find, suggest and predict new attributes or attribute-mixtures for new services for two reasons. Firstly, for inventing new or improved messaging services for customers. And secondly, for proposing new business models and opportunities for telecommunications service providers which are constantly seeking for new markets. As more and more message services - and specially message services for spatial messaging - will evolve; students, researchers (e.g.: program designers, linguists, psychologists) and entrepreneurs will analyze such services from different views and different levels of interest. Therefore, the second goal is establishing an additional view - an attribute based one - on textual based messaging services which may help others implementing new messaging services or analyzing existing ones for further improvements. From the point of

view of attributes, a designer should ask oneself: *"Which attributes do I need* [4]*, which attributes should I combine and which attributes have the most impact on my services?"*

[4] To receive an impression, take a look on table 2 and 3.

2 Related Work

In an earlier system, interest in spatial messaging was low in practical experiments [10], but increasing [5]. Actually, not less than 75[6] location-based, *social* platforms exist. Most of these platforms like GeoNotes [11], Plazes and JotYou[7] have similar functionalities. One can read and write location based messages and/or see (meet) friends who are nearby. Airwriting attempts to go beyond these basic applications and provide a sustainable [12] and privacy aware system (as explained in section 3). In the following, various location based projects are presented; Some of which are complex and specialized. The advantage of this specialization to application fields are functionalities tailored to particular needs. The disadvantage compared to simple messaging services is, the more applications specialize on single issues, the more they disqualify to others.

2.1 Non-Focused Location Based Messaging Services

The main purpose of non-focused location based messaging services is sending and receiving textual messages *without* explicitly focusing to any certain application or application field. Such services can be used for nearly everything where location based textual messages are useful but have also lacks due to their elementary concept.

2.1.1 E-Graffiti E-Graffiti [10] is one of the first scientific projects concerning spatial messaging, which was evaluated by undergraduate students during a semester. The results of the user survey were far from encouraging due to several reasons. One of those reasons may have been the need to use - and therefore carry all the time - laptops for reading and writing messages. Airwriting will present the feature of reusing the combination of attributes for groups to establish a more flexible design to stimulate demand in using a location based platform.

2.1.2 Geo-Notes Geo-Notes [11] is also one of the first scientific projects concerning spatial messaging. It is conceptually, similar to E-Graffiti. Notebook-Clients had to be used for sending and receiving messages. Per Persson and Petra Fagerberg [13] evaluated.

2.1.3 JotYou JotYou [8] enables reading and writing locations-based text messages. The google map service is used for the visualization of the messages online. The functionalities of the server offer creating, sending and viewing messages. Hence, the motion of a group of users can be tracked. JotYou mentions this

[5] ZYB, a social platform, was bought from vodafone for 31.5 Million Euros, www.gomonews.com/vodafone-acquires-100-of-zyb/ (2008)
[6] http://www.bdnooz.com/lbsn-location-based-social-networking-links (2009)
[7] http://www.jotyou.com/ (2009)
[8] www.jotyou.com (2009)

tracking service also for gaming purposes, but does not provide any games directly. It depends on the users, how they will use JotYou for games. Information concerning privacy or security topics is not published online. The prototype of airwriting is very similar to JotYou. Easy to use, intuitive and working.

2.1.4 Herecast Herecast is an indoor location based service, working with WIFI instead of GPS. The software is available for Windows and Pocket PC only. For each access point (location) the country, province, city, area, building, address and even the floor may be stored. Hence, an RSS feed informs in case of updates. Also Airwriting will locate users who are indoor in the future. For that reason, a WIFI technology was implemented and tested shortly in the airwriting architecture but failed in practice and therefore is not part of this diploma-thesis.

2.1.5 Locale Locale [9] is a project of five MIT students and one of the winners of the google android challenge 2008, thus it is a software for the android OS. Locale is able to change the settings (e.g ringtone to vibration or completly silent) of the mobile phone, if a specific condition occurs within a specific location (condition - location -> context). It is not planned for Airwriting to be a context-sensitive app in the near future but considered to be realized if any remarkable success occurs.

2.1.6 enkin Enkin [10] combines several sensors (camera, GPS, orientation, compass) and services for realising a live, 3D navigational concept. The user can direct the phone towards a real object for retrieving more information about it. The concept is similar to Geowand. Enkin participated in the google android challenge, but was rejected for being one of the finalists. The founders of this project had been contacted seperately from google for further cooperation reasons after the challenge was over. Airwriting is basically very similar to Enkin. Attributes are combined to realize new features. Whereas Enkin is targeting one feature (navigational concept), Airwriting tries to realize as many features as possible.

2.1.7 The point to discover: GeoWand Geowand [11] allows pointing the mobile phone towards a direction for retrieving location based information. The gps- and compass sensors are combined for realizing this functionality. Therefore, it is an orientation aware, geospatial service. Additionally, a framework exists [11] which can be used to implement their *local visibility model*, thus a model realizing the point-to-object feature.

2.1.8 Locatik Locatik [11] allows an update concerning the location also via sms; the user just types the current location as a textual description (eg. Eiffel

[9] www.androidlocale.com (2009)
[10] www.enkin.com (2009)
[11] www.locatik.com (2009)

Tower, Paris) to update it. Locatik offers various interfaces to use their services: A widget which can be embedded into other websites, a mobile web site for mobile phones which are not powerful enough to use the client software and the standard client software itself available for the iPhone and Nokia S60 series. Airwriting will possibly offer a user-input-based localization similar to Locatik in the future, but with restrictions. If a user can freely set the current position, there is no guarantee that the user is really at the current position. This problem may effect some security issues and services offered by airwriting.

2.1.9 Google Earth for mobile clients Google Earth for the mobile client is a location based service. It shows the current location on the map, calculates the time from it to the point of interest, lists nearby points of interests, displays real time traffic information and has a function named street view, a virtual 3D view of a location. Google Earth is using google map service, which is for almost all of the following services the favourite - and used - map service. It is very flexible in terms of applications fields due to its layer-structure. This layer structure is similar to groups used in airwriting and other location based platforms. Each layer shows only specific things the user is interested in.

2.2 Social Platforms

Social platforms connect people in different ways, by providing them with the ability to talk to and find each other and to share some content (pictures, movies, music, links) or attitudes. Location based social platforms extend the way of finding each other. Whereas the finding function available at a social platform is just a text-field expecting the name of some user, location based social platforms support finding people spatially. These functions are often named "friend-finders." The prototype of Airwriting doesn't offer a friend-finder functionality, but definitely will in the near future.

2.2.1 Skobbler Skobbler [12] is a location based, social platform which tries to offer different location based services at once; vehicle navigation, on-foot navigation, local area search, address management, and hotel reservations. Skobbler and airwriting are both targeting a wide range of application fields.

2.2.2 Plazes Founded in 2004, Plazes [13] tries to inform its users about friends who are currently close and wants users to tell others what they are currently doing at the specific place. Therefore, they have a "radar" function showing which people are at what location. It is embedded in the social platforms flickr, twitter users, berlin, reboot9 and also at Nokia's map portal. The google map service is used for the visualization of the messages. Notifications about actions can be sent by email and sms. Hence, tools for publishing the current location

[12] www.skobbler.com
[13] www.plazes.com (2009)

of a user are available, namely Fire Eagle, Twitter, Plazes Map Widget and Plazes Google Earth Experience. The privacy settings have three options 1. 'this is public' - messages are public and searchable by search engines like google 2. 'only your contacts' and 'only you & shared contacts'. Also a detailed description of how the collected data from the user is used exists. Plazes also is offering an API for 3rd parties who may want to use their services. Airwriting will take Plazes and its success as its role model. If the concept of attributes succeeds, airwriting should be able to establish its own community.

2.2.3 NowHere NowHere [14] is a social, location based platform similar to dimdix. Places and *favourite* places can be shared. NowHere, compared to others, charges the functionality to locate its users. A registration enforces the announcement of the personal phone number. Airwriting will be free for standard end-users. It services will pay for themselves through advertisements.

2.2.4 Loopt Loopt [15] is one of the most professionally presented, location based social platforms available. It supports a wide range of platforms, phones and even networks. It enforces the use of automatically sent sms text messages, which can lead to costs. Loopt also offers some generic privacy features like the "promise not to share user data" to third parties. Loopt can be seen as a good example for a location based social platform. Actually, it is one of the best ones. Naturally, also Airwriting is trying to stand in at the forefront of such applications but wants to find its niche instead of being just another (good) location based social platform.

2.2.5 Dimdix Dimdix [16] is a location based, social platform. The former name which is still used as its second name is androidlocation. It is available for the Symbian (S60 3rd and 2nd edition), Blackberry, Motorola and Windows Mobile (Smartphone and Pocket PC) platform but still in alpha status. Dimdix offers many device platforms. Airwriting also aims to support as many devices as possible.

2.2.6 Pocket Live Pocket Llife [17] is a location-based social platform which will be (contrary to most of the other platforms) free only in beta status. Hence, some control features for privacy concerns are available. A registration was not possible due to a never ending initialization progress bar. Pocket Life is a good bad example. Many of the location-based projects are never-ending, but the motivation of being one of the first ones forces them to go online with bugs and missing features. Airwriting will only go public online, if it can fulfill some minimum quality standards.

[14] www.nowhere.com (2009)
[15] www.loopt.com (2009)
[16] www.androidlocation.com (2009)
[17] www.pocketlife.com (2009)

2.2.7 BlueMapia BlueMapia [18] is a "'Web and Mobile Social Mapping (service) for Boaters." Because of this explict and delimiting definition, it can be seen as one of the first location based projects which has a different and even particular target group selected, rather than the others who are just aiming at the masses. They separate between points of interest (*"'a geographical area whose content is managed by the editors"'*) and place marks (*"'a geographical point whose content is managed by the user who created it"'*). BlueMapia is available for the Windows Mobile platform only. Also, a BlueMapia TV Channel is available. BlueMapia is a good example for specialization and finding niches. Airwriting (as a social platform) will never specialize in such way, but will offer an API third parties can use. There should be no problem to realize similar platforms with the help of the airwriting framework.

2.2.8 belysio Belysio [19] is a location-based social platform for free, available in the English, German and Polish. They put their focus on connecting people in all forms possible with the mobile phone. Their address book is accessing the mobile's address book and merging it with the contacts of the (registered) belysio friend-list and other IMs Services like Yahoo and MSN.

2.2.9 meetMe The application meetMe [20] is a social networking application available for the iPhone and iPod only. It is not possible to leave messages on a specific place only, but for buddies currently nearby.

2.2.10 BuddyCloud Buddycloud [21] is a social, location-based networking application. A user can share the current location status and find and talk to nearby people. There is a function to "bookmark" places, which results in showing this status to others. It is available for the symbian OS only. Hence, an API exists.

2.2.11 BuddyMob BuddyMob [22] is a location-based messaging service for the google android platform only. It supports integration with MSN, AIM, GTalk, Jabber and Twitter. The privacy type of messages can be set to public or private. BuddyMob is for free.

2.2.12 iFob The application iFob [23] enables location-based messaging on the iPhone and iPod only and works with hotspots (WIFI) similar to Herecast. They explicitly claim not to be a social platform like facebook and aim for easy of use

[18] www.bluemapia.com (2009)
[19] www.belysio.com (2009)
[20] www.icloseby.com (2009)
[21] www.buddycloud.com (2009)
[22] www.buddymob.com (2009)
[23] www.icloseby.com (2009)

by avoiding required fields, acceptance forms and commitments or fees during subscription. There is no privacy agreement or information concerning privacy issues available online or on the client.

2.2.13 Okud Okud [24] is another location-based social platform available for the iPhone only. Picture messages made with the phone can be posted very easily online at the public server. Airwriting messages will have many attributes. One of them is their visibility status. This status defines the life-span of messages, and also what happens if the life-span expires. After the expiration of a message, it does not necessarily have to die. There is also the possibility that messages just go public after some time (or their expiration) at the online portal so that everybody can see them. In this sense, Okud and Airwriting have some similarities.

2.2.14 Aka'aki Aka'aki [25] is a location-based social platform, providing the same functionalities like Locle; sharing your location with others. Additionally, they offer a service for avoiding internet fees, named *enchanter*, which provides transmitting the data via bluetooth to the PC and visa versa instead of transmitting it with the mobile phone provider.

2.2.15 Quiro Quiro [26] is another location-based social platform, available for Germany only. It supports finding and talking to nearby friends, creating bookmarks and offers location based information. There is also the option for becoming invisible. The software can only be downloaded, if the user is telling the right telephone number.

2.2.16 Whereyougonnabe Whereyougonnabe [27] comes with various plug ins or synchronization functionalities with facebook, microsoft outlook, google calendar, tripit [28] and dopplr [29]. It is about sharing the *future* location instead of the current one.

2.2.17 Locle Locle [30] is a location-based platform. Finding friends is also possible without GPS-enabled mobile phones with the help of cell phone triangulation. Locle integrates with Facebook. Bebo and MySpace integration is planed.

---

[24] www.iokud.de (2009)
[25] www.aka-aki.com (2009)
[26] www.myquiro.com (2009)
[27] www.whereyougonnabe.com (2009)
[28] www.tripit.com (2009)
[29] www.dopplr.com (2009)
[30] www.locle.com (2009)

2.3 Social Platforms II (Twitter-Services)

Twitter is messaging services for micro-blogs, therefore blogs limited up to 140 characters. The clients for twitter-services enable updating or creating "twitters" easily and hence, add spatial information to them. Twitter services are easy to implement, and therefore airwriting will also support them.

2.3.1 GPSTwit GPSTwit [31] is a client for the iPhone and Windows Mobile Platform for posting messages to twitter. The support for other platforms like Facebook and BrightKite is planed. There is no "online platform" on the web.

2.3.2 BrightKite BrightKite [32] is another free project similar to GPSTwitter. In addition, they have a feature implemented named "'universe"' showing registered users which are (contrary to the typical "'nearby"' function) far away. Their public available API is build upon the same intention as plazes. BrightKite is available for the iPhone only.

2.3.3 Blummi!.com Blummi!.com [33] is in private beta and therefore not available to test. The idea is similar to GPSTwit but varies in its technical implementation. They propose to implement technologies like micro-format, push, openid and oath.

2.3.4 Sparrow Sparrow [34] is a twitter service for the FireEagly and Twitter platforms. There is no "online platform" on the web and this application is available for the iPhone only.

2.4 Social Platforms III (Photo Blogging Applications)

Photo blogging applications enable posting photos online, viewing and - sometimes - rating them. Location based clients add the spatial information to these photos and are supporting the camera of the mobile device. Airwriting will definitely support messages which have a picture attribute added. If and how blogs will be supported by airwriting is still an open question.

2.4.1 EarthScape EarthScape [35] is a location based photo-blogging application for the iPhone and using a globe-view similar to google earth instead of the list- or radar view. The user can zoom and scroll the globe map by moving the finger on the screen easily for viewing pictures of the community.

[31] www.gpstwit.com (2009)
[32] www.brightkite.com (2009)
[33] www.blummi.com (2009)
[34] clickontyler.com/sparrow (2009)
[35] www.earthscape.com (2009)

2.4.2 LocoBlog LocoBlog [36] is similar to EarthScape but available for J2ME clients only. Airwriting will never support only one device platform nor will it start its services for only one platform in the beginning.

2.5 Social Platforms IV (Dating Services)

Location based dating services allow singles to find each other more easily. The success of such services highly depends on the number of devices supporting GPS or similar techniques. If there are only few potential partners around, such services will fail instantly.

2.5.1 MeetMoi MeetMoi [37] is a location-based dating service. It informs a user about nearby and matching user-profiles to get in contact. There is a free and premium membership. Only the premium membership allows sending initial messages. The free membership offers creating the profile, viewing profiles and sending winks (a standard message to show someone's interest). There will also be a "love" group in airwriting supporting new relationships for singles. If people - for example - are in the same mood or emotional state, they can see each other and start to communicate with each other.

2.5.2 Skout Skout [38] is another location-based dating service similar to MeetMoi but free. It additionally can be used with the laptop. Interestingly, MeetMoi and Skout have the same interface component which is responsible for realizing the looking for people functionality.

2.6 Location Information Services

Location-based information services offer - as the name implies - information concerning the location. This covers all nearby and static [39] entities (e.g. buildings, places, streets), for which information is available.

2.6.1 InfoRadar InfoRadar [14] is a location-based messaging service for PDAs, which is using a radar-view instead of a list-view for the user interface. The authors claim that *(..) "the UI used in location-based messaging should remind the user of being in a mixed-reality space while making reference to physical artifacts easy and natural, providing a substantially different UI compared to e-mail applications."* Furthermore, they explicitly mention attributes for their messaging architecture*"the user defines two attributes (...)"*. It is still open if airwriting will offer a radar view in the future. Airwriting will definitely show messages list-based, and map based. A map based view is very similar to a radar view.

[36] www.locoblog.com (2009)
[37] www.meetmoi.com (2009)
[38] www.skout.com (2009)
[39] dynamic entities like people are not covered, but covered in location based social applications

2.6.2 DGRadar DGRadar [40] is one of the very few location-based projects iplementing the radar-view mentioned in section 2.6.1. If the topic list (photo, restaurant, traffic, travel, shopping, wikipedia) is selected, related virtual object indicators (points) occur on the zoomable radar. It is available for apple's iPhone and iPod Touch. The japanese language is fully supported, the english only inadequately. Figure 1 shows a radar view used in the application DGRadar.

Fig. 1. A radar view used for location based messages. Source: http://www.dgradar.com/ (2009)

2.6.3 findbyclick The application findbyclick [41] invites users to add and text *points* onto google map. The idea behind this project is offering a simpler user interface compared to the interface provided for google maps. The software is available for the symbian platform. Airwriting won't put a layer onto existing services to enhance such services. Airwriting will provide its own unique services or services which compete with third parties.

2.6.4 Urbanspoon Urbanspoon [42] is location based application exclusively for Apple's iPhone 2.0. It is one of the winner's of the 2008 Best iPhone App Ever Award [43]. If the user shakes the iPhone, the software Urbanspoon visualizes a "slot machine" showing different restaurants nearby with different tastes and price categories. Currently, its available for North American Cities and London, only. The accelerator sensor will definitely be supported in a future version of Airwriting.

[40] www.dgradar.com (2009)
[41] www.findbyclick.com (2009)
[42] www.urbanspoon.com (2009)
[43] www.bestappever.com/awards (2009)

2.6.5 AroundMe AroundMe [44] is a location based application available for the iPhone only in English, French, German, Italian, Portuguese, Swedish and Japanese. It allows the user to quickly find information about the closest bar, bank, gas station, hospital, hotel, movie theatre, restaurant, pubs, parking possibilities, supermarket and taxi. If the object of interest is selected, a route can be shown. AroundMe is free and sponsored by ads.

2.6.6 Where Where [45] is a location-based platform providing *widgets*; "Applications" which can be added to the main Where-application. E.g., weather, ski-reports, golf courses and favorite places. Airwriting will not provide widgets in the near future, but a subset of attributes to third parties.

2.6.7 Locly Locly [46] offers exact the same functionalities as AroundMe, despite the language support and the routing feature, which calculates and shows the route from the current location to the place of interest. It is available for the iPhone only and there is no "online platform" on the web.

2.6.8 Qype Radar Locly [47] is very similar to locly, but additionally showing user-rated score values for the objects of interest. Qype radar is free, available for the iPhone and iPod Touch and supports German, English, French, Spanish, and Portuguese.

2.6.9 FoxyTag FoxyTag [48] [15] is designed for posting local tags to inform (and warn) other car drivers for speed cameras. It presents an approach for solving the trustworthiness problem of such tags using a trust engine.

2.7 Product Finder Applications

These kind of services enable - with the help of barcodes - finding information about products like the price. Airwriting will support barcode scanning in the very near future, but in the first instance, for another application fields. The barcode functionality is an additional possibility to ensure that somebody definitely has been on a certain place. As the name implies, the barcode is a code. Only users who have the code, have been to the place. Airwriting will highly support privacy, but also security. For that reason, the functionality of barcode scanning is mandatory.

---

[44] www.tweakersoft.com/mobile/aroundme.html (2009)
[45] www.where.com (2009)
[46] www.locly.com (2009)
[47] http://www.qype.com/qype-radar (2009)
[48] www.foxytag.com (2009)

2.7.1 CompareEverywhere CompareEverywhere [49] is a location based service for all kind of products. The camera of the mobile phone is used for scanning the product's barcode. With this code, related information is retrieved (e.g. product reviews, differences in price regarding to nearby supermarkets) Compare everywhere is available for the Android OS only and one of the winners of the Android Challenge 2008.

2.7.2 GoCart GoCart [50] is very similar to CompareEverywhere 2.7.1. Thus, the phone's camera is used for scanning the barcode of a product. It is even a winner of the google android challenge 2008.

2.8 Tracking Services

If a user is writing location-based messages in short intervals, others may infer the route of this user. Therefore, many location-based services allow some kind of tracking but are not explicitly build for it, even if they have a simple "friend-finder" functionality. In this section, applications which are explicitly build to track users (or entities like cars) are presented. Airwriting won't track its users, and wont offer full featured tracking services but some friend-finder functionality. Airwriting will always uphold the privacy of its users. Airwriting is a framework, but its aim is not to cover everything, it should find its place and niche among frameworks, but will relinquish certain application fields to others. Therefore, this service (to explicitly track people) is one of the services Airwriting will definitely not support.

2.8.1 cab4me Cab4me [51] is a cab service application and one of the winners of the Google Android Challenge [52] 2008. It is about finding cabs as easy and quickly as possible. The functionality of the online portal lets cab companies define their "'service zones"' on the map. If a user wants to find a cab for the current or selected location, all cab companies listed in such zones are shown and can be called immeadiately. Cab4me is available for the Android OS only. Figure 2 shows three screenshots of the application.

2.8.2 BuddyWay BuddyWay [53] enables tracking, saving and sharing trips with friends, and also supports viewing the tracks with Google Maps 3D. BuddyWay is available for J2ME, Windows Mobile and Blackberry devices.

---

[49] www.compare-everywhere.com (2009)
[50] www.biggu.com (2009)
[51] www.cab4me.com (2009)
[52] http://code.google.com
[53] www.buddyway.com (2009)

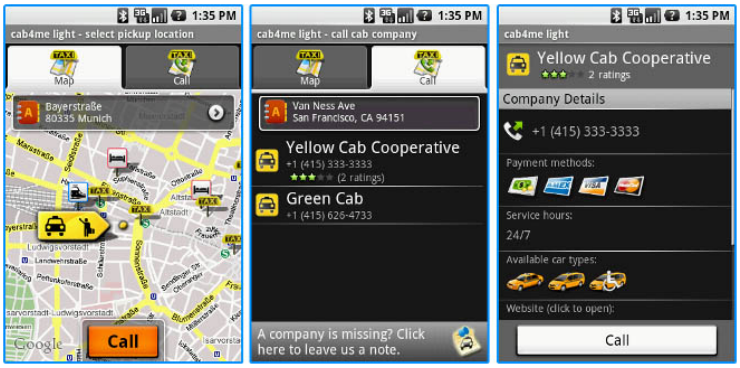


Fig. 2. Three screenshots of the application cab4me. Source: http://www.cab4me.com/ (2009)

2.8.3 MapMyTracks MapMyTracks [54] is a location-based and social platform for sportsperson. It offers creating and tracking all kind of routes (walking, running, swimming) which then may be shared with others. Sport events are listed online and some of them are supported for tracking purposes with MapMyTracks.

2.8.4 ecorio Ecorio [55] is also a winner of the Google Android Challenge 2008. This program tracks the user's trips by public and non-public transfer and calculates the "carbon footprint". For these footprints offsets can be purchased. The idea behind encorio is protecting the environment.

2.8.5 Life360 The focus of Life360 [56] is tracking contacts in real time on the map for security reasons and the possibility to inform contacts in case of emergencies about the current location and the emergency itself. This application is available for the android OS only and one of the top 10 winner's of the Google Android Challenge 2008.

2.8.6 Footprinthistory With footprint [57] virtual footprints are left and shared. To track someone's footprints is possible. There is also a facebook plug in available to public the footprints.

2.8.7 Carticipate Carticipate [58] is an application for sharing motorized trips. The idea behind carticipate is saving gas (thus carbon) and therefore, money.

[54] www.mapmytracks.com (2009)
[55] www.ecorio.com (2009)
[56] www.life360.com (2009)
[57] www.footprinthistory.com (2009)
[58] www.carticipate.com (2009)

Users can post their ride and others may send them a request for joining them. Due to the GPS sensor, carticipate also can match users with the same route. This software is available for the iPhone only.

2.9 Games

Airwriting will explicitly support the design of games. Thus the first test of the application was done with the game scavenger hunt. In the following, some promising location-based games are presented. Airwriting wont offer such a high quality for gaming purposes but is aware of the importance of games in general to stimulate people.

2.9.1 virtualpunk The java and location based game virtualpunk [59] is a RPG[60]-adventure. Players travel around for gaining items and for performing tasks. Monsters may harm the player but are vincible. There is a global high score list for all players available.

2.9.2 Parallel Kingdom Parallel Kingdom [61] is similar to virtualpunk. The players have to travel around to win the game by gaining virtual resources and items and avoiding or defeating virtual monsters in the real world. This game is available for the iPhone, iPod Touch and the Google Android Platform. Figure ?? shows a screenshot of the game. Airwriting as a social platform will never be able to offer such games, but possibly Airwriting as a framework will do.

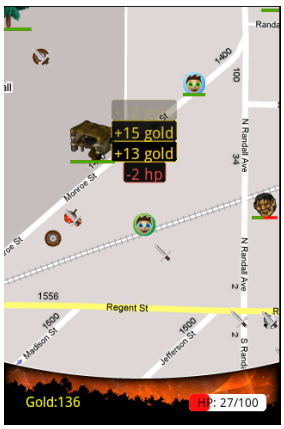


Fig. 3. A screenshot of the game Parallel Kingdom. Source: http://www.parallelkingdom.com/ (2009)

[59] www.virtualpunk.com (2009)
[60] Role Play Gaming
[61] www.parallelkingdom.com (2009)

2.9.3 Tourality Tourality [62] is a location-based game. The rule: Find "spots" before others do. It is possible to play in single user vs. single user mode, or team vs. team mode. Spot games are based on spot templates, which means that different rules for gaining the spots can be defined. Games can be reviewed with the help of game relevant statistics like the traveled distance, ranking and visited spots. Tourality is free, available for J2ME devices and was the winner of the austrian business plan competition challenge. Figure 1 shows one screenshot of the game with spots.

Fig. 4. A radar view used for location based messages. Source: http://tourality.com/ (2009)

2.9.4 Spacerace In their paper, Drab and Binder [16] present the location-based game Spacerace. The idea is to collect virtual objects located in the real world named "crystals" as quick as possible for gaining points in the high score list. The game is a real world approach of virtual capture-the-flag games.

2.9.5 Softrace Softrace [63] is another winner of the Google Android Developer Challenge 2008. This game invites people to compete against each other and is tracking all kind of physical activities but especially running routes.

2.9.6 Pacmanhattan Pacmanhattan.com [64] is a location-based game trying to put the popular game Pacman [65] into the real world. It is not using the GPS sensor for retrieving the location. Currently, it is not available for the public.

2.9.7 Geocache Geocaching [66] is a location-based hide and seek game. Currently 699.060 geoacaches are active. To search for a geocache, lots of search

[62] www.tourality.com (2009)
[63] www.softrace.net (2009)
[64] www.pacmanhattan.com (2009)
[65] http://en.wikipedia.org/wiki/Pac-Man
[66] www.geocaching.com (2009)

functions are offered: by address, zipcode, state, country, keyword, areacode and by waypoint. There is a free and premium membership which differs in the degree of accuracy concerning the search functions. The premium versions enables organizing favorites, custom searches, searching along planed routes, access to features offered by waymarking.com and wherigo.com.

2.9.8 Fastfoot Fastfoot [67] is another multiplayer game. The mission for 4 of the 5 players is to find the fifth one, the mission of the fifth one is not being captured. The playing time is about 25 minutes. There will be different game modes in the future (varying the radius, the game time and amount of players).

2.10 Edutainment

This kind of application is a combination of a game, information and e-learning service. Unfortunately, from all of the projects, only REXplorer is trying to educate people. Airwriting will definitely look for useful application fields which try to educate in a stimulating way.

2.10.1 REXplorer REXplorer is a location- and gesture recognition-based game for tourists. Users have to draw gestures into the space with their rental mobile phone at specific, historical placed buildings to reveal a message. It was designed as an academic experiment and is used for commercial purposes nowadays in Regensburg city.

2.11 Frameworks

2.11.1 FireEagle FireEagle [68] is an Open Source Framework for location-based applications. FireEagle stores the current location of the user, and offers interfaces for accessing or updating it by third party applications for free. Several location-based platforms Voila, Nulaz [69], Fire Fone, BrightKite, Doppir, Dipity [70], ekit, Lightpole, Movable Type, Navizon [71], Outside.in Radar, Loki [72] and the already closed Pownce [73] are build with the help of this framework. Due to the amount of services using FireEagle, it seems to that there is a demand for frameworks offering location-based services. Therefore, Airwriting also will build interfaces for thirds parties.

[67] www.fastfoot.com (2009)
[68] http://fireeagle.yahoo.net/
[69] http://nulaz.net/
[70] http:/dipity.com/
[71] www.navizon.com/
[72] http:/www.loki.com/
[73] http://pownce.com/

2.11.2 Metosphere Metosphere [74] is a *"mobile location browser"* and a *"GeoBlog"* for the android phone. Furthermore, an iPhone Web Application is available. The goal is to provide a framework for location-based services in the future such like geochaching assistant, location messaging client, proximity based kml/GeoRSS Reader, Flash Mob Communicator, Cummunity Campus Emergency Alert Notifier and Meatspace MUD games.

2.12 Non spatial messaging services

2.12.1 Google Wave Google Wave [75] which will be available in autumn 2009 is a promising and highly interactive, collaborative application combining chat-, mail-, newsgroup- and web based word processor services with the help of Web 2.0 technologies. It allows users to chat *within their mails* and even *replaying their conversations*. It is not an attribute based architecture but a services based one. While the former architectures allow selecting and combining attributes, the latter architectures allow selecting and combining services. An attribute based architecture can provide a basis for services based architectures.

2.13 Summary

Figure 5 will give a brief overview of the projects listed above [76] with focus on a) the locating technologies used b) the type of visualization (displaying items) c) the general web relevance estimated by google hits in absolute numbers and categories. Last but not least, d) will indicate if the project has a scientific background. Figure 6 gives a closer look on used attributes. As shown (and highlighted with the brackets), clusters of same attributes could be identified for different applications types. This fact indicates, that the mixture of attributes is determining different application types and that these "mixtures" should be given a special focus for the design of a messaging system. Figure 6 also shows, how privacy and security aware the projects are with the help of a simple color code. This code is explained in table 1.

Color	Available for the public	Privacy Agreement	Custom settings for privacy	Security Functionalities
grey	no	no	no	no
red	yes	no	no	no
orange	yes	yes	no	no
yellow	yes	yes	yes	no

Table 1. The color code reference for for privacy and security issues.

[74] www.metosphere.com (2009)
[75] http://wave.google.com (2009)/
[76] despite Google Wave

Project-name	Locating Techn.	Displaying Items	Google Hits		Scientific
Non-Focused Location Based Messaging Servies					
E-Graffiti	GPS	List View	30000	C	Y
JotYou	GPS	List View	6800	B	
Herecast	WIFI	2-Dim Map View	3000	B	Y
Locale	GPS, Cell-ID	2-Dim Geo-Map View	170.000	D	Y
Enkin	GPS	2-Dim Geo-Map View, Augmented Reality View	13.000	C	Y
GeoWand	GPS, Compass	2-Dim Geo-Map View	240	A	Y
Locatik	GPS	2-Dim Geo-Map View	1280	B	
Google Earth Mobile	GPS; Cell-ID	3-Dim Geo-Earth View	35.000.000	E	
Photo Blogging Applications					
EarthScape	GPS, Cell-ID	3-Dim Geo-Earth View	2550	B	
LocoBlog					
Dating Services					
MeetMoi	GPS	2-Dim Geo-Map View	50.000	C	
Skout					
Social Platforms					
Skobbler	GPS	2-Dim Geo-Map View	113.000	D	
Plazes	GPS	List View	115.000	D	
NowHere	GPS	unknown	2.000	B	
Loopt	GPS, Cell-ID	2-Dim Geo-Map View	570.000	D	
Dimdix	GPS, Cell-ID	2-Dim Geo-Map View	6660	B	
PocketLive	GPS, Cell-ID	2-Dim Geo-Map View	5000	B	
BlueMapia	GPS	2-Dim Geo-Map View	6300	B	
Belysio	GPS	2-Dim Geo-Map View	28.000	C	
MeetMe	unkown	2-Dim List View	800	A	
BuddyCloud	GPS, Cell-ID, BT, WIFI	2-Dim List View	7.500	B	
BuddyMob	GPS, Cell-ID	2-Dim Geo-Map View	6670	B	
iFob	WIFI	List View	2430	B	
Okud	GPS, Cell-ID	List View	60	A	
Aka'aki	GPS, Cell-ID	List View	422.000	D	
Quiro					
Whereyougonnabe	GPS	unknown	200.000	D	
Locle	GPS	2-Dim Geo-Map View	65.500	C	
Social Platforms (Twitter)					
GPSTwit	GPS	no view	300	A	
BrightKite	GPS, Cell-ID	2-Dim Geo-Map View	900.000	D	
Blummi	GPS	unkown	500	A	
Sparrow					
Location Information Services					
InfoRadar	GPS	Radar View	7000	B	Y
DGRadar	GPS, Cell-ID	Radar View	82.000	C	
FindByClick	GPS	2-Dim Geo-Map View	1750	B	
Urbanspoon	GPS, Cell-ID	2-Dim Geo-Map View	500.000	D	
AroundMe					
Where	GPS	2-Dim Geo-Map View	unknown	-	
Locly	GPS, Cell-ID	2-Dim Geo-Map View	4000	B	
Qype Radar	GPS, Cell-ID	2-Dim Geo-Map View	3000	B	
FoxyTag	GPS	Radar View	15.000	C	
Product Finder Applications					
CompareEveryhwere	GPS, Cell-ID	2-Dim Geo-Map View	5000	B	
GoCart	unkown	unkown	300	A	
Tracking Services					
cab4me	GPS, Cell-ID	2-Dim Geo-Map View	7000	B	
BuddyWay					
MapMyTracks	GPS	2-Dim Geo-Map View	6000	B	
Ecorio	GPS, Cell-ID	2-Dim Geo-Map View	35.000	C	Y
Life360	GPS, Cell-ID	2-Dim Geo-Map View	88.600	C	
FootprintHistory	GPS		unknown	E	
Carticipate	GPS, Cell-ID	2-Dim Geo-Map View	581.000	D	
Games					
VirtualPunk	GPS	2-Dim Geo-Map View	500	A	
ParallelKingdom	GPS, Cell-ID	2-Dim Geo-Map View	7000	B	
Tourality	GPS	2-Dim Map View	15.000	C	Y
SpaceRace	GPS	2-Dim Map View	100	A	Y
Softrace	GPS, Cell-ID	2-Dim Geo-Map View	1400	B	
Pacmanhattan	Text	unkown	4500	B	Y
Geocache	GPS	List View	1.050.000	E	
Fastfoot	GPS	Radar View	300.000	D	
Edutainment					
REXplorer	GPS		5660	B	Y
Frameworks					
FireEagle	GPS , Text	none	130.000	D	
Metosphere	GPS, Cell-ID	2-Dim Geo-Map View	4300	B	

Fig. 5. Benchmark of location based projects

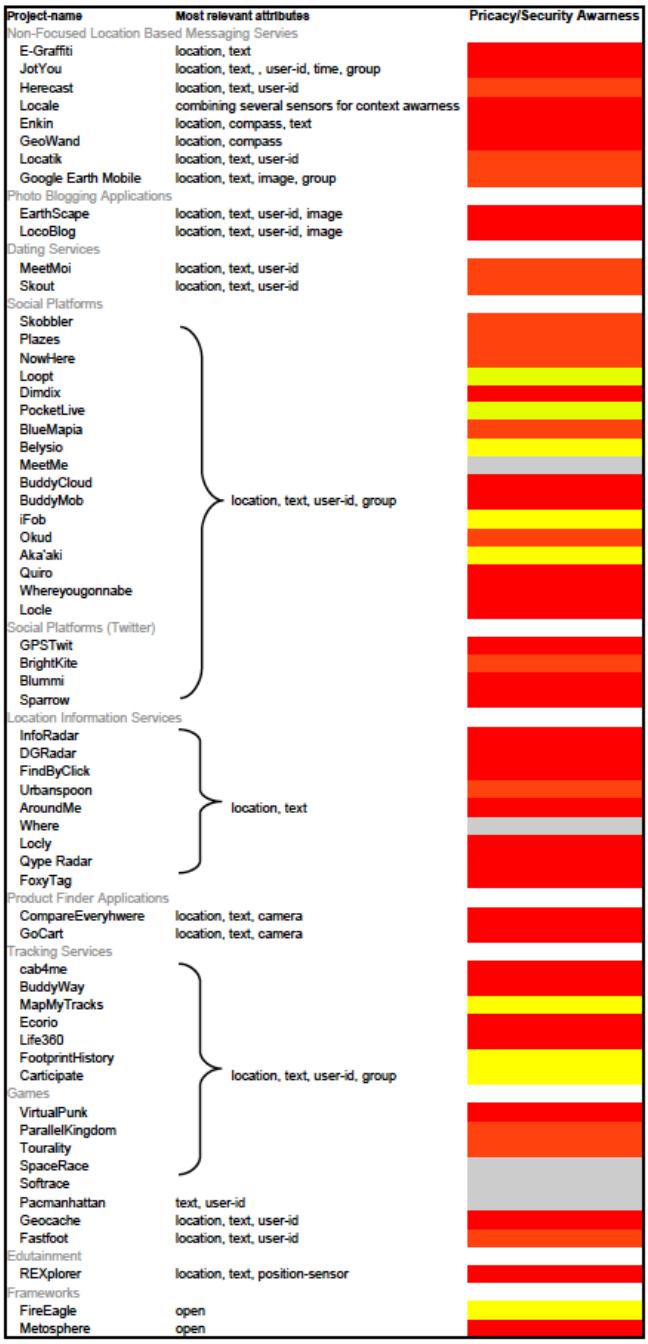


Fig. 6. Benchmark of location based projects II

Most of the location based, social applications mentioned above are providing similar functionalities and have lacks in supporting the user's privacy. It is all about sharing someone's location, and getting proper information concerning the current places. Some of them are offering plug-ins for other platforms like facebook, and others try to offer as many as possible client platforms (android, symbian, iPhone, blackberry, windows mobile) Few, less social platforms but still location based applications like cab4me, Geowand, REXplorer, locale and encorio are innovative [77]. But why are they called innovative? Is it the user-interface? The quality of code, luck, the technical infrastructure (e.g. scalability), marketing or a mixture of these possible reasons? This master-thesis doesn't give concrete answers for such questions, but it extends them with further ones: A location based system may be innovative, if it is a) using new attributes b) using common attributes in a different way or c) if it is reusing attribute combinations (groups) for various applications and offering them to third party platforms. In the following chapter, such an architecture providing these functionalities (a, b, c) is presented in theory in chapter 3 and in chapter 5 as an implementation. The terms attribute and attribute combination will be defined in the following chapter.

[77] 3 of 5 won the google challenge

3 Attribute based Messaging with Mobile Devices

This section is divided into four sections. The main part (section 3.1 will explain the core concept of attribute based messaging, namely attributes. It will explain, what attributes are in this context and how they should be used. It will list various attributes and it will define and explain the following aims for an attributer based architecture;

- explore and provide as many different attributes as imaginable
- explore and offer as many attribute combinations as imaginable
- find groups and applications for such attributes and attribute combinations

The second part (section 3.2 will list some groups, which are reusable attribute mixtures. The third 3.3 and fourth 3.4 part are describing taxonomy and compatibility issues.

3.1 Attributes

Attributes are the main concept for composing messages in Airwriting. In all messaging systems, the most important attribute is the *text* itself. Also, the *group membership* can be seen as an attribute (for group based systems like newsgroups). Another and currently popular attribute is the *location* (for location based messages). One of the initial scientific aims of Airwriting is to *explore and provide as many different attributes as imaginable* regardless of their significance. This is a basis for combinations of attributes, flexible application design, and general exploratory research on messaging applications.

Table 2 illustrates some of the message attributes currently implemented in Airwriting: The first column defines the name, the second shortly explains it's use with an example and the third one defines whether the user must use the attribute to send a message (any non-mandatory attributes can be freely combined as shown in table 5 as long as they are not mutually exclusive). The fourth column declares if the user can set the attribute by entering some attribute values directly through the user interface (keyboard, touchscreen) or if the attribute values are measured by additional sensors (acceleration, temperature, pulse, etc.). This conscious distinction between *directly* and *indirectly* definable attributes is a further and fundamental approach of the message architecture. The last column indicates whether the value assignment of an attribute is enforced or checked by the server or the client. If the client assigns the value, cheating is possible; if the server enforces certain attribute values, it is not. For example, the location attribute could be faked using a modified client. Possible strategies to avoid or defuse cheating are described in [2].

The most of these implemented attributes listed in table 2 can also be identified in various other architectures. Airwriting will offer some more, unique attributes incrementally and some of these attribute candidates are shown on table 3 and table 4.

The second aim of the architecture used for airwriting is to *explore and offer as many attribute combinations as imaginable*, because single attributes are

Attribute	Description / Example	Mandatory	Directly definable	Enforcement
group-id	ID of the group where messages are sent to or received from. E.g., Judas writes his message to the “love” group.	yes	yes	client
longitude, latitude	A message is linked to a specific place with the help of GPS, and therefore is only visible (ready to receive) for another user where the message was sent from. E.g, user Judas writes a message “I want to kiss you” at "Stephansplatz". If his girlfriend Maria is coming to Stephansplatz, she will receive it. Location is determined by GPS and can not be directly defined in the user interface (note that Judas cannot “cheat” his current position using normal clients, but could fake it with a modified one).	yes	no	client
text	The text of the message. E.g., Judas sends Maria the text "I want to kiss you!”.	yes	yes	client
username, password	A user may optionally log in on the client. E.g., user Maria is playing the game Scavenger Hunt (which will be explained in section 5 logs in with her registered nick name, but the second player Judas doesn't and therefore remains anonymous.	no	yes	server
start-date, end-date	Messages will be available a specific time. E.g., Judas wants to send Maria a virtual kiss message on valentine's day. Therefore, he sets the time scope of the message to the 14. February.	no	yes	server
amount	Messages can be grabbed from a place n times. After the n. “pick” action (i.e., reading a message), the message will disappear. E.g., Judas only wants to send one virtual kiss. He sets the pick value to 1 of his message. When Maria receives the kiss message, it will be the only one.	no	yes	server
encrypted	Messages can be locked by textual passwords. E.g., Judas adds a special password to the virtual kiss message that only Maria knows. Nobody besides Maria will be able to read the kiss message.	no	yes	client
radius	The scope (radius in meters) of a message the client is sending. E.g., Judas wants his message to be visible not only at Stephansplatz Church but really at Stephansplatz location. Therefore he sets the location radius to 30 meters. (instead of the default value of 10 meters)	no	yes	client

Table 2. Example for message attributes; all of these are currently implemented in Airwriting, among others.

Attribute	Description / Example
move	Messages can move virtually also around the world. Thus, a location based messages is not linked anymore to a certain place in a static way. A use case for this attribute could be bottle post based gaming applications.
keep	Messages appear, if a user stays on a place for a specific time. This attribute could be part of a sales campaign.
speed	Messages appear, if a specific speed is reached. This attribute is useful for anti radar trap services.
drop	Picked messages disappear from the mobile client and reappear for the community on the current location of the message. A game which requires the transport of a virtual object like could access the functionality of this attribute.
borrow	Picked messages disappear from the mobile client and reappear for the community on the original location of the message. This attribute may realize applications based on petri-nets.
transfer	Picked messages move to another client directly. This attribute enables client based transfers of messages. A use case for this attribute could be hide and seek based games.
global delete	A picked message may be deleted from the server and is therefore not available anymore for others. Secure applications may demand such a feature.
local hide	If a picked message is locally hidden, it is only visible at the server (online platform).
cluster	The messages of a location become visible for n-users, if those n-users are on that location together. This attribute could be part of a sales campaign or group based competition games.
multi place	The same message is visible on different places. This attribute could be part of a sales campaign.
composition	Messages can be split and merged together
rate	Messages can be rated. Social applications will demand such a functionality.
answer	Replying a message is possible. Social applications will demand such a functionality.
change	Messages can be changed completely. This attribute is useful for wiki based applications.
appendix	Messages can be append by messages. Imagine a poet's contest.
sequence	A message becomes visible to the client only, if the predecessor message was received. This attribute is important for games and virtual tour guides.
exclusive	A message X looses it's visibility for a client, if the client received message Y or vice versa. This attribute is important for dynamic games where decisions can be within the game.
puzzle	A message becomes visible, if n users with n different messages meet on a specific place. This attribute is useful for investigative games or sales campaigns.
emotion	This directly definable attribute lets users define their emotional state. Emotional messages are only visible for users with the same emotional state. E.g. groups for depressive people.
money	This directly definable attribute lets users define their willingness to pay. With the help of this attribute, supplier can grab their target group more efficiently.
multi language	A messages is available in various languages.

Table 3. Further attribute candidate examples

Attribute	Description / Example
compass	The visibility of a message is dependent on the direction of the mobile phone. This attribute could help in labyrinth based games or in tourist applications.
microphone	A message is sent by the mobile phone, if a specific sound is occurring and detected with the build-in microphone of the mobile device. Security and health applications could benefit from this functionality.
temperature	A message is sent by the mobile phone, if a specific temperature is exceeding a certain threshold. Security and health applications could benefit from this functionality.
pulse	A message is sent by the mobile phone, if a specific pulse frequence is exceeding a certain threshold. Security and health applications could benefit from this functionality.
cam-scan	A message is sent by the mobile phone, if a specific image or video sequence is captured.
acceleration-scan	A message is sent by the mobile phone, if a specific acceleration value is reached.

Table 4. Additional sensor based attributes

often not enough or even not useful enough for creating a meaningful message. Often, only the combination of attributes creates value. Imagine an eMail message which has a text (attribute) added but no picture (attribute) attached. It could be completely misinterpreted if the text is a serious warning message and the (missing) picture is a funny joke. Hence we believe in the creativity of the user community, which will demand such attribute mixtures for any reason.

Based on the first and second aim, the third one is derived for use cases: Finding *groups* and applications for such attributes and attribute combinations. Different groups re-use such combinations in different ways as shown in table 5 and therefore increase the value of a single combination. In trying to fulfil these three aims, Airwriting follows a top-down approach for realising an attribute based messaging on Figure 7: Some attributes are building several combinations for groups, and such groups have many messages. The purpose of these aims is firstly the attempt for flexibility. If attributes or attribute combinations (and therefore groups which are using them) turn out to be useless, they can simply be ignored without weakening the architecture or implementation. Secondly, it is call for comparative studies to determine why some combinations fail or are successful. And finally, it provides the capability for managing a vast amount of new message types and messages itself.

3.2 Groups

For efficiency, flexibility and user-experience reasons, the visibility of messages is regulated by their membership to groups. A group consists of the group name (freely definable by the creator, which can be any registered user), the creation date, the attribute mixture set (and if all of these attributes are mandatory for

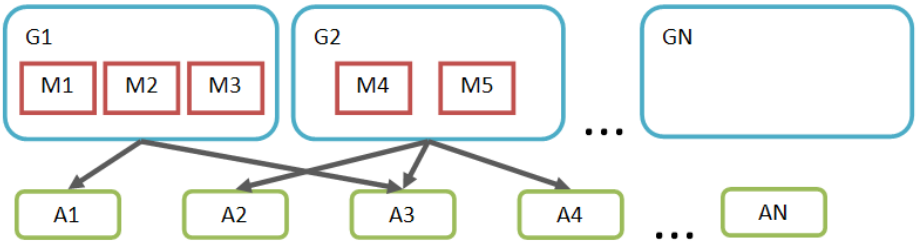


Fig. 7. Service Model from the user's point of view

messages to this group), a list of members, messages from users and membership settings which can be private or public.

Table 5 shows three examples for groups, whereas the group "Scavenger Hunt" is described in more detail in section 6.1. This particular group is not required to set all attributes; for example the "encrypted" attribute in this group is not mandatory and therefore, messages to this groups can consist of locked (encrypted) and unlocked (plain text) messages.

Group name	Combined set Of attributes	Description / Example	All attributes have to be set
Aidtips	group-id, text, longitude, latitude,	Provide additional information for people with disabilities - A disabled person visits a tourist attraction and suddenly needs a wheelchair accessible toilette.	yes
Amor	group-id, username, text, longitude, latitude, start-date, end-date	A location based contact service - A single is looking for a partner within a radius of 50km on valentine's day.	yes
Scavenger Hunt	group-id, text, longitude, latitude, encrypted	A location based game - A group of students is playing "scavenger hunt" - to win, they need to unravel some mysteries.	no
History	group-id, text, longitude, latitude, start-date, end-date, change	The idea of history messages is simple; Messages which will never be deleted. They should be historical, related to the place, interesting and useful for future generations.	no

Table 5. Group Examples

3.3 Taxonomy

Table 2 and Table 3 show more than 35 attributes. This amount leads to two questions. Firstly, how many *different* attributes may exist? A classification of attributes could help to answer this question and the knowledge of attribute-types would be highly helpful for a clear and scaleable design of an attribute based architecture. The second question is, how should an attribute based architecture be designed in terms of efficiency, scalability, extendability and compatibility (see next section)? If attributes are the core-concept, their nature should be well understood. A taxonomy is the first step to categorize attributes. In the following, some possible classifications are briefly presented.

Attributes could be classified by their *amount of parameters*. The location has two parameters, the longitude and latitude value. In this case one may argue, that the location attribute is just a combination of other attributes. Therefore, attributes could be classified by *the amount of combined attributes*, too. Another classification could take the *dependency* into account. The encryption attribute is dependent from the text attribute. It makes no sense for the encryption attribute to exist, if there isn't something to encrypt. Attributes may also be classified by their ability to be substituted or not. In the future, the GPS attribute could be substituted by cell phone triangulation. A user centered classification would be the *classification by ranking*. Attributes or attribute combinations (groups) which are used frequently, have a higher value than attributes which are used infrequently. The classification *ease of implementation* may help in coming to a decision, if one of several similar attributes has to be chosen for implementation. One further classification type could be the *amount of capable client models*. Last but not least, the two other possible classification types were mentioned before; Enforcement and directly definable.

3.4 Compatibility

Attribute based architectures should also be able to communicate with other attribute based architectures, whereby those architectures can have different sets of attributes and communication protocols. Therefore, they should be able to handle *incomplete* or *open* sets of attributes. This means that attribute based messages should also declare an *implementation request level* for their recipient. Imagine two different messages (A, B) which have two similar attributes (picture, text) and two different clients (X, Y) which are both only implementing one attribute. The one client (X) is a text only client for visually impaired people. This client must interpret the text attribute and should ignore the picture attribute of message A. It should read and parse as much as possible from the message, ignore unknown attributes and translate the content understandingly. The implementation request level for this application scenario is low. The other client (Y) wants to receive a picture of message B without interpreting the text attribute. But this is - in this example - prohibited. The content of this message consists of both the picture and text attribute. A client interpreting this message has to read these two attributes, otherwise this message should not be readable

at all. In this case, the implementation request level is high. Messages for legal and secure transactions may demand this functionality.

4 User Interfaces of Mobile Devices

Mobile devices are - compared to personal computers - limited in terms of available screen space, interaction possibilities, cpu/gpu power, internet connection speed, and internet connection reliability. Even if researchers overcome some of these restrictions in the future, it is most likely that there will always remain a gap [17] in mobile and non-mobile devices - at least for their field of applications. Therefore, usability guidelines for personal computers cannot be adapted without modifications to mobile devices, although some generic similarities exist. In this chapter, the theory of mobile user interface design will be shortly presented as well as the practical implementation regarding the theory.

Hassanein [18] and Head present a holistic ubiquitous usability model which includes the elements user, task, interface and environment as illustrated in figure 8. This model helps in finding an appropriate interface type for mobile and non-mobile applications.

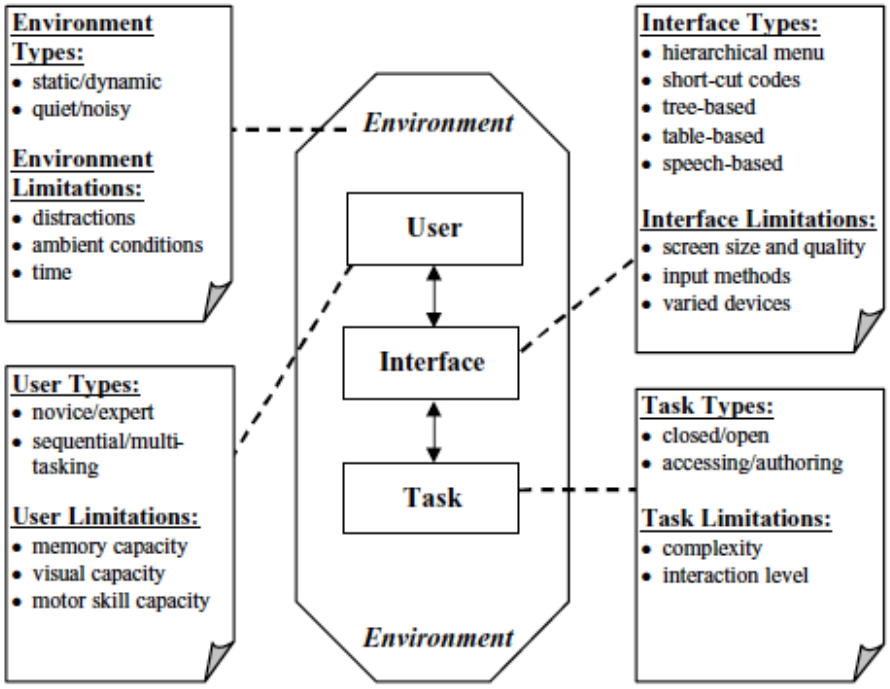


Fig. 8. Ubiquitous Usability Model from Hassanein and Head

Interface types and limitations are the elements of this model. E.g.: The environment type is static like an office workplace or dynamic like a walking

tourist. One user limitation is the visual capacity. A special focus of this model is placed to interface types which are hierarchical, short cut codes, tree-based, table based and speech based. These types have advantages and disadvantages relating to the elements user, task and environment. The authors compare all of these four interface types, table 9 shows the advantages and disadvantages of two of these types, which are used for the user interface of the presented implementation in chapter 5.

This model lacks in providing concrete usability guidelines for mobile devices but acts as a good, first entry point in this material. A closer look in terms of usability for mobile devices gives Kane et al [19] in *evaluating the feasibility of walking user interfaces (WUIs)*. They were increasing the size of the virtual elements (list, button) of the user interface and came to the result, that re-adjusting the size was very useful. For that reason, all virtual elements of the implementations of the mobile clients can be easily adapted in size. Gong and Tarasewich [20] list guidelines for mobile devices explicitly. Due to the prototype character of the clients, not all guidelines were implemented [78].

Nikkanen [21] tries to provide a complete list of *general*, *content* and *navigation* design guidelines for mobile devices. This list is the result of literature research studies from the author, not a model and not holistic (top-down) as the model from Hassanein and Head [18] but some may interpret this literature exploration as a bottom-up approach and therefore, it is not less useful and will be quoted below. Thus, references to the implementation of Airwriting [79] will be established for each quoted guideline to explain how the guidelines were taken into account.

General design guidelines

- *Design for users on the go. The design for mobile devices must include context and forgiveness [22], and provide time-critical information [23].*
  On the go means that in every little moment the telephone could ring or something else could happen causing the user to close or pause the running application immediately. For that reason the message, login and password text input fields are auto-saved. If the user has no time to press *ok* or *save* the text content is not lost and will reappear if the user is restarting the application again.
- *Enable fast use. Two major considerations for the users of a mobile service are the cost of access and the speed of downloading content [24]. Many users are paying for mobile services by the minute, so if they cannot get the information they are looking for within a short period of time they will stop using the service [25] [26].*
  The GPS initialization process takes some time up to few minutes which could cause the user to lose patience. For that reason the GPS initialization

[78] For example, the personalization guideline supporting different screen designs was not implemented - but could in later versions.

[79] These references refer to the Android version only as the most advanced implementation

Tree-Based	***User*** • supports novice users [37] • relies on recognition rather than recall (addressing the memory capacity limitation) [32] • moderately reduces the average number of key presses (addressing the motor skill limitation)	***User*** • somewhat cumbersome for expert users [37] • tree structure is based on designer, not user, intuition [29] • quickly occupies limited screen space (contributing to the visual capacity limitation)
	Task • reduces backtracking actions (reducing complexity for accessing tasks) • overall information structure is conveyed (reducing complexity for accessing tasks)	***Task*** • increases screen clutter (contributing to the complexity limitation) • difficult to use with a deep and/or broad tree structure [27]
	Environment • appropriate for both quiet and noisy environments • time efficient	***Environment*** • awkward within certain ambient conditions (in particular, dim lighting)
Table-Based	***User*** • could support both novice and expert user • relies on recognition rather than recall (addressing the memory capacity limitation) [32] • reduces the average number of key presses (addressing the motor skill limitation)	***User*** • table structure is based on designer, not user, intuition • quickly occupies limited screen space (contributing to the visual capacity limitation)
	Task • somewhat reduces backtracking actions (reducing complexity for accessing tasks) • information structure is partially conveyed (reducing complexity for accessing tasks) • can present more data in a tight space [27],[33]	***Task*** • increases screen clutter (contributing to the data limitation) • may not be suited for all types of tasks, applications or information
	Environment • appropriate for both quiet and noisy environments • somewhat time efficient	***Environment*** • awkward within certain ambient conditions (in particular, dim lighting)

Fig. 9. Comparison of the tree and table based interface from Hassanein and Head

process is visualized; The Google Map View is visible and indicating the GPS status.

- *Keep it simple. The old adages about keeping a system simple stupid and about "less being more" certainly apply for mobile devices and services. For instance, the most successful PDA devices do not attempt to replace the PC, but to complement the PC use, and the use of some other traditional tools [27].*
  Further usability studies are planed to improve user experience. Hence, the logging system of Airwriting supports counting the number of clicks. One aim to improve the user experience is decreasing this number with every update as long as it makes sense.
- *Provide feedback and navigation cues. It should be obvious what the application is, and how one can navigate from the page [28] [22].*
  There are no activities which do not give feedback. If the user is navigating, there is either a navigation bar or a back-item visible. If the user is creating or sending a message, feedback is given by notification boxes.
- *Include self-recovering capabilities. Even if the network goes down, the service or application need not [27] [22]. There should be means to restore the values or written text, or to have them restored automatically.*
  Written text is auto-saved. If the network is down or GPS is not available, an information box is shown to the user.

Content design guidelines

- *Present the most important content first. The most important content should appear at the top of the page [29] [30] [27] [23] [22].*
  The start screen is showing the most important features; the map with messages if they exist, the read and write buttons and user profile indicators. Everything else is accessible with the main menu or option menu.
- *Keep content compact. It is recommended to keep the pages short [29] [30] [31] [32] [25] [27].*
  There are only two list views implemented. The create group view is a list, and the create message view. All the other pages do not have scroll or list views, thus the visible content of the other views fill the standard display size by default.
- *Don't make the page layout complicated. It is recommended to keep pages simple and task-oriented, possibly text only, and to avoid elements that don't add direct value to the content [29] [31] [25] [27].*
  There are no picture elements despite the message icon in the message screen, the cloud icon to indicate messages on the map and the group icons in the group screen.
- *Use simple text elements and styles. The elements used in text layout should be clear and simple [29] [25] [24] [22].*
  The standard and recommended text and element styles are taken.
- *Pay attention to page titles. It is important that the page title elements are descriptive, since they enable bookmarking and knowing where one is [32]*

[23] [26]. The titles should however be short, preferably less than 15 characters [25] [27].
The page titles have the same or very similar name to link names. Bookmarking is not necessary and even not possible because the views are no web views but non linkable views within the application.

- *Keep documents small. Because there are various memory restrictions in mobile devices, the documents should be kept as small as possible [25] [24].*
  Each view is showing one functionality. This is the recommended approach as mentioned in the Android guideline documentation.
- *Use compact link names. Long linked text can make a page difficult to read and time consuming to scroll. It is recommended to use only one or two words as the title of the link [24] [22].*
  The link names of the main screen and the menu link names (which are shown in the menu view) consist of only one word. No further link names exist.
- *Design clear forms. Forms should not be too long [32]. A clear way to cancel the form filling and for going back should be provided, but attention should be paid to form resets, since on small devices, forms are laborious to refill if all values are reset by accident [24].*
  Each text input field has a hint text by default. These hint texts support the "clearness-guideline". Every form used in the Airwriting client software is short because there is no need to scroll. Hence and as mentioned before, the written values are saved automatically, even if the client software (airwriting) is interrupted due a call or something else.
- *Use smart graphics. If graphics are used at all on small devices, they should be made informative, small and simple [27].*
  As mentioned before, only few graphic elements are used.

Navigation design guidelines

- *Minimize steps in navigation. With small screen devices, it is very important to design for economy of navigation [29] [28] [32] [23] [24]. Users will be frustrated by scrolling through long lists of options, filling out complex search forms, and seeing needless pages along the navigation path.*
  The guidelines of the iPhone usability documentation are recommending a navigation depth of 3. This depth is taken into account for Airwriting continuously.
- *Selecting instead of typing. It is recommended to consider whether it is possible to ask the user to choose from a default list using select lists, checkboxes or radio buttons rather than typing in a selection [29] [25] [27] [26] [24] [22]. Alternatively one can offer a default list together with an input box.*
  No selection has to be typed.
- *Keep the navigation consistent throughout the service. The way in which a user makes his or her way through the pages that constitute a service, interacting via links, menus and data input should be kept consistent throughout the service [25] [22].*

The use of Airwriting is straightforward; The main screen is showing the read and write menu buttons and the user profile indicators.

- *Design flat menus. It is recommended to keep menus flat, because it is often difficult to form an overview of a service containing too many layers, and because a deep hierarchy makes the use more difficult [29] [25] [23] [22].*
  The navigation depth [80] never exceeds the depth of 3 as mentioned in the recommendations of the iPhone usability documentation guidelines.
- *Cross link. The Back functionality is the most important way to go back. However, when users need to go back several levels, links to the starting page and subsection main pages are useful [32] [25] [23] [22]. A simple tree design is efficient, but the deeper the navigational hierarchy gets, the more necessary it becomes to get back to the starting point, and also to other pages.*
  All mobile phones for the Android OS are providing a back button by default. Due to the short navigation level depth of 3, no links are necessary.
- *Provide confirmations for important actions. Confirmations must be there for actions like changing important values or deleting items. Even though the user needs to click OK on the confirmation page, that requires much less effort than e.g. returning to a list to check if an item was really removed [32].*
  There are confirmation messages for sending messages, login and registering. The beta version does not allow to delete messages or groups from the client, and therefore, no confirmation message for deleting is needed.
- *Searching should be intuitive. Searching should be a step-by-step, logical process [23]. Once the search is performed, the results must be easy to scan, and the information should enable making good, informed choices within the results [28] [32] [23].*
  There are currently no search functions available for the Android beta version. The only search feature which will be available is group-search. The next version will provide a text input field to search for group names. The results will be shown in the same group list which is showing all local groups. If the user wants to make a choice, and therefore select a group, the group list item has to be selected. This approach is implementing the recommendation of the Android usability guidelines.

[80] Each view may link to another view and this number of view-links is called navigation depth

5 Implementation

This chapter will present an implementation of an attribute based messaging architecture named 'Airwriting'. It is a private, mobile, group based and spatial messaging service. It is implementing all attributes listed on Table 2. Furthermore, it is implementing the group Scavenger Hunt Vienna listed on Table 5 which is explained in section 6.1. In the following, the server, and three clients (iPhone, Android, J2ME) will be presented. Each section is divided into a back-end section describing the functionalities and a front-end section describing the GUI view of the implemented components (server, clients). The server is described more detailed in [2]. The sections itself have a short introduction about why the specific client architecture has been chosen and which challenges had to be faced relating this architecture. For the HTTP communication between the server and the clients, JSON [81] - a lightweight, text based data interchange format - is used.

5.1 Server

5.1.1 Backend The Server Backend of Airwriting is responsible for the following operations:

- The Backend acts as a communication port for mobile clients. Mobile clients query the server for messages in their location. They may also send messages to the server. All the information such as user accounts, groups, messages are managed by the server.
- Provides an internet portal for online functionalities.
- Offers services and aggregated statistical data to third parties.

As seen in Figure 10 several open source frameworks have been combined to realize the server backend. Below there is a short description of each of them:

PostgreSQL [33] is an open source relational database system. Airwriting makes use PostgreSQL on the database layer with the PostGIS [34] extension. PostGIS enables PostgreSQL to operate on geographic objects. Airwriting saves the scope of messages as geometries in the database.

Hibernate [35] is used to map the objects of the application into the relational database and back. The Bussiness logic resides in service classes which are managed by the Spring Framework [36]. These service classes form the controller layer of the server application. They use Hibernate to interact with the database layer. The view layer utilizes the Tapestry5 Framework [37] which is a new component based Web Framework. Tapestry5 builds upon the Java Servlet API, so it works on any servlet container. For Airwriting we use Tomcat as the servlet container.

[81] JavaScript Object Language

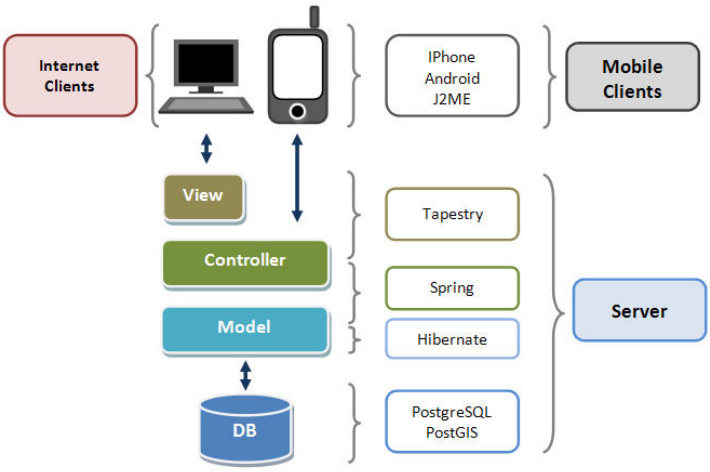


Fig. 10. Infrastructure of Airwriting

5.1.2 Frontend Only creating messages is supported in the prototype version. Reading location based messages is possible, but only realized for testing the server application. If the user wants to read location based messages, the exact location has to be known and set. Figure 11 shows a prototype mask for creating messages. The user can set the address by typing in the address location textually, or by pointing to it with the help of google maps for leaving a location based message. Hence, the radius can be set as well as the amount attribute.

5.2 Android Client

5.2.1 Backend The Back-end of the android client is based on Java, whereby the framework itself is based on linux. The layers of this framework are illustrated on figure 12. Mainly the Application Framework layer is used to implement the client.

The domain model of the android client is simple. In the prototype, (just) all serializable objects are stored if possible. This means the messages, groups and related attribute and setup configurations as shown in figure 13.

All main functional parts should be structured as *activities* in android. Mainly, an activitiy[82] is something the user can interact with visually within a window. Due to the importance of activities, their live-cycle is shown in figure 14. Between starting, running and killing activities, there are also other states needed to handle incoming calls or other disrupting processes like onPause(), onStop(), onResume() and on Destroy().

The activities of the airwriting android client are shown in figure 15. It is easy to see, that all activities of the airwriting client are one-to-one to the window-

[82] http://www.code.google.com/android/reference/android/app/Activity.html (2009)

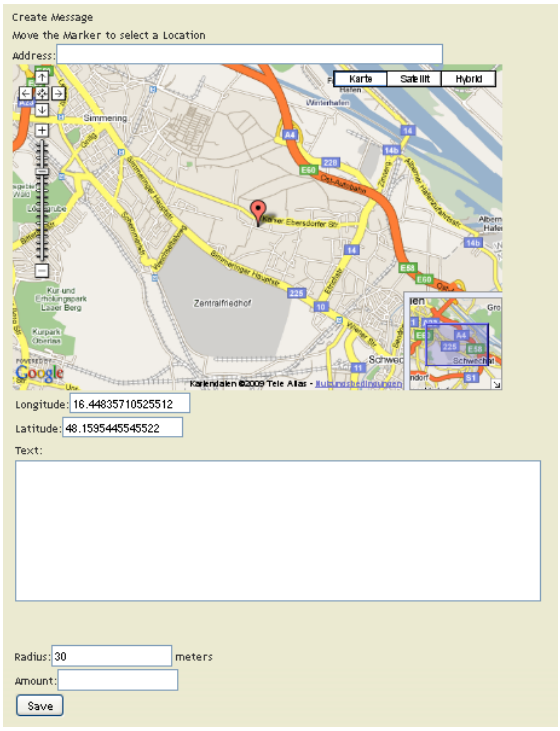


Fig. 11. Creating a message with the prototype version of the online platform of airwriting

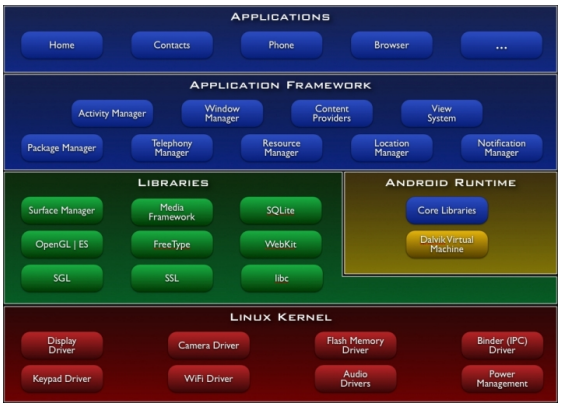


Fig. 12. Layers of the Android Framework

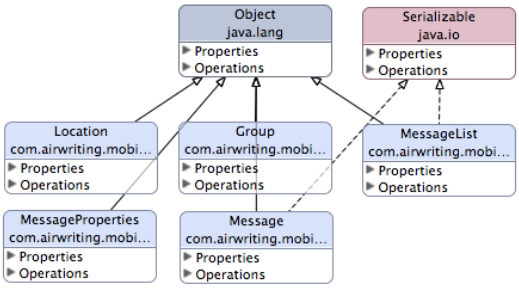


Fig. 13. The domain model of the Android Client

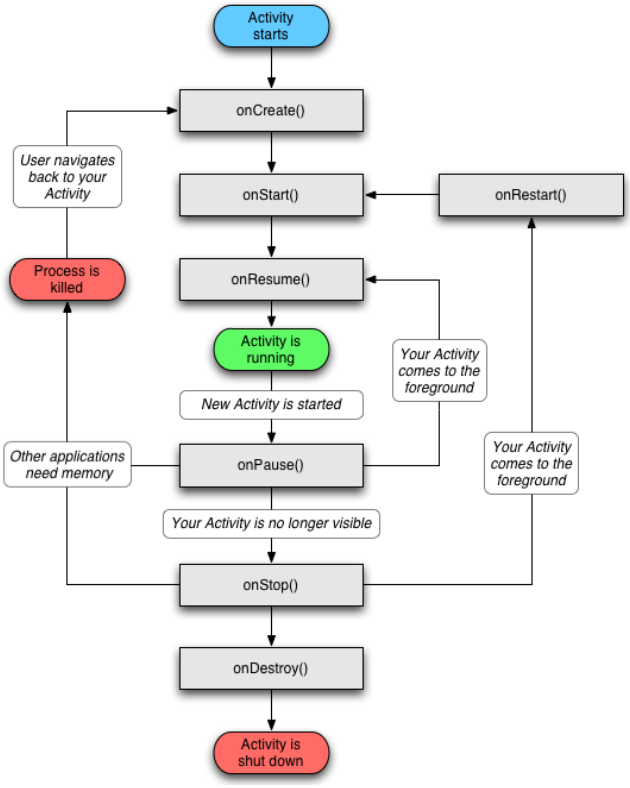


Fig. 14. The Activity Live-cycle of the Android Framework

views on figure 19(a). The login view is not visible, but appears at the first time starting the client and can be reopened again if the standard menu[83] button is clicked. The list activity, right to figure 15 is similar to the activity, but enables list views and is used for reading the header of messages.

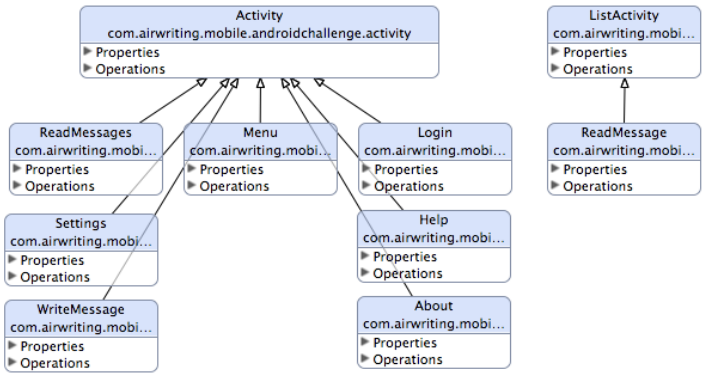


Fig. 15. Activities of the Android Client

The visual elements of the activities are build with various additional helper classes as shown in figure 16. These helpers realize for example the icon-text based menu view as mentioned in the front-end section. They also help in layouting the elements. They android framework offers two different ways to layout elements; Coded or with the help of XML files as shown in figure17.

Dialogs[84] as shown in figure 18 are very common in the implementation of this client and are used for all set-able functions.

5.2.2 Frontend Figure 19(a) shows the main screen of the Airwriting Client. The two bigger buttons indicate the two main functions offered, namely reading or writing messages. The smaller buttons have additional functionalities; The "Recommend" function is for recommending the Airwriting Project by mail or sms. "Improve" is the explicit invitation for users to make Airwriting better. "Settings" will offer typical adjustments and options like the connectivity style (automatically or manually), sound behavior and others.

All buttons have a text at the bottom due to future extensions like a) dynamic button generation (standard icons can be used with different textual descriptions) and b) for poor eye sight support - for people with reading impairments, the text size can be increased. This icon and text based menu had to be build and was not a ready class from the framework.

[83] This button is a real and default button on all android hardware devices which have a keyboard. Touchscreen-only devices are not available yet

[84] http://www.code.google.com/android/reference/android/app/Dialog.html (2009)

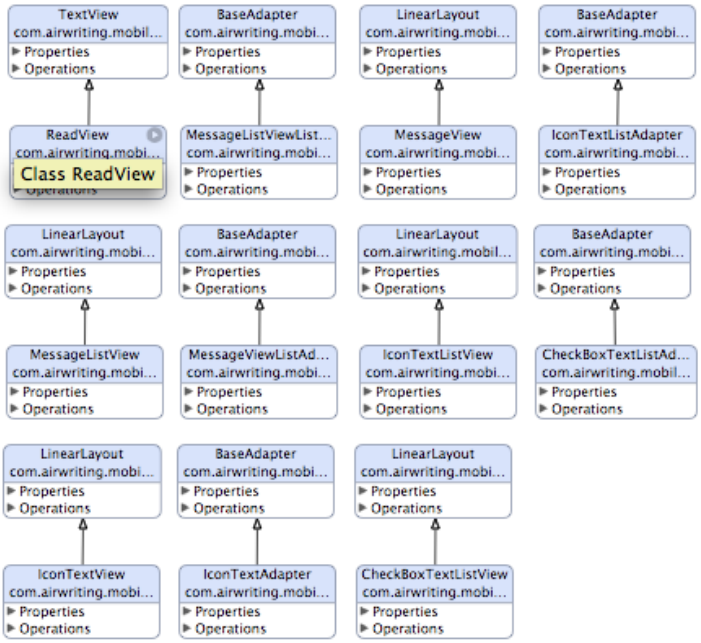


Fig. 16. GUI Helper Classes of the Android Client for Airwriting

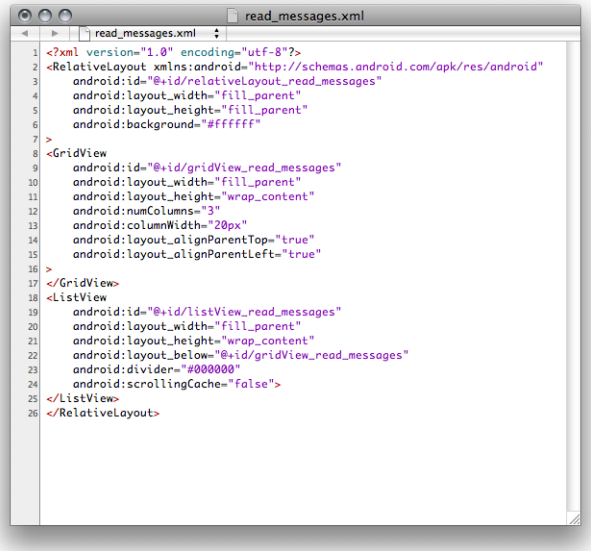


Fig. 17. The Android Client: Lay-outing the GUI with the help of XML

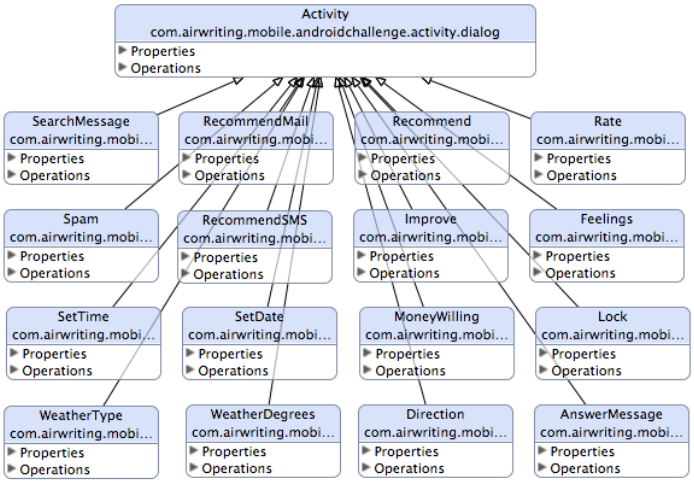
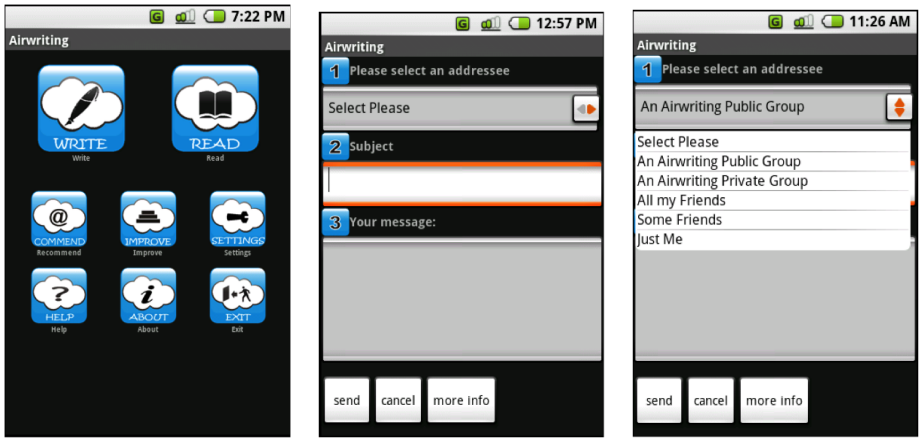


Fig. 18. Dialogs of the Android Client



(a) The main screen of the Android Client Version

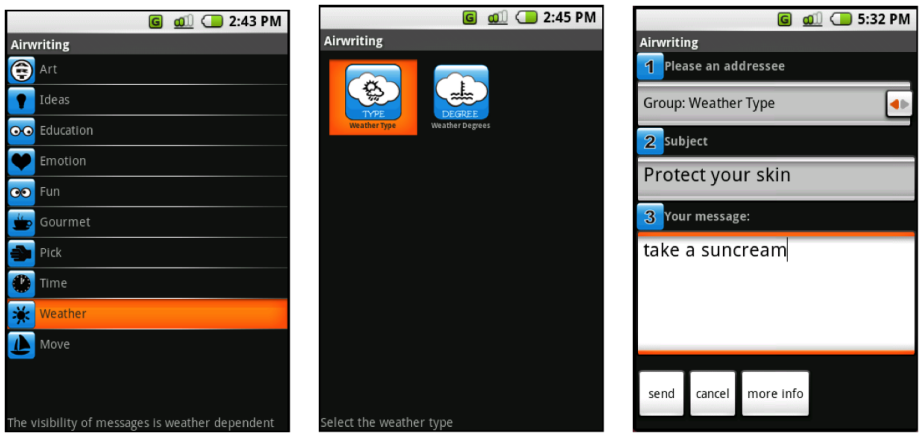
(b) The write menu of the Android Client Version

(c) Selecting an adressee with the Android Client Version

Fig. 19. Screenshots of the Android Client Version

After the selection of write 19(b), the user has to specify an addressee 19(c). There are various possible recipients; All Airwriters - there is a public group named Airwriters visible for everyone. The subscription to this group is set by default for each user. Groups can be public or private and have to be created online at airwriting.com - Public groups are joinable for everyone. An Airwriting Private Group - User need an invitation or password for joining private groups. All my friends - Airwriters can contract friendships among each other. A message written with this selection is visible for all your current and future friends. Some friends - It is the same function as "All my friends" but only a definable subset of friends. Just Me - Write a message to yourself only. It will reappear every time you come back to the place where it was written from.

Let's assume that the user is clicking on the Airwriting Public Group section. Figure 20(a) shows some categories trying to cluster groups with the same attributes. Figure 20(b) demonstrates two examples for weather groups. The first one, named weather type will generate messages which depend on the following weather states: snowy, rainy, and sunny. The second one, weather degrees, generates messages which depends on specific degree ranges. If the user selects one weather group, he is returned to the writing screen 20(c). After writing his message, he has to set the attributes (in this case the weather degree). This logic of setting all the attributes at the end has one reason: Later on, a group will have messages with several message attributes. We think it would be too confusing, if the user has to set them before the message is written. This approach of selecting the attributes at the end will be redesigned in the future and evaluated.



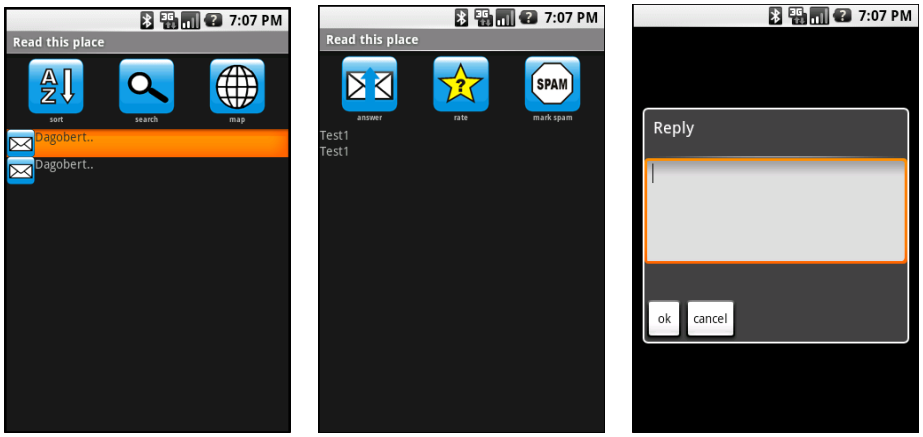
(a) Categories for Groups in Android

(b) Weather Groups in Android

(c) Writing a Weather Message in Android

Fig. 20. Android Client Group Examples

If the user clicks the read button, the read menu appears (figure21(a)) showing headers of messages, a search button which enables to search for the content and headers of a text and a map button visualizing the messages geographically. After the selection of a message, the read view is shown (figure21(b). The user can answer messages (figure21(b)) , rate them if the message is rate-able and mark it - if necessaey - as spam.



(a) The read menu of the Android Client

(b) Reading a message with the Android Client

(c) Replying a message with the Android Client

Fig. 21. Android Client Read Views

5.3 J2ME Client

5.3.1 Backend This client is realized with the help of two frameworks. The first one is the J2ME Polish framework (version 2.0.5), which has a useful precompiler and supports different resource folders for various devices. The second one is the J4ME framework (version 1.0.3), including an additional location API supporting mobile devices which do not officially support the location API but have bluetooth. An additional API is the bouncycastle cryptography API enabling the client to crypt and decrypt messages which have the lock-attribute set. The elements of the graphical user interface can easily be modified in size by altering the CSS values of the CSS file supported by J2MEPolish.

The domain model of the J2ME Airwriting Client Version Prototype consists only of two entities, messages and groups. They are shown at figure 22. The graphical user interface is build with the standard form class, and not that clearly separated as proposed in a MVC-Pattern supported by android and the iphone sdk. Figure 23 shows the architecture of the J2ME GUI.

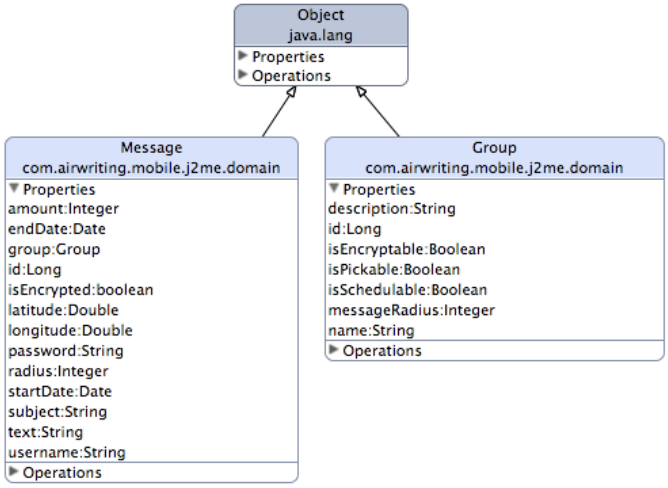


Fig. 22. A simple domain model for the J2ME Airwriting Client Prototype

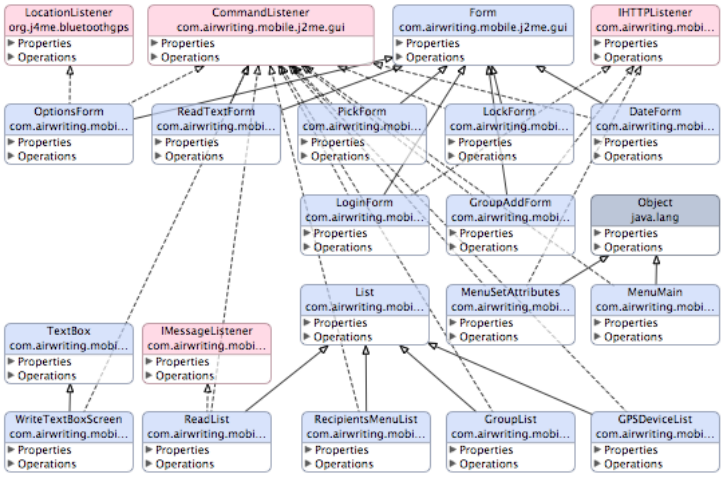


Fig. 23. The GUI architecture of the J2ME Airwriting Client Prototype

The service classes, responsible for retrieving and sending messages are shown in figure 24

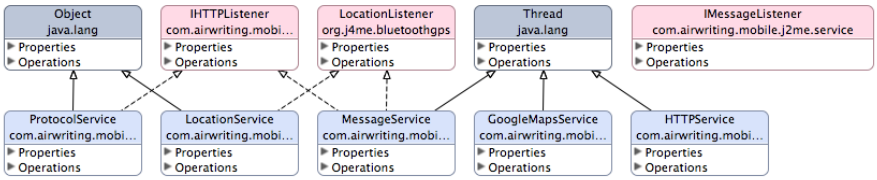


Fig. 24. The service architecture of the J2ME Airwriting Client Prototype

5.3.2 Frontend The front-end of the J2ME client is mainly realized with the extended form class of the j2me polish framework. This extension, compared to the standard J2ME form class allows a better flexibility relating to graphical layouts. Figure 25(a) shows the main screen as a scrollable list. The login view shown in figure 25(b) allows the user to login freely. But, due to the reason this client version was used to test the usability, some kind of identification function had to be implemented. Therefore, the test instructor was adding a dummy - but unique - user name in the option view shown in table 26(d). The test clients were logging all activities (button clicks, http-communication, gps-data) for analyzing usability. These logs were send manually and explicitly by the test instructor after the test as shown in figure 26(e). More about the usability test is written in section 6.

(a) Main screen of the J2ME Client Version

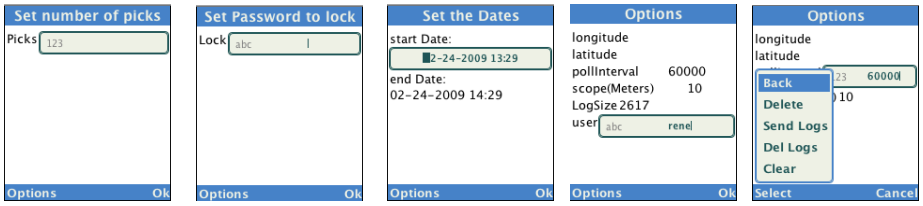
(b) Login screen of the J2ME Client Version

(c) Writing a message with the J2ME Client Version

(d) Selecting a group with the J2ME Client Version

(e) Selecting attributes with the J2ME Client Version

Fig. 25. Screenshots of the J2ME Client Version I



(a) Setting the pick attribute of the J2ME Client Version

(b) Seting the lock attribute with the J2ME Client Version

(c) Setting the time attribute with the J2ME Client Version

(d) Option screen of the J2ME Client Version

(e) Sending the logs with the J2ME Client Version

Fig. 26. Screenshots of the J2ME Client Version II

5.4 iPhone Client

5.4.1 Backend The iPhone SDK supports applications written in C, C++ and Objective C but Objective C is supported best. The Airwriting client has been built completely with Objective C. The prototype doesn't store offline data like messages, selected options or groups and therefore, has no domain model yet. The views are organized by view-controllers, which are similar to activities in android. The view controller itself is responsible for the visual content and delegates handle additional activities like responding to a user selection. The relationship between view controllers and their delegates is shown in figure 27.

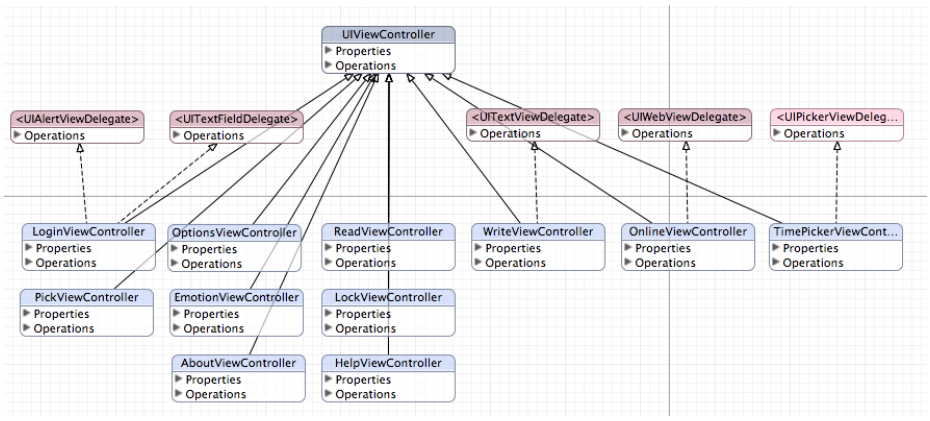


Fig. 27. View controllers and their delegates used in the iPhone Airwriting Client

The relationships between the controllers without views responsible for the internet connections and general application delegates (UIApplicationDelegate, UITabBarDelegate) are shown in figure 28.

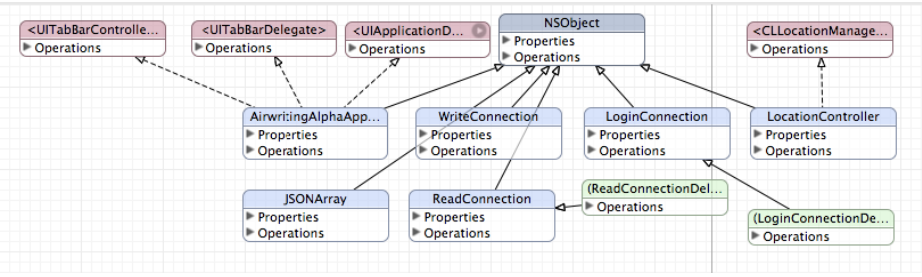
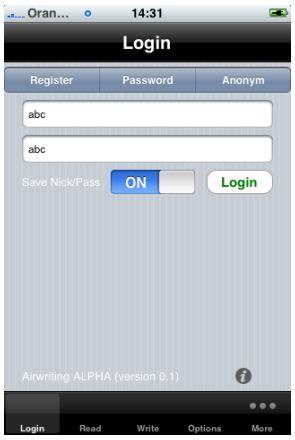
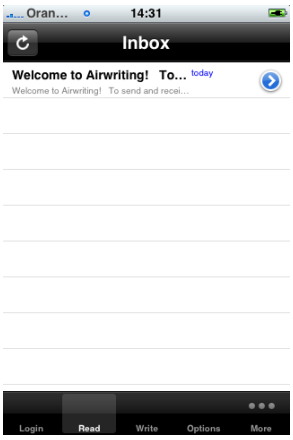


Fig. 28. Application controllers and delegates used in the iPhone Airwriting Client

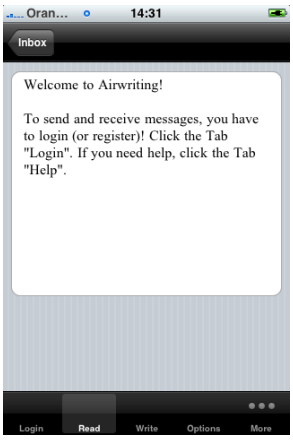
5.4.2 Frontend The GUI of the iPhone client welcomes the user with the login screen as shown on figure 29(a). The UITabBar (on the bottom of the screen) of this figure has the TabBarMenuItems "login", "read", "write", "options", "html reader", "quick message", "help", "about" and "online". The reason for the choice of these UITabBarItems is explained in table 6 shortly.



(a) Login-screen of the iPhone Client Version

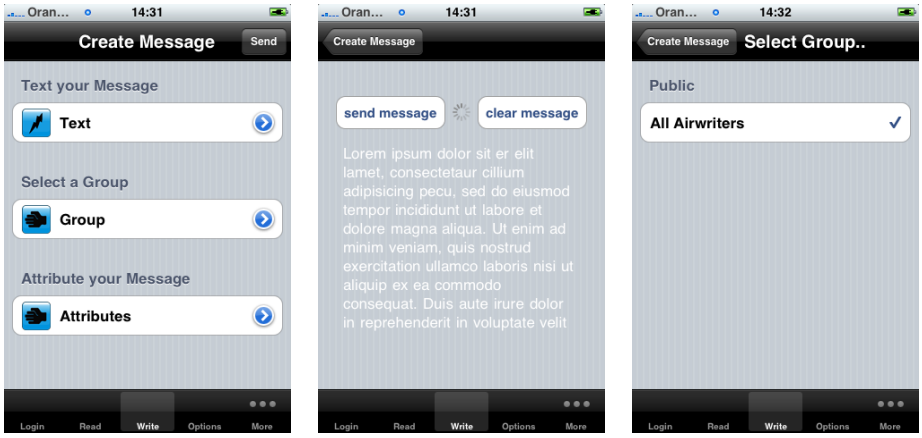


(b) The read menu of the iPhone Client Version



(c) Reading a message with the iPhone Client Version

Fig. 29. Screenshots of the iPhone Client Version I



(a) Write-screen of the iPhone Client Version

(b) The read menu of the iPhone Client Version

(c) Selecting a group with the iPhone Client Version

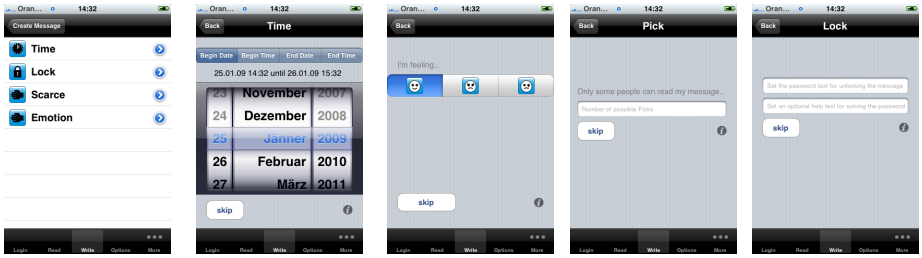
Fig. 30. Screenshots of the iPhone Client Version II

Some of the attributes listed in table 2 are implemented in the iPhone prototype. Figure 31 shows four set-able attributes, namely time, lock, scarce and emotion. The text attribute for setting the text of the message is also an attribute and could occur in this list but was put in the write menu due to its mandatory status [85].

5.5 Experiences, differences and similarities of the clients SDKs

Table 7 will summarize the pros and cons of the three different SKDs and will compare them among each other.

[85] The attributes in this list don't have to be set, but the text must be.



(a) Attribute screen of the iPhone Client Version

(b) Selecting the time attribute of the iPhone Client Version

(c) Selecting the emotion attribute with the iPhone Client Version

(d) Selecting the scarce attribute with the iPhone Client Version

(e) Selecting the lock attribute with the iPhone Client Version

Fig. 31. Screenshots of the iPhone Client Version III

UITabBarItem	Reason for choosing this item
Login	For testing purposes of the anonymity function, the functionality of using the client as a registered user, and checking the server status, the login function was one of the first functions implemented.
Read	Reading local messages is one of the main features of airwriting. For that reason, this item is visible constantly. If new messages are available, a speech ballon appears within this item and shows the number of new messages.
Write	Writing of local messages is the other main feature of the airwriting client. If the user is pressing this button, the writing menu appears as shown on figure 29(b).
Options	The Option Item is one last feature, which was chosen to be visible in the UITabBar permanently. All of the other items are behind the "more" item and are, if the more item is clicked, listed.
HTMLReader	It was very helpful to visualize the plain textual content of the received message before processing it internally by the client's message parser for two reasons. Firstly, we could receive unsupported messages, thus messages which have unknown attributes. Secondly, the debugging process was eased.
Help, About, Online	This items where implemented to complete the appearance of this beta version. If the help item is clicked, some helpful information concerning the use of airwriting is shown. The about item shows the authors of this software and the online item starts safari for establishing a connection to the online platform of airwriting.

Table 6. UITabBar Items of the iPhone chosen for the Airwriting Client

Platform	Pros	Cons
J2ME	– easy to learn – well-engineered – many additional Frameworks available (e.g.: J2ME Polish) – high market share (many phones support J2ME)	– the graphical user interface has to be implemented in a very flexible manner. There is not "one" display size as such for the iPhone. – internet services for phones which have not a big display are still not pushed as for ones with touchscreen from the telecom provider. – GPS is not a standard for J2ME phones. – Java applications are slower than native applications. – different phone models implement different and additional functionalities which leads to problems for programming software consistently
Android	– easy to learn – open source – service orientated architecture – good support (many third party support forums) – one central market store – google challenges – intuitive GUI, big display	– few devices – very slow GUI compared to iPhone OS – each OS update has to be done manually (for developer devices) – multitouch is not enabled on most devices (OS 2.0 will enable it)
iPhone	– fast – internet services are at no extra cost with iPhone contracts – one central market store – intuitive GUI, big display – very popular	– expensive (contract plus iPhone) – high learning curve (Objective C) – the app has to be "approved" by Apple – the App can be removed from the appstore without reasons for a refusal – Apple claims 30% of the profit if applications are sold – the OS is bound to Apple's iPhone – restrictive. No flash and Java applications allowed. – applications are not allowed to run as a background application

Table 7. Comparison of the three different client SDKs

6 Usability Study

The first working concept of Airwriting was realized as a prototype game (subsection 6.1) and was analyzed (subsection 6.2) to test and verify the user's perceivability of the project and the concept at all.

6.1 Initial Prototype Application

The rules and challenges of the game scavenger hunt as shown on Table 8 are easy; 1. All participants in the game have to unravel a mystery question as quick as possible. 2. For that reason, the gameplay owner has to provide one locked and some unlocked messages to the gaming field, an area of approximately one square kilometer. 3. The unlocked messages act as pieces for solving the puzzle question. 4. Additional help messages define the gaming field borders 'border fence is reached'.

Sequenced Instructions	
(1) Welcome to Airwriting at the Technical University of Vienna! This game will take one hour at maximum and is placed on a one-square kilometer gaming field. You will get a notification if you cross the borders. Beware, if you cross them more than 3 times, you will loose this game. To win the game, you have to unravel one of two mystery questions which are located at this starting point (so you have to return to your current location at the end) - Lets start, your first quest is simple: find the lord for your next instruction! Hint: Follow the right wing of the owl.	
(2) The Lord: Hello Airwriter, you will come to heaven if you win this game. If not, you will burn in hell - sorry.. there is not enough space for everyone in heaven. But don't worry, your next quest is simple: say a prayer at the next visible church!	
The Lord	**The Devil**
(3a) The Lord: Good prayer! you will be an angel soon. but first, go to the Karlsplatz church and pray again!	**(3b)** The Devil: stop praying for the lord and don't go to Karlsplatz church but find some lost souls for me. The next is.. Hm.. Just 3 mins away!
(4a) The Lord: Good prayer! you will win this game as an angel, I will give you the first part of your password: holy - but to win, you have to finish your last quest: find a lost wounded soul in the park!	**(4b)** The Devil: you have found a lost soul. Yeah! Bring it to Naschmarkt! I will give the first part of your password: hell.
(5a) The Lord: you found the wounded soul! Bring it back to the owl! The second part of the password is soul. Run back to the owl.	**(5b)** The Devil: My well obeying servant! Thank you for this soul. Your second part of the password is fire. Hurry back to the owl!
(6a) Password: holysoul. Good prayer! You won the game. I will give you wings to fly!	**(6b)** Password: hellfire. Hahahah! you won the game! come and join me in hell!

Table 8. The Scavenger Hunting Game

Gameplay Scenario

Maria and Judas are students at the technical university of Vienna. They both have heard of a location based game close to their university published at airwriting.com and are interested in playing it. They visit the page with their mobile clients, download and install the client software immediately. Both of their mobile phones have a GPS-sensor and Internet connection, therefore they can just start to play and do not care about Internet connection costs. After the launch of the airwriting client software, a group 'Scavenger Hunt TU Vienna' becomes visible. They join it and receive 3 messages. Message (1) as the welcome and unlocked message and two locked messages (6a and 6b) they cant read. Both start following the 'right wing of the owl' order. After some meters, they come to the stone statue and receive a message (2). Maria says to Judas: 'This is easy the church is straight away!' They keep on moving and receive two messages at the church (3a and 3b). Maria decides to pray further, and Judas to find the lost soul. Judas is looking around and receives a warning 'border fence is reached' and turns back to the church. He takes another route and gets the next message (4b) and with it the first part of the password 'hell'. At the same time, maria is arriving at Karlsplatz church and also getting her first part of the password 'holy' (4a). Maria is told to look for a soul. She starts to walk around again.. Judas arrives at 'Naschmarkt' and receives his last order and part of the password 'fire' (5b). At the same time, maria is finding the soul (5a). Both start running back to the owl. Maria is first, and unlocks one of the two locked messages and wins.

6.2 Experimental Design and Analysis

Procedure

The game Scavenger Hunt was tested at Karlsplatz in Vienna with 20 users within a time period of 10 ten days. Each test was designed to take a maximum of one hour. At the start, the idea and application was explained (5 mins). After this, the users got a pre-configured mobile device (Nokia N95, Nokia E50) with external GPS devices and started to play at the owl. All important activities were logged on the client and server. After the test, the log from the client was sent to the server by the test instructor manually via HTTP to the server. Finally, the users were told to shared their experiences to the test instructor. A short survey with about 20 questions was later sent out by mail.

Participants

The participants in this study were sampled from a group of students between 20 and 27 years, 2 entrepreneurs at the age of 36 and 29 and two older participants at the age above 50. All of them are using the Internet and their mobile phone regularly.

Results

19 of the 20 tested users (95%) would use airwriting if it would be available on their mobile phone, even as a permanent background application (88%). The satisfaction factor for the usability of the alpha version is moderate to good. 80% liked the alpha prototype but 29% had sometimes problems in using it. Most testers would use it both for gaming and writing messages (62%), only 8%

would use it just for gaming purposes. Another 30% would just write with it. 97% of the users would use it both on the the client and the online platform, the rest just on the client. We also asked the participants to quantify the monetary worth of the security/privacy aspect (which airwriting implements) by asking; 'How much would you pay for the security aspect?' They maximum values were EUR 150 per year, per month EUR 2, EUR 0.3 per message. The lowest values were EUR 0.3 per message and some of them stated that they would prefer a voluntary donation system (2 people). The large degree of variation in the perceived personal value of such services is based on a wide variety of factors and would need to be analyzed in more detail when creating a charging/costing system.

We also asked them if they had any ideas for attributes, attribute mixtures or groups. The most interesting answer was a feature which realizes mutually exclusive messages, (therefore, if a user is reading message A on place X its not possible anymore to read message B on place Y). The prefered times for setting up an individual group were 5 minutes (31%), 10 minutes (54%), 20 minutes (8%), the rest deciced for the optional answer that the time depends how funny it is creating a group.

The analyzed log files showed several things; Firstly, that more than 50% of the users have restarted the airwriting client at least once. Whereby the maximum restart value was 4. The reason for restarting the client that often was not due to errors in the software, but due to the exit button being next to the option button of the N73 model, which was pressed by accident. Secondly, that the average time for getting messages was 2.665 (standard deviation 2.699) seconds. Thirdly, that the users were switching to different views about 32,41 times (standard deviation 22,63). Because there are only 7 views (Menu, Login, Write, Read, Options, Setup, Exit) this value is rather high and was not expected. For each view switch, approximately 4,5 clicks are needed. This means, that 144 clicks were done on average for view switches by the user. We will take this value as a reference for further improvements.

7 Conclusion

7.1 Summary

This master-thesis describes the concept and a working implementation of an attribute based architecture named airwriting, where everything of the message becomes an attribute like it's text and location. More than 35 attributes are described in lists which could be used for various purposes like gaming, tourist information services and privacy sensitive applications. More than 60 spatial messaging applications are shortly presented and analyzed in terms of their attributes. It could be shown, that most of them are using only a few, less than 5 attributes and that most of them are using the same subset of attributes (location, user-id, text, group, image). This means that many of these services are very similar, especially the social platforms lack in providing innovative functionalities.

Additionally, privacy aspects were analyzed. Only 10 of 60 projects allow you to set the visibility status of the user. This means, that only these 10 projects allow to set the visibility status to "hide" or similar states supporting the user's privacy. More than a half, exactly 31 projects had no privacy-aggreement on their website.

The implementation of airwriting was tested with 20 users with a location based game on 3 different platforms (j2me, iPhone, Android). The results of this initial user study indicate that the concept of the architecture seems clear and accessible, even if the 6500 analyzed server and client logs show that there is place for many improvements in terms of design, implementation and in terms of user experience.

7.2 Contribution

This master-thesis contributes

- the concept of an attribute based architecture for designing new and innovative messaging services.
- various attributes which could be used for such messaging service.
- one implementation of an attribute based architecture for three different platforms (j2me, android, iPhone). This example may help in understanding the concept more practically.
- a usability study with results indicating that the concept is understandable.
- a short presentation and evaluation of more than 60 messaging services.

7.3 Outlook

Empirical studies will prove if the concept "attribute based messaging" is helpful and even successful as a further model for designing and implementing mobile messaging services. For testing and evaluating this concept, researchers, application developers, and users are invited to contribute in suggesting attributes,

creating combinations, groups, applications and content (messages themselves). Hence they are invited to design, suggest and implement third party applications which use the concept of attribute bases messaging as it's underlaying model or airwriting services itself. Also, compatibility issues between different attribute based architectures should be considered, implemented, evaluated and therefore tested in practice.

If the concept of attribute based messaging turns out to be useful in the future, taxonomy issues, more detailed and more extensive user studies with multiple iterations of user interface prototypes and to further quantify privacy protection mechanisms and their respective costs are planned and more different platforms (windows mobile, blackberry, palm, symbian) would be targeted.

List of Figures

List of Tables

References

1. Mayrhofer R., Sommer A., S.S.: Airwriting: A Platform for Private, Mobile, Spatial Group Messaging. (April 2009)
2. Saral, S.: Airwriting: A Platform for Private, Mobile, Spatial Group Messaging. Master's thesis, Technical University, Austria, Vienna (April 2009)
3. : Nielsen web page. `http://blog.nielsen.com/nielsenwire/` (2009)
4. Martin, M.J.C.: Managing Innovation and Entrepreneurship in Technology-Based Firms. 1 edn. Wiley-Interscience (September 1994)
5. Kolodovski, A.: Push - pull - thinking, Roskilde , Denmark, Riso National Laboratory (March 2006)
6. Counts, S.: Group-based mobile messaging in support of the social side of leisure. Comput. Supported Coop. Work **16**(1-2) (2007) 75–97
7. Ballagas, R., Kratz, S.G., Borchers, J.O., Yu, E., Walz, S.P., Fuhr, C.O., Hovestadt, L., Tann, M.: REXplorer: a mobile, pervasive spell-casting game for tourists. In Rosson, M.B., Gilmore, D.J., eds.: Proc. CHI 2007, ACM (April 2007) 1929–1934
8. Rahemtulla, H.A., Haklay, M., Longley, P.A.: A mobile spatial messaging service for a grassroots environmental network. J. Locat. Based Serv. **2**(2) (2008) 122–152
9. D'Roza, T., Bilchev, G.: An overview of location-based services. BT Technology Journal **21**(1) (January 2003) 20–27
10. Burrell, J., Gay, G.: E-graffiti: evaluating real-world use of a context-aware system. Interacting with Computers **14**(4) (2002) 301–312
11. Persson, P., Espinoza, F., Fagerberg, P., Sandin, A., Caester, R.: Geonotes: A location-based information system for public spaces. In: Readings in Social Navigation of Information Space. (2002)
12. Froehlich, P., Simon, R., Muss, E., Stepan, A., Reichl, P.: Envisioning future mobile spatial applications. In: People and Computers XXI, British HCI (2007)
13. Persson, P., Fagerberg, P.: Geonotes: a real-use study of a public location-aware community system. Technical Report T2002-27, Swedish Institute of Computer Science
14. Rantanen, M., Oulasvirta, A., Blom, J., Tiitta, S., Mantyla, M.: Inforadar: group and public messaging in the mobile context. In: Proceedings of the Third Nordic Conference on Human-Computer Interaction. Messaging (2004) 131–140
15. Deriaz, M., Seigneur, J.M.: Trust and security in spatial messaging: Foxytag, the speed camera case study. In: PST. Volume 380 of ACM International Conference Proceeding Series., ACM (2006) 20

16. Stephan A. Drab, G.B.: Spacerace – a location based game for mobile phones. In: Proceedings of PerGames. (2005)
17. Wobbrock, J.: The future of mobile device research in hci. In: CHI 2006 Workshop Proceedings: What is the Next Generation of Human-Computer Interaction? (April 2006) pp. 131–134.
18. Hassanein, K., Head, M.M.: Ubiquitous usability: Exploring mobile interfaces within the context of a theoretical model. In Eder, J., Mittermeir, R., Pernici, B., eds.: CAiSE Workshops. Volume 75 of CEUR Workshop Proceedings., CEUR-WS.org (2003)
19. Kane, S.K., Wobbrock, J.O., Smith, I.E.: Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces. In ter Hofte, G.H., Mulder, I., de Ruyter, B.E.R., eds.: Mobile HCI. ACM International Conference Proceeding Series, ACM (2008) 109–118
20. Gong J., T.P.: Guidelines for handheld mobile device interface design. In: Proceedings of the DSI 2004 Annual Meeting. (2004)
21. Nikkanen, M.: One-handed use as a design driver: Enabling efficient multi-channel delivery of mobile applications. In Crestani, F., Dunlop, M.D., Mizzaro, S., eds.: Mobile HCI Workshop on Mobile and Ubiquitous Information Access. Volume 2954 of Lecture Notes in Computer Science., Springer (2003) 28–41
22. Weiss, S.: Handheld usability. In: Symposium on Designing Interactive Systems. (2002) 412
23. Bevan, Nigel: Serco Usability Services. Serco Usability Services, United Kingdom, London (2000-12-27) 44-20-7421-6471.
24. Wallace, P., A., H., D., S., Z., B., K., B.: i-mode developer's guide. (2002)
25. : Nokia mobile internet toolkit xhtml guidelines. `http://www.forum.nokia.com` (2002)
26. Singhal, S.: WAP — The Wireless Application Protocol: writing applications for the mobile Internet. Addison-Wesley, pub-AW:adr (2001)
27. Pearrow, Mark: The Wireless Usability Handbook. Charles River Media (2002)
28. Jones, M., Buchanan, G., Thimbleby, H.: Sorting out searching on small screen devices. Lecture Notes in Computer Science **2411** (2002)
29. Buchanan, G., Farrant, S., Jones, M., Thimbleby, H.W., Marsden, G., Pazzani, M.J.: Improving mobile internet usability. In: WWW. (2001) 673–680
30. Jones, M., Marsden, G., Mohd-Nasir, N., Boone, K., Buchanan, G.: Improving web interaction on small displays. Computer Networks **31**(11-16) (1999) 1129–1137
31. Kaasinen, E., Aaltonen, M., Kolari, J., Melakoski, S., Laakko, T.: Two approaches to bringing internet services to WAP devices. In: Proceedings of the Ninth International World-Wide Web Conference. (2000)
32. Kaikkonen, Anne, Roto, Virpi: Navigating in a mobile XHTML application. In: Proceedings of ACM CHI 2003 Conference on Human Factors in Computing Systems. Volume 1 of Designing applications for handheld devices. (2003) 329–336
33. : Postgresql web page. `http://www.postgresql.org/` (2008)
34. : Postgis web page. `http://postgis.refractions.net/` (2008)
35. : Hibernate web page. `http://www.hibernate.org/` (2008)
36. : Spring web page. `http://www.springframework.org/` (2009)
37. : Tapestry5 web page. `http://tapestry.apache.org/tapestry5/` (2008)