Context-Aware Data Discovery

Dmitry Namiot
Lomonosov Moscow State University
Faculty of Computational Math and Cybernetics
Moscow, Russia
e-mail: dnamiot@gmail.com

Manfred Sneps-Sneppe
Ventspils University College
Ventspils International Radio Astronomy Centre
Ventspils, Latvia
e-mail: manfreds.sneps@gmail.com

Abstract— **This paper discusses context-aware data browsing and data retrieval for mobile subscribers. We describe existing models as well as provide a new description for our SpotEx approach. Our model for context-aware data discovery uses mobile phones as proximity sensors. In our concept, any existing or even specially created wireless network node could be used as a presence sensor that can open (discover) access to some dynamic or user-generated content. The content itself could also be linked to social media. An appropriate mobile service (context-aware browser) can present that information to mobile subscribers. Potential use-cases for the proposed approach include any project associated with hyper-local news data. For example, projects providing Smart City data, delivering indoor retail information, etc.**

Keywords-context-ware computing; Wi-Fi; proximity; collaborative location;browsing;data mining.

I. INTRODUCTION

In the paper that introduced the term 'context-aware' for the first time, Schilit and Theimer [1] describe context as location, identities of nearby people and objects, and changes to those objects. Other 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. Context awareness is a term from ubiquitous (pervasive) computing and describes deals with linking changes in the environment.

Day [2] defines context as any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the applications themselves. Probably, it is a more developer-oriented definition.

Hristova [3] states that context-related information can consist of user profiles and preferences, their current location, the type of connection to the mobile network, the type of wireless device being used, the objects that are currently in the user's proximity, and/or information about their behavioral history.

Modern applications adopt a context-aware perspective to manage:

a) Communication among systems and among users, or between the user and the system

b) Situation-awareness, like modeling the physical situation (environment aspects) and location or user's personal situation (current activity)

c) Knowledge chunks: determining the set of situation-relevant information, services or behaviors [4].

In this article, we deal with context-aware knowledge chunks (case c). In our definition each chunk is simply some user-defined data fragment, associated with the context.

The basic element for all the above mentioned definitions is location. In general, getting location information for mobile subscribers can be pretty standard nowadays (GPS, cell-id, assisted GPS [5]). The picture is much more complicated for indoor positioning. The Global Positioning System (GPS) loses accuracy indoors [6]. For indoor positioning systems (IPS) we see a wide variety of design approaches. In practice, IPS systems can use various optical, radio, or even acoustic technologies. However, all of them require the utilization of their own protocols with their own API. Heterogeneous scenarios are typical for IPS.

Wi-Fi based positioning is one of the most used cost-effective approaches to indoor location. It is a completely software-based solution that can utilize existing Wi-Fi access points installed in a facility and Wi-Fi modules already present in the user devices. Wi-Fi based IPS can provide meter level accuracy, which should be sufficient for most location-aware applications [7].

An interesting approach is cooperative location (CL). The Cooperative Location-sensing system (CLS) is an adaptive location-sensing system that enables devices to estimate their position in a self-organizing manner. Classical Wi-Fi based IPS 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. Phase 2 is the real location estimation. With CLS devices can estimate position without the need for extensive infrastructure or training.

Simply put, CLS employs the peer-to-peer paradigm and a probabilistic framework to estimate the position of wireless-enabled devices in an iterative manner without the need for an extensive infrastructure or time-consuming training. [8]

The cost and time saving motivation for CL (CLS) is clear. Cooperative Location-sensing systems are infrastructure-less systems. The peer-to-peer paradigm in CLS means that devices gather positioning information from other neighboring peers. This can take place with any distance estimation method available (for example, using signal strength). CLS updates its positioning estimates from

time to time by incorporating freshly received information from other devices.

Another interesting idea is the development of dynamic location based services. One of the key features here is the ability for users to select services that are specifically associated with their current location. For example, a flexible scope model that defines the association between services and location, as well as a service location infrastructure organized by spatial criteria and optimized for location-based queries [9].

Our own development in this area (SpotEx – Spot Expert [10]) uses proximity ideas for discovering a new context to mobile users. We use proximity as a replacement for location. It seems to us that the final goal (at least for the majority of location based services) is to get data related to the location, rather than the location itself. For example, data feeds about nearby places could be the real goal rather than precise values for latitude and longitude. Location is used as an intermediate result only. It is just a base for subsequent geo-spatial queries. So, why not request data directly if we can estimate the location (get the proximity information)? Simply, in traditional Wi-Fi based indoor-positioning most existing LBS applications follow the schema:

Detecting Wi-FI-nodes > getting location info > getting data for the selected location

Using proximity, we can skip one step, at least:

Detecting Wi-FI-nodes > getting data for the estimated location

II. About context-aware browsers

In general, context-aware data retrieval can be regarded as an extension of classical information retrieval. The objective of incorporating contextual information into the retrieval process is to attempt to deliver information most relevant to the user within their current context [11]. The data retrieval model contains a collection of documents (data chunks), retrieval query, metrics for relevance and the context itself.

User's retrieval requests may be issued automatically or, alternatively, each data chunk in the collection could have a trigger context. When the user's present context matches the context on a data chunk, the chunk (document) is triggered.

The Context-Aware Browser (CAB), introduced in [12], could be described as a mobile browser able to automatically and dynamically load web pages, services and applications selected according to the current context of the user. A CAB itself is a typical context-aware application: a CAB is able to automatically retrieve and constantly update contextual information gathered from the surrounding environment. Initially a CAB can search for specifically tailored applications. But it is important to search "traditional" web pages and applications on the Web also.

The core part of the CAB is the Context server that collects sensor information.

Two approaches have been taken into consideration in developing the inferential system: rule-based systems and Bayesian networks. Rule-based systems have already been used in the context-aware research field [13]. The actual experiments used Bluetooth-enabled phones as sensors. Bayesian networks have been used in the context-aware field mainly as systems to determine the uncertainty of contexts [14].

The biggest problem in the above-mentioned projects is the content description. For example, the original CAB article [12] proposed a large spreadsheet that should be populated with all context data involved in the process. Who will do that and how will it be done? It practically duplicates the amount of work for content-related sites.

Another idea that could be deployed for CABs is service-oriented computing. Service oriented computing assembles application components into a network of services. Services can be loosely coupled to create flexible and dynamic business processes or agile applications [15]. In addition, the orchestration of Web services can support and simplify the exchange of context information in large scale environments, thus enabling Web services systems to utilize various types of context information to adapt their behaviors and operations to dynamic changes. A typical example is Contextserv [16].

III. SpotEx

Originally, the main idea for SpotEx came from indoor positioning. Spotex uses only a first step from algorithm of any Wi-Fi based IPS. We mean 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, we add 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 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 service was first described in an article published in NGMAST-2011 proceedings [17]. SpotEx actually began as a new approach for indoor navigation and migrated to context-ware data discovery later [18]. The current paper describes the redesigned and extended version of SpotEx as well as outlining future plans. The new developments include a completely redesigned logical engine (more formal language, new functions), a module that implements statistics (historical proximity log) and upcoming integration with social networks.

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 (also supporting mobile web) that work 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 also be positioned on the device, e.g. for Wi-Fi Direct.

- 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) here is simply text 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. 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 on 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.

Most of modern smart phones let users open Wi-Fi hotspots right on the phone. We can associate our rules with such mobile hotspots. In this case the messages (data snippets) become linked to the phones. It is, in fact, a typical dynamic LBS. The available services move with the moved phone. Services automatically follow the phone.

Figure 1. Personal Wi-Fi host spot

We think that this use case is the best demonstration for the SpotEx model: the anchor network is defined right on the mobile phone so that the rules could be linked to the mobile phone. A smart phone is all that is needed for creating a new information channel. It is an infrastructure-less approach that doest not need a grid of devices as in CLS models.

Note again, that this approach does not discuss security and connectivity issues. We do not need to connect mobile subscribers to our hotspot. SpotEx is all about using hotspot attributes as triggers in data discovery. The term Wi-Fi proximity is sometimes used in connection with Wi-Fi marketing and in practice simply means setting a special splash screen for hotspots. This splash screen can show some advertising (e.g., branded messages) for users during the connection to that hotspot. SpotExis different as it regards Wi-Fi hotspots simply as presence sensors.

Each rule is a logical production (if-then operator). The conditional part includes the following objects:

Wi-Fi network (SSID, mac-address)
RSSI (signal strength - optionally)
Time of the day (optionally)
ID for the client (mac-address)

In other words it is a set of operators like:

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

Figure 2 presents use case for proximity marketing in retail area.

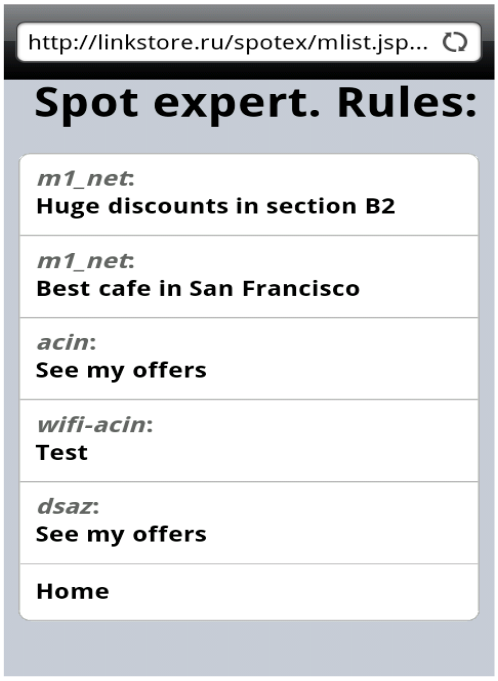

Figure 2. SpotEx rules in proximity marketing application

Because our rules present the standard production rule based system, we can use an old and well know Rete algorithm [19] for the processing. A Rete-based expert system creates a network of nodes. Each node (except the root) corresponds to a pattern presented in the left-hand-side (in the condition) of a rule. The path from the root node to a leaf node defines a complete rule's condition. Each node has a memory of facts, which satisfy that pattern. This structure presents essentially a generalized tree. As new facts are asserted or modified, they propagate along the network of nodes. It causes nodes to be annotated when new fact matches existing pattern. When a fact or combination of facts causes all of the patterns for a given rule to be satisfied, a leaf node is reached. Leaf node triggers the rule [20].

The current implementation for mobile client based on Android OS and uses WiFiManager class from Android SDK. This API let us collect the following information about nearby networks:

SSID - the network name.
BSSID - the address of the access point.
Measurements – frequency, signal level and capabilities. Capabilities present the authentication, key management, and encryption schemes supported by the access point.

All the above-mentioned attributes are the basic elements for our rules. We add to them the standard functionality (like time related operations) as well as historical and statistical functions. It is a formation for conditions. After that we can prepare rules like this:

IF IS_VISIBLE('mycafe')
AND SIGNAL_LEVEL()> -60db
AND TIME_BETWEEN(1pm,2pm)
AND NOT_VISIBLE('myStore')
THEN {present the deals for dinner}

Block {present the deals for dinner} is data (information) snippet for this rule. Each snippet (rule's conclusion) contains a title (text) and some HTML content. This content could be presented as a link to external site, for example. Snippets present news data for campuses, coupons/discounts info for malls, etc.

There are two options for setting content in the rule:

a) provide a link to some external web site/mobile portal.

SpotEx works as some universal discovery tool. Mobile web users become aware about context-relevant web resources. Owners for the web resources can describe own sites via rules rather then present for them individual QR-codes or NFC-tags for example.

b) describe HTML code and let rule-editor automatically create mobile web page for it.

The whole system works as a custom content management system (CMS). SpotEx rule editor creates mobile web pages for data snippets and hosts them on the own server. It means that for presenting our data to mobile subscribers we can use all resources that could be defined on HTML pages, including any multimedia content.

Rule editor works in the both desktop and mobile web.

SpotEx mobile application, being executed, creates dynamic HTML page from titles (according to the rules that are relevant in the given context) and presents that mobile web page to the mobile user. It works just as a classical rule based expert system: matches existing rules against the existing context and makes the conclusions. Existing context here is obtained snapshot for "Wi-Fi environment". In other words, it is a list of hotspots with attributes. Rule engine converts it via Rete algorithm into list of available titles. It is a dynamically created mobile web page too. This final page presents each discovered title as a hyperlink, pointing to the appropriate data snipped. Any click on the provided title opens the snippet (shows or discovers data to mobile user).

It means that for the mobile users the whole process looks like browsing. Mobile browser becomes aware about hyper-local (network info depended) content. It is a typical example of the above-mentioned context-aware data retrieval.

The context-aware retrieval model includes the following elements:

- a collection of discrete documents (base for capturing)
- a set of user's retrieval needs (it is captured in a query)
- a retrieval task(it is responsible for the matching and delivery of documents);

• the user's context. Note, that it could be used for the query formulation as well as for the matching of documents.

As it follows from our description, all the above-mentioned tasks are components of SpotEx.

Originally, SpotEx browser supports "pull model", versus the "push model", proposed by Bluetooth marketing, for example. Using timer-based updates (auto-refresh) for web pages we can simulate "push model" in the user-safety mode.

We can note also, that as any browsing, the whole process is anonymous. Of course, any data snippet may lead to some business web site/portal, where the target site may ask about login, etc., but the SpotEx itself is anonymous. Unlike social networks like Foursquare, users do not need to disclose their identity just for looking deals in the mall, for example.

Despite anonymous browsing, we can still collect some meaningful statistics in SpotEx. Because the model requires Wi-Fi to be switched on, we have unique ID for the each client automatically. It is MAC-address. Actually, it is global UUID. So, we can still distinguish our mobile clients, even without the login info for them. For example, it let us detect that the particular data snipped is opened by the person, who did that twice during the last week, etc.

SpotEx CMS create ordinary mobile web pages. This fact lets us use pretty standard methods for web server logs analysis in order to discover activities for our users.

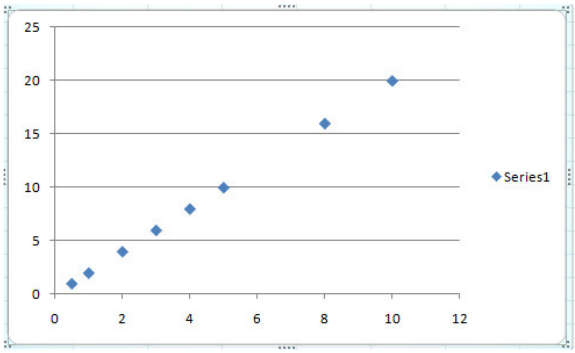

Figure 3. Clicks vs. frequency

For example, Figure 3 illustrates a plot for clicks (opened data snippets per month – y axis) versus visits per month (x axis).

A statistical analysis of the server's log let us examine traffic patterns by time of day, day of week, etc. Server's log lets us discover frequent visitors, for example. SpotEx may use that information in the productions (within the conditional parts of rules). E.g., for proximity marketing some mall may offer special things for frequent visitors, etc.

In general, the fact that we could have for the proposed model the standard web-log opens the door for many existing approaches in data analysis. For example, Hidden Markov Models (HMMs) based on local sequence patterns discovered in data. We can try to find interesting sequences, whose frequency differs in our database from that predicted by the model [22].

And note also, that using statistical data analysis right in the discovering rules is a huge step forward from the model that simply calculates proximity. But this movement requires the real time processing for big data.

Our next play with big data belongs to usage of the history of movements in our data discovery. For example, in the modern LBS applications that are mostly circling near the idea of "check-in", we lack the history of the movement almost completely. It is especially true on the micro-level (indoor). Suppose I have a new check-in in Foursquare. How do I come to this place? In the ordinary web browsing, any hyperlink click can have a referrer field. Where are referrers for LBS? And context-aware applications can fill this gap. Lets us see the project Funf [23], for example.

Funf Probes are the basic collection data objects used by the Funf framework. Each probe is responsible for collecting a specific type of information. These include data collected by on-phone sensors, like accelerometer or GPS location scans, etc. Actually, in Funf many other types of data (context info) can be collected through the phone - from information on the media files stored on the device to call-logs, application usage, or browsing history. Each probe can be remotely configured to be enabled/disabled, what scan intervals or triggers should be used, etc. In other words, Funf is a rich data logger. And that log could be a source for data discovery too. In the current version SpotEx uses a snapshot for currently "visible" wireless networks. But we can use also a history: a set of wireless networks users saw prior that. Think for example about a big mall. The route (path) used for reaching our current position could be logged. This path could be presented also in terms of proximity: a set on nodes mobile user was nearby prior to the current place. And "path" info could be used in our discovery rules too.

Actually, it is a task similar to discovery of convoys in trajectory databases [24]. The challenge is to analyze this information in the real-time. Yes, it is interesting task to provide some a-posteriori analysis like reality-mining [25] did. But it is more interesting to deploy pattern's info in our rules. Simply, it could be slightly different situation for our content rules offering something for one mobile subscriber or to the group of them.

Clustering is the process of organizing objects into groups so that the members of a group are similar and so that members of distinct groups are dissimilar, according to some specific definition of similarity. In the context of trajectory data, clustering attempts to group trajectories based on their geometric proximity in either spatial or spatiotemporal space. Clustering of trajectories involves the process of grouping similar positions, trajectories, and objects (mobile phones in our case). And of course, the existing clustering algorithms should be adapted to the new type of "mail stones" – wireless nodes. It is exactly the next biggest task for SpotEx, associated with big data processing.

The next big direction of our development targets the simplicity of data preparation for SpotEx model. Instead of the separate database with rules (as it is described above), we can add the ability to provide a special markup for existing

HTML files. For example, custom HTML5 attributes are good candidates for encoding our conditions. Such approach let as describe (declare) rules right in HTML code. HTML block presents a rule, block's body describes data chunk and block's attributes describe conditions. The visibility for such block depends on the network context, and this feature could be supported via plain JavaScript code. This JavaScript code should be loaded locally, because it must work with the current context (with the network environment).

We can achieve that via a special light implementation of local (on the phone) web server. This local web server should respond to one type of requests only. It will return the current context (networks environment) in JSON (JSONP) format.

In this case, JavaScript code loaded from local server can read all the HTML attributes related to SpotEx. JavaScript code can simply set CSS visibility style depends on the context. Such a simple trick let us make any existing HTML page "Wi-Fi context aware". If the script is not available, our page should work as a "standard" HTML page.

A very interesting question in this relation is how effectively can we use CSS3 queries for setting our context-aware rules?

The next prospect development adds markup to the whole data feeds. For example, there are so called geo-rss feeds [26]. GeoRSS is standard proposal for geo-enabling (geo-tagging) "really simple syndication" (RSS) feeds with location information. GeoRSS proposes a standardized way for location encoding.

By the similar manner, we can try to incorporate context information into data feeds. Mobile application can collect wireless info context (network name, access point address, frequency, signal level, etc.) We can use this collection for calculating fingerprint (e.g., MD5 hash code) and incorporating it into our RSS feeds. The final idea is to create standardized way in which context is encoded (similar to GeoRSS, for example). Why is it especially interesting? It is not only about SpotEx-like application, collected data feeds by the context. It is about the mobile search in general. The search application on mobile has got access to the current context info. It could be used for getting data feeds with the similar context-related fingerprint.

Of course, our "context-encoded" data feed could be a data feed obtained from the social network (e.g. RSS feed from Twitter). We can associate (probably, temporarily) any appropriate data feed from Twitter with context info. It could be data feed from mall's marketing, for example. This association makes data feed discoverable for hyper-local search. And we do not need to change profile settings in the social network.

Network proximity ideas from SpotEx could be used in communication tools. It is WiFiChat service [27]. This mobile application uses the principles described in this article and offers on-demand communication tools (web chat and discussions groups) for mobile users nearby the same Wi-Fi access point. It could be described as SpotEx with the predefined content. The typical use case is on-demand communication board for passengers in train (bus) with Wi-Fi access point. For Wi-Fi access points opened right on the mobile phone, this application can present the personal communication hub. This Android application is actually a wrapper for web mashup that combines HTML5 web chat engine and cloud based forums from Disqus:

Figure 4. Wi-Fi Chat application

It is the typical tool for the ad-hoc communications on the go.

IV. THE FUTURE DEVELOPMENT

Here, we see several almost obvious steps. At the first hand, it is open API for SpotEx database. Currently, SpotEx mobile application internally obtains data in JSON (JSONP) format from our server-side infrastructure. Open API will make this interface public available.

As soon as API will be live, the next step is almost mandatory. We need to move the rest of the development into web domain. The good candidates for this step are web intents [28]. Web Intents present a framework for client-side service discovery and inter-application communication. Services can register their intention to be able to handle an action in the user's defined callbacks (on the user's behalf). Applications simply request to start (invoke) an action of a certain verb. For example, it could be some like share, edit, view, pick, etc. The system finds the appropriate services for the user, based on the user's preference. It is the basic.

Intents are the core part of Android Architecture. Three of the four basic OS component types (activities, services, and broadcast receivers) are activated by an asynchronous message called as intent.

Intents provide runtime binding for individual components to each other, whether the component belongs to your application or another.

Created intent defines a message to activate either a specific component (explicit intent) or a specific type of component (implicit intent).

For example, our intent might convey a request for an activity to show a list of Wi-Fi networks or to open a data chunk. In some cases, developers can start an activity to receive a result. It is exactly what we need for SpotEx. For example, we can issue an intent to let the user pick a list of

nearby access points and have it returned to us - the return intent includes data in JSON format.

Intents let us hide low-level programming (read – getting network context) from developers and deal with the web development only. Here is the schematic example for web intents integration. It uses the hypothetical SpotEx web intent:

1. Start intent's activity and pass it extra data (network context info)

```
var spotexIintent = new Intent();

spotexIntent.action = "http://webintents.org/spotex";
spotexIntent.putExtra("AccessPoints",
List_Of_Networks);
window.navigator.startActivity(spotexIntent);
```

2. Get local data (it is JSON rather than XML) and display them in our application

```
window.navigator.onActivity = function(json_data) {
var output = document.getElementById("output_div");
output.textContent = JSON.stringify(json_data);
    };  }, false);
```

The key element here is *onActivity* callback. It returns JSON formatted data back to JavaScript code. Additionally, web intents based approach is asynchronous by its nature, and we do not need to organize asynchronous calls by our own.

On practice, Web intents support means the local web server (web server right on the mobile phone).Web server will support HTTP requests to local wireless sensors.

Also, we are planning to add Bluetooth proximity too. But, by our vision, we should avoid the typical Bluetooth usage cases (push proxy as per classical Bluetooth marketing). Vice versa, in SpotEx Bluetooth nodes should be used the same manner, we are using Wi-Fi access points. Bluetooth nodes are presence triggers we can describe our rules for.

The next prospect area SpotEx could be extended towards is Wi-Fi Direct specification. Wi-Fi Direct devices (mobile phones, printers, cameras, PCs, and gaming devices)can connect directly to one another without access to a traditional network. Connected devices can transfer content and share applications anytime and anywhere. Devices can make a one-to-one connection, or a group of several devices can connect simultaneously. They can connect once for a single exchange, or they can link together automatically each time they are in proximity [29].

Wi-Fi Direct specification lets any single Wi-Fi Direct device to be in charge of the Group of devices. This role includes administrative functions. Selected device can control which devices are allowed to join the Group and when the Group is started. All Wi-Fi Direct devices must be able to negotiate which device adopts this (in charge) role when forming a Group with another Wi-Fi Direct device. The Group control device will play a role of "server" for the whole group. It will execute modified rule engine and provide the above described dynamically assembled web page with data chunks for the each device in the group.

V. CONCLUSION

This paper describes redesigned SpotEx model for context-aware data discovery for mobile users developed on the ideas of Wi-Fi and Bluetooth proximity. The proposed model uses smart-phone as a proximity sensor. Service can use existing as well as the especially created networks nodes as presence triggers for delivering user-defined content right to mobile subscribers.

The paper describes how the proposed model could be linked to the social streams too. Another new development in this paper covers the process of collecting historical proximity logs.

SpotEx service could be used for delivering commercial information (deals, discounts, coupons) in malls, hyper-local news data, data discovery in Smart City projects, personal news, etc.

ACKNOWLEDGMENT

The paper is financed from ERDF's project SATTEH (No. 2010/0189/2DP/2.1.1.2.0/10/APIA/VIAA/019) being implemented in Engineering Research Institute "Ventspils International Radio Astronomy Centre" of Ventspils University College (VIRAC).

REFERENCES

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

[2] A. Day, Understanding and Using Context (2001). Human-Computer Interaction Institute. Paper 34. http://repository.cmu.edu/hcii/34 <retrieved: Aug, 2012>

[3] N. Hristova and G. O'Hare, Ad-me: Wireless Advertising Adapted to the User Location, Device and Emotions, HICSS-37, 2004.

[4] C. Bolchini1, G. Orsi, E. Quintarelli, F. A. Schreiber, and L. Tanca Context modeling and context awareness: steps forward in the context-addict project. IEEE Data Eng. Bull., 34(2): pp. 47–54

[5] R. Ahas, S. Silm, O. Järv, E. Saluveer, and M.Tiru Using Mobile Positioning Data to Model Locations Meaningful to Users of Mobile Phones, Journal of Urban Technology Vol. 17, N. 1, 2010 pp. 3-27

[6] Comparison of Wireless Indoor Positioning Technologies http://www.productivet.com/docs-2/Wireless_Comparison.pdf, <retrieved: Aug, 2012>

[7] Yi-C. Chen, Ji-R. Chiang, H. Chu, P. Huang, and A.W. Tsui Sensor-assisted Wi-Fi indoor location system for adapting to environmental dynamics, MSWiM '05 Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems pp. 118- 125

[8] K. Vandikas, A. Katranidou, L. Kriara, H.Baltzakis, T.Papakonstantinou, and M. Papadopouli, Autonomics '07 Proceedings of the 1st international conference on Autonomic computing and communication systems, Article No. 2 , 2007

[9] R. José, A. Moreira, H. Rodrigues, and N. Davies The AROUND Architecture for Dynamic Location-Based Services in Mobile Networks and Applications Vol. 8, N. 4, pp. 377-387, DOI: 10.1023/A:1024531629631

[10] SpotEx Project: http://spotex.linkstore.ru, <retrieved: Aug, 2012>

[11] G.J. F. Jones and P.J.Brown , Context-Aware Retrieval for Ubiquitous Computing Environments, MOBILE AND

UBIQUITOUS INFORMATION ACCESS, Lecture Notes in Computer Science, 2004, Vol. 2954/2004, pp.371-374, DOI: 10.1007/978-3-540-24641-1_17

[12] P. Coppola1, V. Della Mea1, L. Di Gaspero, D. Mischis1, S. Mizzaro1, E. Nazzi1, I. Scagnetto1, and L. Vassena1 Context Aware Browser Dagstuhl Seminar Proceedings 09101 Interactive Information Retrieval http://drops.dagstuhl.de/opus/volltexte/2009/2151 <retrieved: Aug, 2012>

[13] T. Zhang. An architecture for building customizable context-aware applications by end-users. Proceedings of Second IEEE International Conference on Pervasive Computing and Communication, PerCom 2004, Doctoral colloquium.

[14] T. Gu , H. K. Pung , D. Q. Zhang , H. K. Pung, and D. Zhang. A Bayesian Approach For Dealing With Uncertain Contexts (2004)

[15] M.P. Papazoglou, P.Traverso, S. Dustdar, and F.Leymann Service-oriented computing: state of the art and research challenges. Computer 40, 2007, pp. 38–45

[16] Q.Sheng, S.Pohlenz, J.Yu, H.Wong, A.Ngu, and Z.Maamar ContextServ: a platform for rapid and flexible development of context-aware web services. ICSE, 2009, pp. 619–622

[17] D. Namiot and M. Schneps-Schneppe About location-aware mobile messages, NGMAST, 2011, pp. 48-53 DOI: 10.1109/NGMAST.2011.19

[18] D. Namiot and M. Sneps-Sneppe Proximity as a Service, Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on Print ISBN: 978-1-4673-1672-9, pp. 199-205

[19] E. Friedman-Hill Jess in action: rule-based systems in Java, Manning Publications Co. Greenwich, CT, USA 2003 ISBN: 9781930110892

[20] C.Forgy, RETE: A fast algorithm for the many pattern/many object pattern match problem, Artificial Intelligence, 1982, 19(1): pp. 17-37

[21] S. Mizzaro and L. Vassena A social approach to context-aware retrieval, World Wide Web, Vol. 14, N. 4, 2011, pp. 377-405, DOI: 10.1007/s11280-011-0116-6

[22] M. Narasimha Murty and V. Susheela Devi Hidden Markov Models, Pattern Recognition, Undergraduate Topics in Computer Science, 2011, Vol. 0, pp.103-122, DOI: 10.1007/978-0-85729-495-1_5

[23] Funf Open Sensing Framework http://funf.media.mit.edu/ <retrieved: Aug 2012>

[24] H.Aung and K. Tan Discovery of Evolving Convoys, Scientific and Statistical Database Management, Lecture Notes in Computer Science, 2010, Vol. 6187/2010, pp. 196-213, DOI: 10.1007/978-3-642-13818-8_16

[25] A.Pentland, T.Choudhury, N.Eagle, and P.Singh Human Dynamics: Computation for Organizations, Pattern Recognition Letters, Vol. 26, N. 4, 2004, pp. 503-511

[26] C. Reed An Introduction to GeoRSS: A Standards Based Approach for Geo-enabling RSS feeds, 2006, http://portal.opengeospatial.org/files/index.php?artifact_id=15755

[27] Wi-Fi Chat Project: http://wifichat.linkstore.ru, <retrieved: Aug, 2012>

[28] Web Intents: http://webintents.org/, <retrieved: Aug, 2012>

[29] Wi-Fi CERTIFIED Wi-Fi Direct http://www.cnetworksolution.com/uploads/wp_Wi-Fi_Direct_20101025_Industry.pdf, <retrieved: Aug, 2012>