# Network Proximity on Practice: Context-aware Applications and Wi-Fi Proximity

Dmitry Namiot

*Abstract*— **This paper introduces several patterns (use cases) for using network proximity in mobile services. All applications presented here deploy Wi-Fi proximity information (in various forms) for either discovering new data for mobile subscribers or for delivering some customized information to them. Our paper presents a whole range of mobile mashups based on the common background. The typical deployment areas for the proposed approach are context-aware and ubiquitous computing applications. Our own practice included Smart City applications and proximity marketing.**

*Keywords*— **context-aware, proximity, Wi-Fi, fingerprint, rules.**

## I. Introduction

Classically, term 'context-aware' [1] describes context as location, identities of nearby people and objects, and changes to those objects. Indeed, most of the authors define context awareness as a complementary element to location awareness. Location serves as a determinant for the main processes and context adds more flexibility with mobile computing and smart communicators [2].

In the same time, there are many practical use cases, where the concept of location can be replaced by that of proximity. At the first hand, this applies to use cases where the detection for exact location is difficult, even impossible or not economically viable [2]. Another reason for such replacement is related to privacy. For example, we can think here about a privacy-aware proximity detection service determines if two mobile users are close to each other without requiring them to disclose their exact locations. Proximity can be used as a main discovery element for context-aware browser [3]. In this concept, any existing or even especially created Wi-Fi hot spot or Bluetooth node could be used as presence sensor that can play a role of presence trigger. This trigger can open access to some content, discover existing content as well as cluster the nearby mobile users.

A definition for network proximity could be very transparent. It is a measure of how mobile nodes are close or far away from the elements of network infrastructure. In this definition the elements of network infrastructure (e.g., Wi-Fi access points) play a role of landmarks. For some tasks we need to measure how our mobile clients are close each other, using the network infrastructure information only. The classical model uses Bayesian reasoning and a hidden Markov model (HMM) [4]. Classical models took into account not only signal strengths, but also the probability of seeing an access point from a given location. But the biggest practical problem for such models is the mandatory manual calibration phase. Classical Wi-Fi based positioning system works in two phases. Phase 1 is the offline training phase or calibration. In phase 1 a human operator measures the received signal strength indicator (RSSI) from different access points (APs) at selected positions in the environment. This phase creates a radio map that stores the RSSI values of access points at different fixed points. And only phase 2 is the real location estimation. And of course, any changes in the network (network's infrastructure) cause recalibration process. Another problem that could not be solved with calibration (Wi-Fi fingerprints collection) is support for dynamic location based system. For example, Wi-Fi access point could be opened right on the mobile. In this case network infrastructure elements could move. And data linked to them should move too. It is a practical example for dynamic location based systems.

Some of the systems (e.g., [5]) can use a formula that approximates the distance to an access point as a function of signal strength. Using an optimization technique, such systems compute location to an accuracy of about 10 meters using signal strengths from multiple access points. The above-mentioned Spot Expert system (SpotEx) [2] is a good example of the empirical approach to Wi-Fi proximity. It is based on the logical rules (productions) and does not require calibration phase. It is the main goal for our systems – use network proximity without the preliminary scene preparation.

The rest of the paper is organized as follows. Section II describes our SpotEx approach. In Section III we describe context-aware check-ins. In Section IV we discuss our Local Messaging platform. In section V we present context-aware QR-code reader.

## II. Spot Expert

SpotEx service presents a new model for context-aware data retrieval. The model uses only a small part from algorithm of any Wi-Fi based positioning system - the detection of Wi-Fi networks. Due to the local nature of Wi-Fi networks, this detection already provides some data about location, namely information about proximity. As the second step, SpotEx introduces an external database with some rules (productions or if-then operators) related to the Wi-Fi access points. Typical examples of conditions in our rules are: AP

Manuscript received May 24, 2013.

D. Namiot is with the Faculty of Computational Mathematics and Cybernetics Lomonosov Moscow State University, Moscow, Russia (email: dnamiot@gmail.com).

with SSID Café is visible for mobile device; RSSI (signal strength) is within the given interval, etc. Based on such conclusions, we then deliver (make visible) user-defined messages to mobile terminals. In other words, the visibility of the content depends on the network context (Wi-Fi network environment).

The SpotEx model does not require a calibration phase and is based on the ideas of proximity. Proximity based rules replace location information, where Wi-Fi hot spots work as presence sensors. The SpotEx approach does not require mobile users to be connected to the detected networks; it uses only broadcast SSID for networks and any other public information.

Technically, SpotEx contains the following components:

- Server side infrastructure including a database (store) with productions (rules), rules engine and rules editor. The rules editor is a web application (it supports mobile web too), that works with the rules database. The rules engine is responsible for runtime calculations. Note that the database is currently located outside the mobile device, but it could be positioned on the device too. For example, in Wi-Fi Direct environment this database could be saved right on the mobile device.

- The mobile application is responsible for getting context information, matching it against the database with rules and visualizing the output.

SpotEx could be deployed on any existing Wi-Fi network (as well as networks especially created for this service) without any changes in the infrastructure. Rules can easily be defined for the network. Data chunks (previously termed messages or conclusions for rules) here are simply text pieces that should be opened (delivered) to the end-user's mobile terminal as soon as the appropriate rule is fired. For example, as soon as one of the networks is detected via the mobile application. Here is a typical rule:

IF *IS_VISIBLE*('mycafe') AND *FIRST_VISIT()* THEN
{ present the coupon info }

The term "delivered" here is a synonym for "being available for reading/downloading". For end-users the whole process looks like automatic (and anonymous) check-in.

Existing use cases target proximity marketing in the first phase. The whole process looks like an "automatic check-in" (by analogue with Foursquare, etc.) One shop can deliver proximity marketing materials right to mobile terminals as soon as the user is near some selected access point. Rather than checking-in (manually or via an API) at a particular place and getting back information about special offers (similar to Foursquare, Facebook Places, etc.), mobile subscribers can collect deals information automatically with SpotEx. The prospect areas, in our opinion, are information systems for campuses and hyper local news delivery in Smart City projects. Rules could be easily linked to the available public networks [6].

## III. CONTEXT-AWARE CHECKINS

In general, checking rules (conditions) in SpotEx looks like a special check-in service for social networks. What is a typical check-in record in social networks? It is some message (post, status) linked to some location (place). What are the reasons for members in social networks use such special kind of messages? Sometimes it could be stimulated by the business. Practically, user posts advertising for the business in exchange for some benefits. Sometimes it could be used for social connection too. Check-in record lets other members discover publisher's location. For publisher, the same record lets discover nearby friends.

But the key point is the special kind of record in the social network – check-in. It could be customized of course. Business can create own forms for check-in records [7], but they are still some posts in the social networks. In other words, they are always part of the social stream.

What if we create a new type of check-in records and separate them from rest of stream? It means that we will provide a separate database with a list of accounts from the social network being concentrated (at this moment only!) nearby some place. It is a temporal database, check-in records could be changed constantly. This database does not contain the social stream itself. It contains just ID's (e.g., nick names) for accounts.

The second key moment is a new definition for places. Traditionally, for social networks it is a pair from geo coordinates (latitude, longitude) and some description. It is already creates problems for indoor (many places within the big mall will be on the same geo position, etc.). As per new approach, our "new" places should be removed from the social networks too. We will describe our "places" separately and define them via proximity (network proximity in our case) attributes rather than via geo coordinates.

Proximity based definition means that each place should be defined via some metric, introduced for our network. Let us talk about Wi-Fi. As a base for metric we can use visible SSID for networks, counter for access points and RSSI (signal level) for the each device. On practice, it is so called Wi-Fi fingerprint.

Two users (mobile phones) are in the proximity if they have at least one common visible access point. And obviously, the more similar (close each other) the visible network environments are, than the level of proximity is bigger.

Now, suppose we have a mobile application that lets for users confirm their social network identity and link that identity with wireless networks info. This wireless info is simply the same Wi-Fi fingerprint we've described above for SpotEx. Our application will form (fill) our external temporal database that contains social network identity confirmation and an appropriate network info (Wi-Fi fingerprint). Running this application is simply an act of performing new form of check-in. This new check-in is an "external" entity for the social network. Our application does not post data back to the social network. It keeps data outside (in the own database). So, in the terms of privacy, this check-in does not affect (does not touch, actually) account's settings in the social network. In the same time the real ID from the existing social networks let us propose discovery mechanism for the social graph. It simply answers to the question: how to get know about new members outside of your social graph? The build-in check-ins

discovers places for our existing friends. External check-ins database discovers potential new friends. What are the reasons for users perform new check-ins? Actually, they are the same as for "old" check-ins. Business entity can use that information for statistics and deliver some benefits in exchange for "check-in". Of course, it could be used for creating the connections. Check-ins let other know where I am as well as see other people nearby.

We can provide an interesting set of new services with this external check-ins database. At the first hand, we can list other people at any particular location. Actually, it is always a list of people at "this" location only. It follows from the fact, that our proximity based system does not provide a list of locations in the traditional form. Each our "location" described via Wi-Fi sensors. In other words, it is described via visible access points and RSSI. Obviously, all the attributes are dynamical. So, after own check-in, user can see only nearby check-ins. In this approach user is simply unaware about other "places" unless he moves and check-in again.

At the second, we can show (search) social streams nearby. Via public API we can read data feeds for users (if it is possible, of course, and an appropriate stream is not protected). This system can keep the full respect to the existing privacy settings in social networks.

## IV. Local Messaging

Local messaging presents a mashup from passive Wi-Fi monitoring and Cloud messaging for smart phones. Passive monitoring uses so called Wi-Fi probe requests. It is one of the fingerprints-less models, sniffing for beacon frames. Wi-Fi client periodically broadcasts an 802.11 Probe Request frame. Client expects to get back an appropriate probe request response from Wi-Fi access point. As Wi-Fi spec, a station (client) sends a probe request frame when it needs to obtain information from another station. For example, a radio network interface card would send a probe request to determine which access points are within range. A Probe Request frame contains two fields: the SSID and the rates supported by the mobile station. Stations that receive Probe Requests use the information to determine whether the mobile station can join the network [8].

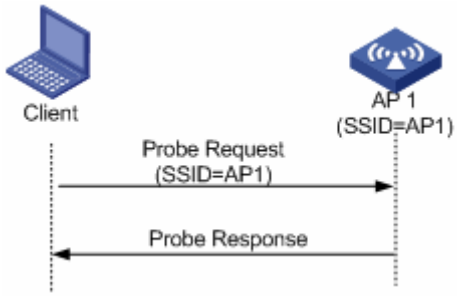


Figure 1. Probe request

Technically, probe request frame contains the following information:

- source address (MAC-address)
- SSID
- supported rates and additional request information
- vendor specific information

Our access point can analyze received probe request. Obviously, that any new request (any new MAC-address) corresponds to a new wireless customer nearby. The client does not require load specific applications, or run specific applications. There are commercial of-the-shelf components that can provide passive Wi-Fi monitoring. For example, it is Navizon I.T.S. [9]

Cloud Messaging for mobile platforms is a service that allows you to send data from your server to your users' device. For example, Google Cloud Messaging (GCM) service handles all aspects of queuing of messages and delivery to the target Android application running on the target device. Apple Push Notification Service (APN) is a robust and highly efficient service for propagating information to devices such as iPhone, iPad, and iPod touch devices.

Architectures of these push notification services have common features. Any application must register to receive push notifications. It typically does this right after it is installed on a device. For example, iOS receives the registration request from an application, connects with APNs, and forwards the request. APNs generate a device token using information contained in the unique device certificate. The device token contains an identifier of the device. It then encrypts the device token with a token key and returns it to the device. This token identifies the subscriber and later will be used for sending push notifications.

Here comes our idea to mashup. During the registration phase we can collect MAC-address too. Our database will save the token and an appropriate MAC-address for every subscriber. Later, we can compare MAC-address with addresses, collected by our passive monitoring and distribute push messages to nearby subscribers. So, it is a classical mashup that looks for the intersections in two databases (subscribers and nearby visitors) [10].

Note, that because we are using MAC-addresses for the re-identification only, we can save in our database some hash-code instead of the real address. This trick lets us keep the privacy.

## V. Context-aware QR-code

Encoding for URLs is, probably, the most widely used option for QR-codes. Mobile users can scan QR-code and go to the URL (mobile site) mentioned there.

QR-code scanning is always performed by the mobile application, right on the mobile phone. It means that this application technically can obtain the context information during the scanning. For example, this application can perform the same task the above-mentioned SpotEx is doing before rules check. Application can calculate Wi-Fi fingerprint. In other words, we can get the list of the visible Wi-Fi access points as well as signal strength info.

Here comes our idea. We can simply add this information to the URL decoded from QR-code picture. This context information should be added automatically. Our new

context-aware QR-code scanner can add Wi-Fi data as HTTP GET parameters to the decoded URL (in other words, add them to query string).

The advantages for this approach are obvious. For example, in some indoor positioning system, based on OR-codes, we can place identical QR-code everywhere. All the location related information will be passed to the decoded URL automatically. We can implement the above mentioned SpotEx system for mobile users scanned QR-codes. Additionally, this approach let us collect Wi-Fi data (statistics) [11].

The technical implementation (for Android platform) is based on the Open Source QR-code reader Zxing [12]. The final goal is to create SpotEx implementation for QR-codes. And of course, the possibility to add more context information to QR-code scanner is open.

## VI. Conclusion

This paper describes several mobile services based on the common approach to Wi-Fi proximity. The proposed model uses smart-phone as a proximity sensor. Described services can use existing as well as the especially created networks nodes as presence triggers for delivering and discovering new content for mobile subscribers. Proposed services could be used for proximity marketing and delivering commercial information (deals, discounts, coupons) in malls, for access to hyper-local news data and for data discovery in Smart City projects.

## References


[1] G. Schilit and B. Theimer Disseminating Active Map Information to Mobile Hosts. IEEE Network, 8(5) (1994) pp. 22-32

[2] D. Namiot “Context-Aware Browsing -- A Practical Approach”, Next Generation Mobile Applications, Services and Technologies (NGMAST), 2012 6th International Conference on, pp. 18-23, DOI: 10.1109/NGMAST.2012.13

[3] D. Namiot and M. Sneps-Sneppe. “Wi-Fi proximity as a Service.” In SMART 2012, The First International Conference on Smart Systems, Devices and Technologies, pp. 62-68.

[4] Ladd, A.M., et al. Robotics-Based Location Sensing using Wireless Ethernet. In Eighth International Conference on Mobile Computing and Networking. 2002. Atlanta, GA, USA

[5] Griswold, W.G., et al., ActiveCampus - Sustaining Educational Communities through Mobile Technology. 2002, University of California, San Diego: La Jolla. p. 19.

[6] D. Namiot and M. Sneps-Sneppe “Context-aware data discovery”, Intelligence in Next Generation Networks (ICIN), 2012 16th International Conference on, pp. 134-141, DOI: 10.1109/ICIN.2012.6376016

[7] D. Namiot and M. Sneps-Sneppe Customized check-in Procedures Smart Spaces and Next Generation Wired/Wireless Networking Lecture Notes in Computer Science, 2011, Volume 6869/2011, pp. 160-164, DOI: 10.1007/978-3-642-22875-9_14

[8] M.Gast 802.11 Wireless Networks: The Definitive Guide O'Reilly Media, Inc., 2005, 654 p.

[9] Navizon I.T.S. http://its.navizon.com/doc/index.html Retrieved: Jan, 2013

[10] D.Namiot. "Local Area Messaging for Smartphones." International Journal of Open Information Technologies vol. 1, N.2 (2013), pp. 8-11

[11] D.Namiot and M. Sneps-Sneppe. "Mobile Services and Network Proximity." arXiv preprint arXiv:1305.4348 (2013).

[12] Zxing http://code.google.com/p/zxing/ Retrieved: Feb, 2013