



Friends Radar: Towards a private P2P location sharing platform

Rene Mayrhofer and Clemens Holzmann

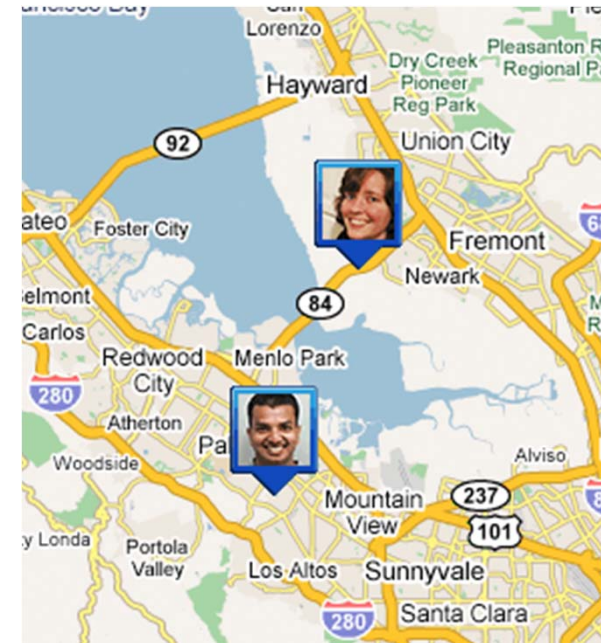
Workshop MCPT 2011, 2011-02-07 15:00-15:30, Las Palmas, Gran Canaria

Social location sharing



Where are you?

- The most often asked first question during a mobile phone call
- Location sharing applications are becoming popular
 - > Google Latitude
 - > Facebook Places
 - > and many more ...



Places

Who. What. When. And now **Where.**

Characteristics of Current Approaches



Centralized

- One server (farm/cluster)
 - > Simple for clients
- One (virtual) database

Infrastructure based

- Need (live) connectivity between client and server, typically UMTS

(Mostly) Map based

- Optimized for „far-field“ usage
- Existing Augmented Reality (AR) mobile phone applications for static objects

Main Issue: Privacy

All clients use one (virtual) server

- Location updates pushed to server
- Friends queries either pushed by or pulled from server
- Server keeps track of all registered users and their reported locations

Did you read the ToS?

- „By **submitting**, posting or displaying the content you give Google a **perpetual, irrevocable, worldwide, royalty-free**, and non-exclusive license to reproduce, adapt, modify, translate, publish, publicly perform, publicly display and distribute **any Content** which you submit, post or display on or through, the Services.
..... You agree that this license **includes a right for Google to make such Content available to other companies**, organizations or individuals with whom Google has relationships for the provision of syndicated services, and to use such Content in connection with the provision of those services.”
(Google Terms of Service, <http://www.google.com/accounts/TOS>, 2011-01-31)

Main Issue: Privacy

Did you read the ToS?

- „When you access Facebook from a computer, mobile phone, or other device, we may collect information from that device about your browser type, location, and IP address, as well as the pages you visit.
..... We occasionally pair advertisements we serve with relevant information we have about you and your friends to make advertisements more interesting and more tailored to you and your friends.
..... We **share your information** with third parties **when we believe** the sharing is permitted by you, **reasonably necessary** to offer our services, or when legally required to do so.
..... **Even after you remove information from your profile or delete your account, copies of that information may remain** viewable elsewhere to the extent it has been shared with others, it was otherwise distributed pursuant to your privacy settings, or it was copied or stored by other users. However, your name will no longer be associated with that information on Facebook.”
(Facebook Terms of Service, <http://www.facebook.com/policy.php>, 2011-01-31)

Privacy vs. Centralized Architecture



Centralized approach

- Need to trust **current and future** service, company, operators, administrators, country that hosts the service, specific software implementation, etc.
- If any party defaults, private data will be gone forever (storage is cheap!)
- Note: centralized hub for message passing would be possible with encrypted location messages, but not if centralized processing is part of the approach

Decentralized approach

- Trust you own mobile phone (bugs/exploits still possible and likely)
- Trust your chosen server operator or infrastructure provider to transport data
 - If transmission is encrypted, don't have to trust privacy policy

Friends Radar

Decentralized

- Client software on mobile devices
 - > all data storage and all processing done on clients
 - > location updates are pushed selectively to „friends“
- XMPP instant messaging infrastructure for data transmission
 - > either (decentralized) server infrastructure
 - > or direct, infrastructure-less peer-to-peer messaging possible (XMPP link-local messages)

Multiple Views

- Map based for „far-field“ usage
- Augmented Reality (AR) for „near-field“ usage
- Integrated compass view for quick awareness (near or far)

Architecture: XMPP



- XMPP is a standard protocol for instant messaging (and a lot more), previously called Jabber (and still used in projects/products)
- based on federated server infrastructure
 - everybody can install an XMPP server without registering with any central authority
 - servers communicate among each other (TCP port 5269)
 - clients can connect directly to any server (TCP port 5222) with optional TLS transport security
 - gateways to other protocols available

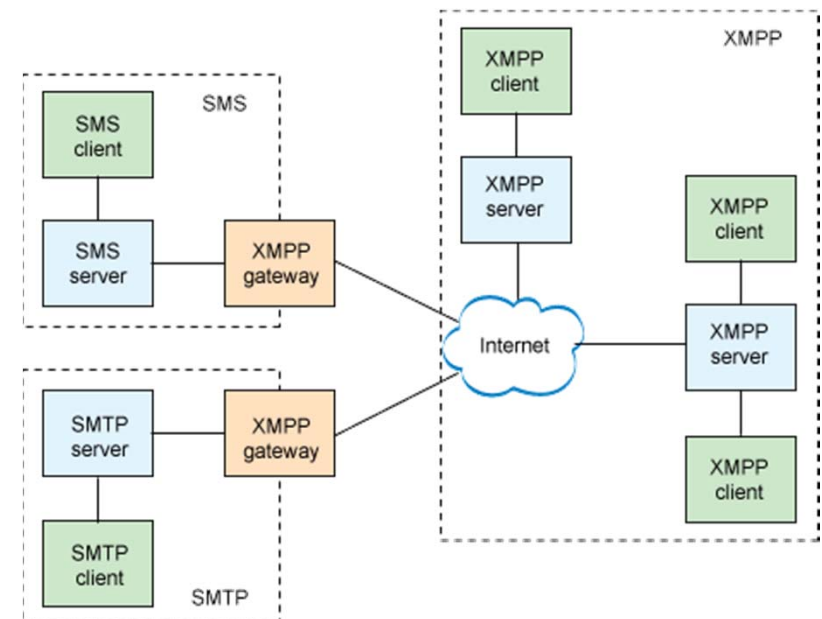


Diagram from <http://www.ibm.com/developerworks/webservices/library/x-xmppintro/index.html>

Architecture: Friends Management



Friends management in Friends Radar

- relies on standard XMPP „roster“ functionality
- list of contacts („Friends“) stored on XMPP server
- using XMPP contact groups (also stored on server) to define if location updates are pushed to and displayed for specific contacts



Conceptually problematic

- pushing location updates **to** a contact is different from displaying location updates received **from** a contact
- might want to select independently, but more options for user (understandability issue)
- vn current version: pushing location updates iff displaying contact's location

Comparison to Centralized Approaches



Advantages

- Privacy issues
 - > level of required trust significantly reduced
 - > no single point of data recording
 - > friends can still record location updates
- Flexibility
 - > encryption end-to-end or on transport level
 - > centralized or decentralized message transmission possible
 - > infrastructure-less possible

Disadvantages

- Performance
 - > overhead in terms of number of messages
 - > more processing and data storage on (limited) client
- Data aggregation not (easily) possible
 - > e.g. number of people attending an event
 - > e.g. optimizing routes based on traffic as additional service
 - > possible with decentralized, but large overhead

Preliminary performance analysis



Parameters for worst-case analysis

- n = number of friends, assuming fully connected network
- f = number of location updates / time
- s = average message size

Parameter for best-case analysis

- m = average number of clients with „interesting“ location updates (e.g. filtered by location, close vs. far, etc.)
- for worst-case analysis: $m=n$

Preliminary performance analysis



Centralized location server

- Client point of view
 - > client -> server: $f*s$
 - > server -> client: $(n-1)*f*s$
 - > all messages: $n*f*s$
- Message point of view (n clients)
 - > n^2*f*s
- best case client PoV
 - > client -> server: $f*s$
 - > server -> client: $m*f*s$
- best case message PoV
 - > $n*(m+1)*f*s$
 - > for $m \ll n$ (or m constant)
 - $O(n)$

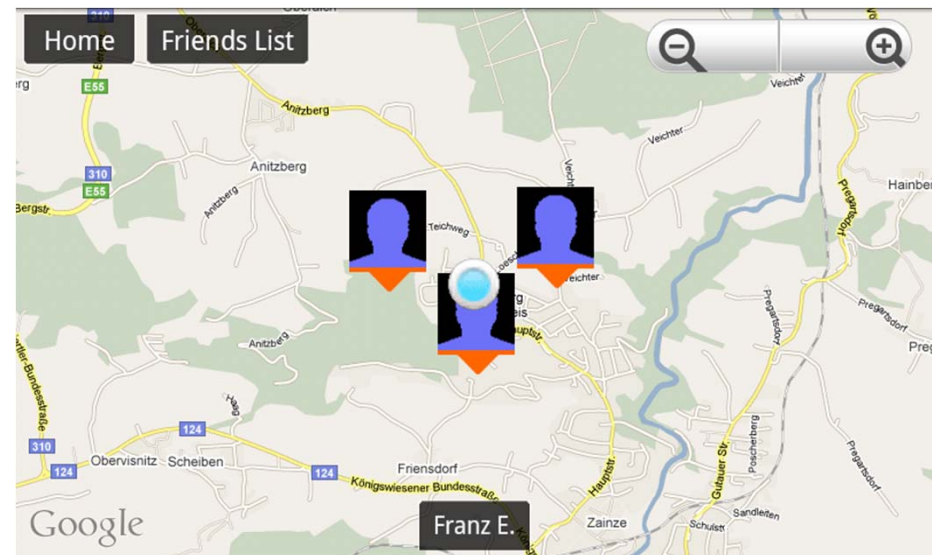
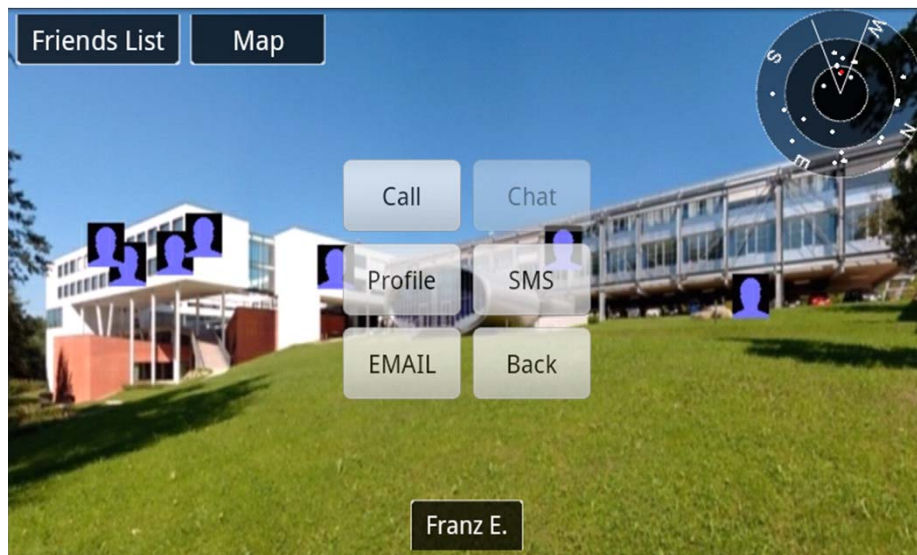
Friends Radar

- Client point of view
 - > client -> server: $(n-1)*f*s$
 - > server -> client: $(n-1)*f*s$
 - > all messages: $2*(n-1)*f*s$
- Message point of view (n clients)
 - > $n*f*s$ in fully connected network
 - > $2n*(n-1)*f*s$ with central message hub
- best case message PoV
 - equivalent to worst case because filtering can not be done in messaging hub but at each recipient
 - $O(n^2)$ with messaging hubs

Friends Radar Prototypes

Implementations

- Android (beta version available in the Android Market)
- bada (feasibility study)



Next Steps

Client platforms

- Additional implementations
 - > tablets
 - > iOS
 - > Windows Phone 7
- Additional sensors
 - > using gyroscopes for improved Augmented Reality view

Message transmission

- (Optional) end-to-end encryption over XMPP PubSub infrastructure
 - > encrypting location updates with multiple public keys
- XMPP link-local transmission over WLAN ad-hoc mode

Next Steps

Interoperability

- Pulling location updates from centralized services for display in the same GUI
 - > e.g. Google Latitude, Facebook Places
- Integration with account meta-data and non-location information from social networks
 - > e.g. display Facebook status message or wall messages in AR view
- Integrating „static“ location data in AR view (e.g.



Thank you for your attention

Contact: **rene.mayrhofer@fh-hagenberg.at**

PGP Fingerprint: 7FE4 0DB5 61EC C645 B2F1 C847 ABB4 8F0D C3C2 4BD

PGP Fingerprint: