АВТОМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА. – 2015. – No.3. – С.49-59.

О ПОДВИЖНЫХ БЕСПРОВОДНЫХ ТЕГАХ

Д. НАМИОТ, *кандидат физико-математических наук*
Факультет Вычислительной математики и кибернетики МГУ им. М.В. Ломоносова
Ленинские горы, 119991 Москва, Россия
E-mail: dnamiot@gmail.com

М. ШНЕПС-ШНЕППЕ, *доктор технических наук, профессор*
Институт математики и информатики Латвийского Университета
бульв. Райня 29, LV-1459 Рига, Латвия
E-mail: manfreds.sneps@gmail.com

Тема статьи относится к услугам мобильной сети, связанным с контекстом. Мы рассматриваем вопросы разработки мобильных сервисов с использованием беспроводных тегов. При этом предлагается новая модель использования (применения) Core Bluetooth устройств. В нашей модели устройства с Bluetooth выступают в роли тегов и служат для задания (привязки) и последующего определения контекстно-зависимых данных, доступных мобильным пользователям. В качестве таких динамических тегов могут выступать как мобильные телефоны, так и уже существующие устройства с поддержкой Bluetooth. При этом единственного мобильного приложения достаточно для охвата всех этапов жизненного цикла контекстных сервисов. Приложение обеспечивает представление тега (тегов), привязку к ним информационного наполнения, а также просмотр данных, привязанных к другим тегам. Возможные области применения – это приложения для торговых и сервисных организаций и контекстно-зависимые сервисы для Smart Cities.

Ключевые слова: Bluetooth, контекстно-зависимые сервисы, беспроводные теги, сетевая близость, облачные сообщения

1. ВВЕДЕНИЕ

Эта статья является продолжением наших исследований, связанных с использованием сетевой близости (*network proximity*) в мобильных приложениях. В отличие от традиционного подхода к определению местоположения мобильного телефона пользователя по гео-данным, мы определяем (на самом деле – оцениваем) местоположение посредством анализа близости мобильного абонента к определенным сетевым узлам (например, к точкам доступа Wi-Fi или Bluetooth узлам). Мобильный телефон при этом, например, рассматривается как сенсор близости, и гео-позиционная информация заменяется данными о сетевой близости (*network proximity*) [1].

Одним из решений для организации позиционирования и навигации (в первую очередь, в помещениях) является использование аппаратных тегов. Под термином тег (RFID tag, NFC tag, Bluetooth tag и т.д.) здесь понимается некоторое аппаратное устройство, которое поддерживает

определенный коммуникационный протокол (протоколы). Любая подобная система состоит из считывающего устройства и меток-передатчиков (транспондеров). Сразу отметим, что в данной работе в качестве считывателей рассматриваются мобильные телефоны.

Применение у тегов может быть самое различное: идентификация, обмен данными, навигация в помещениях и т.п. Одним из широко поддерживаемых коммуникационных стеков в таких устройствах является Bluetooth. Это связано с технологичностью производства таких устройств, а также с историей самого протокола, его достаточной проработанностью и т.д. До широкого распространения Wi-Fi, Bluetooth (приложения на базе Bluetooth), по сути, являлся основным способом для связи мобильных устройств.

Настоящая работа является дальнейшим развитием идей, ранее изложенных в статьях [2], [3].

Одним из примеров использования Bluetooth является технология iBeacon, анонсированная компанией Apple в составе iOS 7. Этот подход основан на новом стандарте Bluetooth Low Energy (BLE) [4]. Стандарт BLE был специально разработан для использования в сенсорах, которые долгое время должны работать без замены батареи. iBeacon от Apple представляет собой как раз пример такого сенсора [5]. Идея использования состоит в том, что приложение (сервис) может определять наличие подобного рода тегов поблизости и, в зависимости от этого, выполнять какие-либо действия (получать какую либо информацию), исходя из предположения, что телефон (приемник), на котором работает данное приложение, находится поблизости от конкретных тегов. Каждый тег при этом, естественно, должен как-то идентифицироваться. Именно эта идентификация тегов и учитывается приложениями. Приложения, по сути, привязаны к некоторой группе тегов, и наоборот, группа тегов обслуживает конкретное приложение.

В настоящее время, Apple не является единственной компанией, которая производит подобные устройства [6]. Появился даже альтернативный стандарт AltBeacon [7]. Вместе с тем, BLE пока не получил широкого распространения, то есть, по крайней мере, не заменил и не вытеснил традиционные Bluetooth устройства. Поддержка Bluetooth уже давно существует практически во всех мобильных телефонах. В настоящей работе рассматривается программная модель, которая позволяет создавать Bluetooth теги непосредственно на мобильном телефоне. Основным достижением этой модели является то, что она охватывает весь жизненный цикл сервисов – создание тега (тегов), привязку к ним информационного наполнения, создание сервисов на основе имеющихся тегов, а также их эксплуатацию. Мобильного телефона с поддержкой Bluetooth должно быть достаточно как для использования сервиса, так и для его создания.

2. BLUETOOTH DATA POINTS

Предложенный подход к использованию Bluetooth узлов в качестве тегов для активации мобильного контента основан на идеях использования сетевой близости, описанных ранее в проекте SpotEx. Идея состоит в создании связки между задаваемым (определяемым) пользователем контентом и идентификацией сетевых узлов [8].

Сравним между собой семантику тегов и точки доступа Bluetooth в так-называемом *discovery*-режиме. В этом режиме другие пользователи Bluetooth могут видеть идентификацию сетевого узла. Тег не содержит никаких данных и транслирует лишь собственную идентификацию. Так, iBeacon от Apple транслирует в эфир лишь собственный ID (глобальный идентификатор) и два числа, которые могут конфигурироваться программно (локальный идентификатор). Все данные, которые получает, обрабатывает или передает приложение, работающее с тегами, находятся “вне” тега (в облаке, например). Никакого взаимодействия приложения и тега в части обмена данными нет. Только односторонний обмен – передача идентификаторов от тега (тегов) работающему

приложению. Если мы рассмотрим точку Bluetooth в режиме *discovery*, то картина будет, на самом деле, аналогичная. Сетевой узел передает (транслирует) свою идентификацию (имя и MAC-адрес). Сетевой узел сам по себе не содержит никаких данных. Все данные могут находиться только вне узла. Именно на этом совпадении и построена идея об использовании сетевых узлов как беспроводных тегов. А тот факт, что Bluetooth точка может быть открыта непосредственно на мобильном телефоне, делает такой тег еще и подвижным. Если мы каким-то образом сможем связать (ассоциировать) пользовательские данные с идентификацией сетевого узла, то такие данные можно будет показывать мобильным пользователям находящимся поблизости от сетевых узлов. Близость здесь определяется по доступной (распространяемой) идентификации сетевого узла. Иными словами – мобильный телефон определяет наличие Bluetooth узла. Для Bluetooth, например, это не более нескольких десятков метров. При этом важно, что сетевой узел может перемещаться. Например, телефон с включенным интерфейсом Bluetooth перемещается вместе со своим владельцем. Если данные, как было указано выше, привязаны к идентификации сетевого узла, то они также "перемещаются" вместе с телефоном. Точнее – перемещается область доступности данных, которая автоматически соответствует текущему положению сетевого узла (мобильного телефона в данном случае). Еще более характерный пример – это Bluetooth узлы, которые существуют в современных автомобилях и, естественно, перемещаются вместе с автомобилем.

Заметим, что аналогичные рассуждения верны и для точки доступа Wi-Fi (там используется другой термин – *broadcast mode*).

В SpotEx пользователь (издатель) может связать некоторый контент (практически – произвольный фрагмент кода на HTML или просто ссылку на веб-ресурс) с доступными идентификаторами точек доступа Wi-Fi. Точки доступа могут быть как существующими и публично доступными, так и специально созданными в рамках данного проекта. В частности, они могут быть открыты (запущены) непосредственно на мобильном телефоне (телефонах). Именно последнее утверждение – специальное создание сетевых узлов для целей идентификации (адресации) контента и привело к идее реализации аналога SpotEx на базе Bluetooth. Дело в том, что в отличие от Wi-Fi узлов, узлы (точки) Bluetooth можно создавать программно. Иными словами, приложение на базе мобильной ОС Android может запустить на мобильном телефоне Bluetooth узел. Именно этот факт и позволяет говорить о поддержке всего жизненного цикла – от создания сервиса до его запуска и эксплуатации, на одном телефоне. Это означает, что мобильного телефона с поддержкой Bluetooth должно быть достаточно как для просмотра контента, так и для организации (создания) этого самого контента.

Привязка контента к сетевым узлам означает задание правил (предикатов) видимости. Например, если мобильный узел с именем Café виден в данный момент времени мобильному пользователю, то ему становится доступным (открывается, показывается) некоторый соответствующий этому факту контент. Правила активации (видимости) определяет непосредственно автор (владелец) контента. Для читателя (мобильного абонента) конечное приложение выглядит как браузер. Приложение оценивает доступные сетевые узлы, определяет сработавшие правила и предъявляет для просмотра соответствующий набор данных [9]. На практике это выглядит как формирование динамической веб-страницы, содержание которой формируется из отдельных фрагментов, соответствующих сработавшим правилам. Финальный вид показан на рисунке 1.

Отметим, что речь при этом не идет о присоединении к какому-либо сетевому узлу (точке доступа Wi-Fi, Bluetooth узлу). Собственно контент может располагаться где угодно в сети. Сетевой узел работает просто как триггер. Если мобильное приложение "видит" данный сетевой узел, то телефон (абонент) где-то поблизости. В этом основная идея, и "браузер" становится средством

Рис.1. Данные на основе сетевой близости.

доступа именно к локальному контенту. Это принципиальное отличие от модели представления данных, построенной целиком на уведомлениях. В “браузере” просмотр данных инициируется мобильным абонентом, а не системой.

Для Bluetooth картина использования сетевой близости концептуально одинакова, но есть и важные технические различия. Они касаются расстояния, на котором можно обнаруживать Bluetooth точки (теги активации данных) – это, в целом, будет меньше, чем для Wi-Fi. Определение (поиск) Bluetooth узлов, находящихся в так называемом *discoverable mode* работает медленнее, чем поиск Wi-Fi узлов. С другой стороны, количество Bluetooth узлов, используемых в качестве точек привязки данных, будет (опять в среднем) больше, чем Wi-Fi узлов. Еще одним важным моментом является то, что точку Bluetooth можно создать (открыть) на мобильном телефоне программно (Android SDK поддерживает это на уровне API). Эта модель – привязка пользовательских данных к точкам Bluetooth и называется далее Bluetooth Data Points (BDP).

Важно еще раз отметить, что установления связи между двумя объектами (на основе Bluetooth или иного протокола) не предполагается. Соответственно, модель не добавляет никаких новых ограничений с точки зрения безопасности. Большим плюсом является также то, что используется стандартный механизм анонсирования, существующий в Core Bluetooth. Это исключает необходимость каких-либо модификаций на аппаратном уровне и позволяет использовать модель с практически всеми современными смартфонами (поддержка Core Bluetooth существует в каждом из них). Относительно скорости определения (нахождения) тега другим узлом можно отметить следующее. Поскольку используется базовый механизм Bluetooth, существующий в мобильной операционной системе, то скорость, естественно, не зависит от модели Bluetooth Data Points. По нашим тестам, в практических инсталляциях, типичное время сканирования Bluetooth устройств мобильным телефоном на платформе Android (версия Android SDK 4.2) составляло около 2 секунд. При этом обнаруживались устройства на расстоянии 15-20 метров (открытый зал в торговом центре, полный заряд батареи мобильного телефона). Очевидно, что это никак не может быть использовано в приложениях реального времени и не может служить заменой специализированных сетевых решений для управления движением и т.п. Модель применения для BDP – это именно информационные системы, которые не являются критическими приложениями. По своей природе, как базовая ОС (Android, в нашем случае), так и процесс сканирования сети для поиска Bluetooth узлов, конечно, не могут использоваться в системах реального времени.

Можно оценить и возможные скорости движения в системе (перемещаться могут как потребители информации, так и теги, к которым привязываются данные). Если (как указано выше) тег (точка Bluetooth на телефоне) обнаруживалась на расстоянии 20 м за 2 секунды, то отсюда легко получить оценку скорости. Посетитель торгового центра (кампуса, офисного здания) – это типичный клиент для такой информационной системы. Естественно, если перемещаются и тег и потребитель данных, речь идет о разности их скоростей. В наших экспериментах с этой моделью для автомобилей в качестве тега выступала точка доступа Bluetooth, присутствующая во многих современных автомобилях [10]. Такой подвижный тег с успехом определялся с помощью мобильных телефонов, находящихся в соседних автомобилях, поскольку в городских условиях скорость движения почти всегда примерно одинаковая (не говоря уже о простом нахождении в дорожной пробке).

3. КАК ИЗМЕРЯЕТСЯ СЕТЕВАЯ БЛИЗОСТЬ

Формальное определение для оценки местоположения мобильных абонентов, основанное на сетевой близости, приводится во многих статьях [11].

Радио-сигнатура. Это простейший подход, описанный в работе [12]. Метрика близости основана на подсчете количества раз, которое конкретное устройство видело определенную точку доступа [13]. Мобильное приложение на телефоне периодически записывает адреса видимых сетевых узлов. Радио-сигнатура (так называемый *fingerprint*) определяется на основе подсчета процента времени (доли временных интервалов), в течение которых конкретный MAC-адрес присутствовал во всех записях. Совокупность (вектор) таких долей и есть то, что называется *network fingerprint*. Она (сигнатура) вычисляется для каждого интересующего нас места. Эта сигнатура используется для сравнения позиций мобильных абонентов.

Сравнение двух сигнатур f_1 и f_2 выполняется следующим образом. Пусть M есть объединение MAC-адресов в f_1 и f_2 . Для MAC-адреса $m \in M$ обозначим его доли вхождения как $f_1(m)$, так и $f_2(m)$. Тогда схожесть S сигнатур f_1 и f_2 может быть вычислена следующим образом:

$$S = \sum_{m \in M} (f_1(m) + f_2(m)) * \mathrm{MinMax}(m),$$

где $\mathrm{MinMax}(m) = \min(f_1(m), f_2(m)) / \max(f_1(m), f_2(m))$.

Понятно, что значение S будет возрастать, если в сравниваемых сигнатурах присутствует больше одинаковых адресов.

Ранговая корреляция измерений. Использование радио-сигнатур базируется на предположении, что мобильное устройство всегда измеряет сигнал одинаково. Очевидно, что это является достаточно строгим предположением. В реальных условиях заряд батареи, например, не будет одинаковым, и одно и то же устройство может выдавать разные значения в зависимости от заряда батареи, ориентации и т.д. Поэтому в сравнениях вместо абсолютного значения силы сигнала используют ранжирование. То есть, сравнивают списки точек доступа, отсортированные по силе сигнала. Например, если получили три точки доступа с соответствующими значениями RSSI (SS_A; SS_B; SS_C) = (-50; -20; -40), то мы можем заменить абсолютные значения рангом (R_A; R_B; R_C) = = (3; 1; 2) [14]. А сравнивать ранги можно с помощью коэффициента ранговой корреляции Спирмена (*Spearman rank-order correlation coefficient*) [15].

Корреляция абсолютных значений. Для анализа абсолютных значений RSSI в пространстве сигналов можно ввести Евклидово расстояние или вычислять коэффициент Tanimoto [16]. В обоих случаях расчеты начинаются с вычисления средних значений сигнала. Для каждого сетевого узла по измеренным векторам сигналов S_x вычисляется вектор средних значений S'_x. В случае Евклидова расстояния используется попарное сравнение векторов S'_a и S'_b, где один из них представляет некоторый недостижимый эталон с уровнями сигналов, например, равным – 100 dBm, и вычисляем значение расстояния

$$d_{a,b} = \| S'_a - S'_b \| .$$

Для вычислений на основе коэффициента Tanimoto расстояние подсчитывается следующим образом [15]:

$$d_{a,b} = 1 - (S'_a \cdot S'_b) / (\| S'_a \|^2 + \| S'_b \|^2 - S'_a \cdot S'_b).$$

В обоих случаях расстояние между векторами увеличивается при расхождении абсолютных значений векторов.

В случае Bluetooth тега для оценки дистанции мы можем сравнивать измеренный сигнал (RSSI) с калибровочным значением (оно обычно также передается тегом). Это калибровочное значение описывает значение RSSI на расстоянии в 1 метр. Обозначим это значение как R_1, а значение силы сигнала тега, измеренное конкретным мобильным устройством – R_2. Тогда:

$$\text{dBm\_ratio} = R_1 - R_2 ,$$

линейное отношение может быть получено по стандартной формуле:

$$\text{linear\_ratio} = 10^{(\text{dBm\_ratio} / 10)}.$$

Если мы примем, что сила сигнала уменьшается как $1/r^2$ (r – расстояние), то:

$$R = R_1 / r^2 ,$$

$$\text{r} = \sqrt{linear\_ratio} .$$

Но это будет именно оценка расстояния. В реальных условиях (помещениях) сигнал, в силу отражений, будет, скорее всего, уменьшаться медленнее.

4. ХРАНЕНИЕ ДАННЫХ

Простейшей формой указанного выше предиката видимости (доступности) данных является правило, в котором условие описывает видимость одного сетевого узла. Иными словами, каждый предикат выглядит следующим образом:

Если (Сетевой Узел Видим) То {фрагмент данных для представления пользователю}.

Это означает, что условие в правиле может быть представлено просто как идентификация некоторого сетевого узла. Для случая Bluetooth узла таким идентификатором может выступать MAC-адрес. Иными словами, условие – это доступность в данный момент конкретного MAC-адреса.

Данные (тело правила, заключение) могут быть представлены как некоторый JSON-массив. В качестве основного элемента выступает текст, который при отображении на стороне клиента после запроса будет показываться как некоторый HTML фрагмент. При этом отображении мы сможем автоматически добавлять специальные теги, отображая текст, содержащий email или телефон

как ссылку и т.д. Другие элементы JSON-массива могут отображать специальным образом интерпретируемые фрагменты. Например, ссылку на Facebook (Twitter, Linkedin) профиль.

В итоге, для представления данных мы имеем классическую схему key – value. Ключ здесь – это MAC-адрес, значение – JSON массив. Как результат, мы получаем классическую для NoSQL систем модель. На практике мы использовали Apache Accumolo [17].

С точки зрения модели данных, BDP сохраняет следующую информацию:

(recordID, MAC_address, data_array).

Каждая запись имеет некоторый уникальный идентификатор (*recordID*) и описывает один фрагмент данных (*data_array*), связанный с конкретным адресом (точкой Bluetooth) – *MAC_address*. Естественно, что для одного Bluetooth узла мы можем вести несколько записей (несколько фрагментов данных).

Для сбора статистики и анализа производительности к каждой записи необходимо добавить, по крайней мере, два служебных поля – время, когда запись была создана и время, когда она была последний раз модифицирована. Также необходимо дать возможность авторам отключать (включать заново) свои объявления. В итоге мы приходим к следующей структуре:

(recordID, MAC_address, timestamp_created, timestamp_modified, status, data_array),

здесь поле status представляет логическое значение (или целое 1/0). Это поле описывает текущий статус фрагмента данных (активен или нет).

Массив данных содержит JSON структуры. Это позволяет организовать гибкую обработку на стороне клиента. Кроме того, это поддерживает одинаковый формат выдачи, как для клиентских приложений, так и для программных интерфейсов (API). JSON массив, возвращаемый системой содержит список фрагментов данных:

```
[
{“type”:”some_type”,”data”:”some_data”},
{“type”: ...}, …
]
```

Поле *type* описывает здесь один из стандартных типов, поддерживаемых системой. В текущей реализации система поддерживает следующие типы: *text, url, image, email, phone, fbprofile, twprofile*. Тип *text* здесь есть просто последовательность символов, *url* и *image* описывают веб-ресурсы, *fbprofile* и *twprofile* также представляют веб-ресурсы, но интерпретируются специальным образом. Например, *fbprofile* представляет собой URL, который ссылается на некоторый профиль Facebook, *twprofile* – то же самое, но для Twitter. На уровне моделей применения это позволит решать такие задачи, как публикация ссылки на собственный аккаунт Facebook и Twitter, так что он может быть виден находящимся поблизости мобильным пользователям. А на уровне реализации это оставляет за конкретным программным клиентом решение о том, как отображать данные. Можно показывать просто ссылку на профайл, а можно, например, показать еще и фото из профайла пользователя (оно доступно через публичный Facebook API).

Общий алгоритм работы – следующий. Мобильный клиент определяет видимые Bluetooth точки. Далее для каждого кандидата, по его MAC-адресу, запрашивается возможная связанная информация в хранилище. Иными словами, типичный запрос будет обращаться за получением информации для конкретного MAC-адреса. Это будет запрос, связанный с первичным индексом. Данные о подобных бенчмарках для Apache Accumulo показывают производительность до 100

миллионов транзакций в секунду [18]. Для дальнейшей оптимизации мы можем сократить число запросов к базе данных, рассматривая, например, только Bluetooth узлы с определенным именем. Также возможен оффлайновый режим работы, когда вся информация по какому-либо зданию (торговому центру) или некоторому географическому региону будет предварительно загружена в приложение-браузер.

Для оценки “близости” каждого такого тега в текущей реализации используется простое сравнение силы сигнала (RSSI). Программный интерфейс (API) просто выдает фрагменты данных в порядке, соответствующем убыванию силы сигнала.

5. АКТИВНЫЙ И ПАССИВНЫЙ МОНИТОРИНГ

Возможны два варианта определения близко расположенных тегов (BDP). Во-первых, мобильный абонент может явно запустить сканирующее приложение. Полная аналогия с тем, как мы запускаем веб-браузер, чтобы посмотреть какие-либо веб-страницы.

Другая возможность – это сканировать сети в фоновом режиме. Схема работы остается такой же, только вместо прямого просмотра доступных сообщений, пользователь будет получать уведомления с ними. Здесь используются так называемые push-уведомления (Рис.2).

Это является вполне работающим вариантом, как показано в работе [19]. Основная проблема здесь – скорость определения видимых Bluetooth узлов. Даже двигающийся шагом посетитель торгового центра, например, будет “проходить” имеющиеся теги быстрее, чем его телефон зафиксирует их наличие. Эта проблема существует и для BLE тегов (iBeacons). Поэтому мы считаем, что на современном этапе (до появления новых стандартов с резко уменьшенным временем распознавания сетевых узлов) прямой запуск “браузера” для определения сетевых узлов (показа связанной информации) является более предпочтительным. Пользователь сам явно определяет момент, когда он готов (хочет) получить какую-то информацию. На практике (в помещениях) можно предложить отмечать каким-либо видимым значком (по аналогии с наклейками с символом Wi-Fi) наличие (присутствие) тегов.

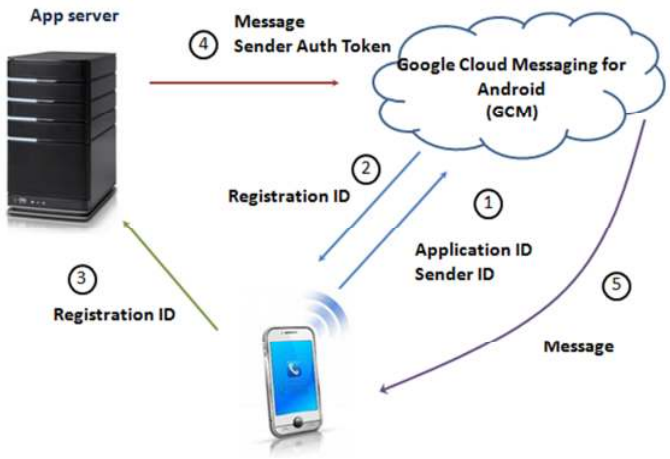

Рис.2. Google Cloud Messaging [18].

6. СБОР И АНАЛИЗ СТАТИСТИКИ

В случае описанного выше пассивного мониторинга (фонового определения сетевых узлов) мобильное устройство будет практически все время находиться с включенным Bluetooth интерфейсом. В таком случае, сбор статистики может вестись также пассивно, с использованием *beacon frames*. В наших собственных разработках мы использовали регистрирующее устройство от компании Libelium [11]. Основное ее достоинство – открытость. Измерения могут накапливаться, например, во внешней базе данных MySQL и, следовательно, открыты для сторонних разработчиков. Другим достоинством системы является то, что измерительное устройство технически представляет собой еще и обычный Wi-Fi маршрутизатор, который может определять и присутствие Bluetooth устройств (рис.3).

Рис.3. Обнаружение смартфонов шлюзом Libelium.

Но более интересным будет являться анализ статистики в случае явного обращения к “браузеру” для просмотра данных, связанных с беспроводными узлами. В этом случае предметом анализа будет аналог веб-лога, собираемого при анализе статистики обычных веб-сайтов. Аналогом IP-адреса здесь будет выступать MAC-адрес клиента, из этого же адреса может быть извлечена информация о типе мобильного устройства (аналог User-Agent), а аналогом URI (запроса) будет список MAC-адресов тегов (BDP).

По аналогии с работой [11] отметим, что основным применением подобного рода статистики будет сравнительный анализ. Мы не можем достоверно предсказать, какой процент из реальных посетителей будет определен (воспользуется приложением). Следовательно, абсолютные цифры являются, по определению, не точными. Но предположение о том, что доля зарегистрированных смартфонов примерно одинакова на всем временном интервале выглядит весьма обоснованным. Именно это и позволяет сравнивать данные за разные дни. Основная цель мониторинга заключаться в сравнении временных рядов. Например, типичным вопросом веб-статистики является – какова оценка количества постоянных посетителей сайта (например, сколько пользователей заходят на сайт постоянно в течение недели и т.д.). Применительно к мобильной статистике можно спросить, как изменилось число постоянных посетителей для контролируемой (замеряемой) области за последние 7 дней, по сравнению с такой же неделей один месяц назад. При этом повторяющиеся регистрации пользователей (MAC-адресов) будут являться аналогом веб-сессии. Следовательно, можно задаваться вопросом о том, как долго посетители остаются в данной области, как часто они возвращаются и т.д.

7. ПРОТОТИП ПРИЛОЖЕНИЯ

В качестве прототипа мы имеем Android приложение, доступное в Google Play [20]. Оно поддерживает все этапы жизненного цикла для BDP: создание объявлений, их публикацию и просмотр (поиск) других объявлений. Работа приложения может быть протестирована с помощью двух телефонов, оснащенных одним и тем же приложением. На первом телефоне мобильный абонент подготавливает некоторый текст (объявление) и публикует его. Для опубликованного объявления приложение открывает точку Bluetooth (в так называемом *discovery mode*) непосредственно на мобильном телефоне автора. Опубликованные данные будут связаны с мобильным телефоном автора. Это иллюстрируется рисунком 4.

Та же самая программа на другом устройстве может быть запущена в режиме просмотра. Приложение находит видимые Bluetooth узлы, получает ассоциированные с ними данные (если это, конечно, BDP и такие данные существуют) и отображает их для пользователя (как браузер). Это проиллюстрировано на рисунке 5.

Рис.4. Публикация данных.

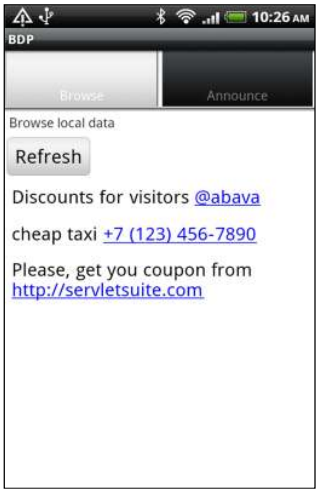

Рис.5. Отображение данных.

При этом, данный браузер автоматически снабжает соответствующие фрагменты текста подходящими HTML тегами. В результате этого номер телефона (email адрес) будет отображаться как ссылка, URL – как гиперлинк и так далее.

Отдельный вопрос, который требует дополнительных исследований – можно ли для просмотра данных использовать обычный браузер. Это требует доступа к сенсорам (сканирования Bluetooth узлов) из JavaScript. Наши начальные эксперименты показывают, что этого можно добиться с помощью компоненты *WebView* в Android. В Android SDK можно определить собственную форму *WebView*, которая будет иметь доступ (возможность использовать) к пользовательскому Java-коду. Этот Java-код будет, по сути, тем же механизмом, который в данный момент работает в BDP. А полученные данные уже будут передаваться в JavaScript код. По такой схеме можно будет встраивать просмотр информации от BDP непосредственно в обычные веб-страницы. Для веб-разработчика подключение BDP будет выглядеть просто как включение JavaScript файла.

СПИСОК ЛИТЕРАТУРЫ

[1] *Namiot D., and Sneps-Sneppe M.* Proximity as a service. *In* Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on,. IEEE. DOI: 10.1109/BCFIC.2012.6217947. pp. 199-205. April.

[2] *Намиот Д.Е.* Персональные Bluetooth теги //International Journal of Open Information Technologies. – 2014. – Т. 2. – №. 3. – С. 35-39.

[3] *Намиот Д.Е.* Мобильные Bluetooth теги //International Journal of Open Information Technologies. – 2014. – Т. 2. – №. 5. – С. 17-23.

[4] *Gomez, Carles, Joaquim Oller, and Josep Paradells*. "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology." Sensors 12.9 (2012): 11734-11753.

[5] *Dilger D.E.* Inside iOS 7: iBeacons enhance apps' location awareness via Bluetooth LE //ed: Apple Insider. – 2013.

[6] *Lerner T.* International Comparisons. In Mobile Payment. Springer Fachmedien Wiesbaden. pp. 137-142, 2013.

[7] AltBeacon http://altbeacon.org/ Retrieved: Oct, 2014.

[8] *Namiot D. & Sneps-Sneppe M*. Context-aware data discovery. In Intelligence in Next Generation Networks (ICIN), 2012 16th International Conference on, pp. 134-141. IEEE. October 2012.

[9] *Namiot Dmitry, and Manfred Sneps-Sneppe*. "Geofence and Network Proximity." Internet of Things, Smart Spaces, and Next Generation Networking. Springer Berlin Heidelberg, 2013. pp.117-127.

[10] *Namiot D. & Sneps-Sneppe M*. CAT - cars as tags. In Communication Technologies for Vehicles (Nets4Cars-Fall), 2014 7th International Workshop on IEEE.

DOI: 10.1109/Nets4CarsFall.2014.7000912, October 2014, pp.50-53.

[11] *Namiot D., and M. Sneps-Sneppe*. "On the analysis of statistics of mobile visitors." Automatic Control and Computer Sciences 48.3, 2014, pp.150-158.

[12] *Azizyan M. et al*. SurroundSense: mobile phone localization via ambience fingerprinting. *In* MobiCom '09 Proceedings of the 15th annual international conference on Mobile computing and networking, pp. 261-272, DOI: 10.1145/1614320.1614350.

[13] *Namiot D. and Sneps-Sneppe M.* Wireless Networks Sensors and Social Streams. *In* Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on (pp. 413-418). IEEE. DOI: 10.1109/WAINA. 2013.27. March 2013.

[14] *Chen Y. et al*. Accuracy characterization for metropolitan-scale Wi-Fi localization. *In* ACM MobiSys, 2005.

[15] *Stuart A*. The correlation between variate-values and ranks in samples from a continous distribution// British Journal of Statistical Psychology, Vol.7, Issue 1, pp.37–44.

[16] *Kjaergaard M. et al*. Mobile sensing of pedestrian flocks in indoor environments using WiFi signals. *In* Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on, pp.95–102.

[17] *Kepner J., Arcand W., Bergeron W., Bliss N., Bond R., Byun C., & Yee C*. Dynamic distributed dimensional data model (D4M) database and computation system. In Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on (pp. 5349-5352). IEEE.

[18] *Sen R., Farris A., & Guerra P.* Benchmarking Apache Accumulo BigData Distributed Table Store Using Its Continuous Test Suite. In Big Data (BigData Congress), 2013 IEEE International Congress on (pp. 334-341). IEEE. June 2013.

[19] *Sneps-Sneppe M. and Namiot D.* Smart cities software: customized messages for mobile subscribers. *In* Wireless Access Flexibility. 2013. pp. 25-36. Springer Berlin Heidelberg.

[20] BDP http://bdp.linkstore.ru. Retrieved: Nov, 2014.

Рукопись получена 05.11.2014,
финальная версия – 09.02.2015.