Локальные веб-страницы

Намиот Д.Е.

***Аннотация*—В статье рассматриваются вопросы использования механизмов контекстно-зависимого программирования для создания веб-сайтов (веб-страниц) посвященных описанию каких-либо объектов (событий) с точно очерченной географической областью видимости (доступности). В работе предложены механизмы для создания мобильных веб-приложений, содержание которых привязано к некоторой определенной географической области. При этом точность такой привязки позволяет различать отдельные области внутри одного помещения.**

***Ключевые слова*—context aware, network proximity, indoor positioning, HTML5.**

I. Введение

В работе рассматриваются вопросы представления локальной (привязанной к некоторому географическому местоположению) информации мобильным пользователям. Речь идет о программировании (создании) мобильных сайтов, для которых содержание страниц соответствует текущему местоположению пользователя. Традиционная схема состоит в определении местоположения пользователя и создании динамических страниц, выдача которых явно задается определенными координатами. Определение географического местоположения является, например, частью стандарта HTML5 [1]. Получив, с разрешения мобильного пользователя, его координаты приложение может сформировать динамическую веб-страницу, содержание которой зависит от текущего местоположения пользователя. При этом формирование страницы может выполняться как на стороне клиента (в браузере), когда веб-приложение запрашивает необходимые данные посредством асинхронных вызовов (AJAX) [2], так и посредством обращения к некоторому серверному приложению (скрипту), которое, получив исходные координаты в качестве параметров, уже целиком формирует выходную страницу. Это направление традиционно относится к сервисам, связанным с позиционированием (Location Based Services [3]).

Существуют различные способы получения информации о местоположении. Не все из них обязательно связаны с непосредственным доступом к устройствам (сервисам) типа GPS [4]. Другие обзоры можно найти в работах [5] и [6], например. Но общим для всех этих сервисов является одно: получение географических координат, описывающих текущее местоположение и последующее определение (формирование) информации для пользователя. С другой стороны, существует направление, называемое контекстно-зависимыми вычислениями, где в качестве характеристики местоположения пользователя может использоваться другая информация, не связанная с географическими координатами. Упрощенно, контекстом называется любая информация, дополняющая географическое местоположение [7,8]. При этом, дополнительная информация (контекст), при наличии определенных метрик, может служить уникальной (до некоторого приближения, естественно) характеристикой местоположения. Или, иными словами, заменить собой информацию о гео-координатах. Зачем это может быть нужно? Самый простой пример – это использование LBS в помещениях [9]. Традиционное гео-позиционирование может быть затруднено, точность позиционирования может быть недостаточной, чтобы различать положения мобильного абонента внутри одного помещения. А вместе с тем, именно различение позиций внутри одного помещение (здания) может оказаться существенным для различного рода сервисов (например, находится покупатель на первом или втором этаже торгового зала).

Именно отсюда берут свое начало подходы (задачи), в которых выдача информации, привязанной к местоположению, может вообще пропускать этап, связанный с определением гео-координат. Действительно, если гео-координаты являются только ключом для последующего поиска гео-зависимой информации, то, оформляю эту же самую информацию другим образом, привязывая ее непосредственно к контексту, мы можем пропустить этап определения гео-координат. Достаточно идентифицировать контекст и использовать эту идентификацию для поиска данных.

Оставшаяся часть статьи структурирована следующим образом. В разделе 2 рассматривается один из возможных подходов к идентификации контекста. В разделе 3 рассматривается возможное применение этой идентификации для веб-программирования.

II. Сетевая близость

Одним из достаточно широко используемых методов идентификации контекста является использование беспроводных сетевых интерфейсов мобильных устройств (Wi-Fi, Bluetooth). Причины этого достаточно прозрачны. Во-первых, эти интерфейсы присутствуют во всех современных смартфонах. Во-вторых, по очевидным причинам, мониторингом мобильных

Статья получена 25 января 2014.

Д.Е. Намиот – старший научный сотрудник факультета ВМК МГУ им. М.В. Ломоносова (e-mail: dnamiot@ gmail.com).

соединений занимается собственно мобильная операционная система. Следовательно, опрос текущего состояния может быть упрощен и не вызывать дополнительных энергозатрат, по сравнению с, например, специально организуемым мониторингом акселерометра.

Информация, получаемая с помощью сетевых интерфейсов, используется для оценки близости мобильного пользователя к элементам сетевой инфраструктуры (network proximity [10). Отметим, что в качестве таких элементов могут выступать и другие мобильные устройства (например, точка доступа Wi-Fi, открытая непосредственно на мобильном телефоне [11]).

Классическая форма использования информации о Wi-Fi устройствах – это коллекция так называемых Wi-Fi fingerprints [12]. Эти цифровые отпечатки характеризуют видимость (достижимость) сетевых узлов и используются, в первую очередь, в задачах позиционирования. Альтернативный подход – это позволить сторонним пользователям непосредственно описывать связи между сетевыми узлами и контентом. Эта тема развивается авторами в проекте SpotEx и связанных разработках [13,14]. Идея состоит в том, чтобы с помощью набора правил (продукций) описать связь между сетевыми элементами и доступным контентом. Иными словами, мобильное приложение (context-aware browser) базируется на внешних описаниях контекста типа следующего:

```
IF  Видима_Точка_Доступа (‘Café’) THEN
{
   представить контент для Cafe
}
```

Условия включают в себя предикаты, которые оперируют атрибутами сетевого окружения (идентификация точки доступа, сила сигнала). Для представления контента используются фрагменты HTML кода. Специальное мобильное приложение сопоставляет текущее сетевое окружение с базой правил, подобных описанному выше и формирует мобильную веб-страницу, которая и предъявляется мобильному пользователю.

Собственно говоря, само название приложения (context-aware browser) наводит на мысль реализовать этот функционал в мобильном браузере. Это позволит отказаться от отдельной базы правил и специального приложения. В качестве такового приложения для пользователей будет выступать обычный браузер. Правила для контента должны будут задаваться непосредственно на мобильных веб-страницах.

В качестве прикладных примеров можно указать приложения для Internet of Things [15,16]. Использование достаточно прозрачно. Данные, предоставляемые мобильным пользователям, зависят, например, от видимости конкретной точки доступа Wi-Fi. Это позволяет, в частности, различать местоположение мобильного пользователя внутри здания (помещения). При этом собственно гео-координаты никак не используются.

В целом, это представляет собой одно из направлений исследований в лаборатории ОИТ ВМК МГУ [17].

III. Информационные сервисы

Технически, для преставления информации о сетевой близости, мы можем говорить о двух подходах.
Во-первых, реализация мобильного браузера может следовать той же идеологии, что и поддержка гео-кодинга в HTML5 [18]. Как это устроено:

```
<script>
function getLocation()
 {
    if (navigator.geolocation)
    {

navigator.geolocation.getCurrentPosition(showPosition);
    }
 }

function showPosition(position)
 {
 var latitude = position.coords.latitude;
 var longitude = position.coords.longitude;
 }
</script>
```

Интерфейсная функция

```
navigator.geolocation.getCurrentPosition()
```

получает в качестве параметра определенный пользователем callback (другую функцию). Эта последняя функция и вызывается при успешном завершении определения координат. Вычисленные результаты передаются туда в виде параметров. Отметим, что это асинхронный процесс.

По аналогии с вышесказанным, реализация мобильного браузера должна добавить интерфейсную функцию, например *getNetworks()*, параметром которой будет выступать также определяемый пользователем callback, куда и будет передаваться в виде JSON массива информация о найденных сетях. Каждый элемент массива описывает одну сеть и содержит, по крайней мере, следующую информацию:

SSID - имя сети (точки доступа)
MAC – MAC-адрес
RSSI – сила сигнала

Отметим, что процесс опроса (определения) сетей является асинхронным. Следовательно, схема с callback вызовом очень хорошо подходит для данной модели.

Наиболее близким по идеологии к данному подходу является Firefox OS [19].

Вот пример из технической документации:

```
interface WiFiManager {
 // request.result set to JS array of wifi networks in range
 DOMRequest getNetworks();

// request fires success if
// successfully able to connect, error otherwise
// any connected; // JSON object (frozen) which contains
// info on the connected network
 DOMRequest connectTemp(any parameters);

 int signalStrength;

// Fires event when we connect to a new WiFi network

 Function onconnect;

// Fires event when we disconnect from a new
// WiFi network
Function ondisconnect;

// Fires event signal strength changes
Function onsignalstrengthchange;
}
```

JSON объект, возвращаемый функцией *getNetworks()* содержит следующую информацию о сети: имя, MAC-адрес, силу сигнала и схему шифрования.

Firefox OS помимо этого предлагает еще и Bluetooth API [20]. Идеология схожая, но общего унификатора нет (поля объектов – разные). Естественно, можно будет создать на пользовательском уровне единую оболочку, которая будет выдавать общий список сетей. Проблемой (помимо собственной распространенности Firefox OS) является статус этих API. Wi-Fi API пока еще только планируется. В то же время Bluetooth API объявлен привилегированным и может использоваться только самой операционной системой. Возможно, чтобы убрать проблемы с безопасностью, стоит разделить эти API. Выделить отдельно уровень, который не содержит операции соединения и обмена данными, а позволяет только опрашивать сетевое окружение. Опрос сетей без возможности соединения не таит в себе угроз для пользователей.

Поле того, как информация о сетях получена, ее использование достаточно прозрачно. Например, каждое из правил, подобных указанному в разделе 2, на веб-странице может быть представлено как некоторый блок:

```
<div id="Café_rule">
   контент для Cafe
</div>
```

который может быть сделан видимым (скрытым) в зависимости от доступности соответствующих Wi-Fi сетей (Bluetooth узлов). Для реализации в этом случае достаточно JavaScript.

В совокупности, такой подход позволит изменить парадигму проектирования сайтов. Не нужно делать отдельные версии для локальных сайтов или событий. Достаточно иметь один общий сайт, на страницах которого и отмечать локальные предложения. Например, как скрытые блоки. Они будут открываться в зависимости от сетевого окружения. То есть мобильный пользователь, открывший общий сайт, но находящийся непосредственно в заданном месте увидит другую (дополнительную) информацию, по сравнению с остальными пользователями того же сайта.

Естественно, что поддержка единого источника данных упрощает (удешевляет) сопровождение.

Другой клиентской моделью, подходящей в данном случае, являются Web Intents [21].

Web Intents представляют собой клиентский (все выполняется в браузере) фреймворк (библиотеку) для опроса сервисов и построения взаимодействия (обмена данными, передачи управления) внутри приложения. Сервис (поименованный фрагмент кода - интент) анонсирует свою готовность поддерживать некоторые операции. Например, редактировать некоторый текст (изображение и т.д.), отправлять сообщения и т.п. Приложение (пользовательский код) запрашивает сервис по некоторой акции (отредактировать, отправить и т.д.), а исполнительная система уже подбирает сервис на основе его анонса. Подобного рода интенты, например, играют важную роль в архитектуре Android.

Для данного конкретного случая интентом должен выступать сервис, который опрашивает сетевое окружение. Обращение к интенту асинхронно, указывается желаемое действие и callback – функция, которая получит данные после выполнения (в случае веб-интентов – это функция *onActivity*).

Использование Web Intents для переноса задач, основанных на сетевой близости, непосредственно в браузер было предложено в ранних работах автора (см., например, [22]). Основной проблемой, связанной с Web Intents остается их экспериментальный статус и полная неопределенность с дальнейшим развитием.

Здесь необходимо отметить схожую по идеологии инициативу от Mozilla Labs - Web Activities [23]. Дальнейший статус этой инициативы также неясен.

Следующий, недооцененный на наш взгляд, инструмент – это локальный веб-сервер. Первые реализации, насколько нам известно, относились к работам Nokia [24]. В оригинальной работе Nokia это был перенос сервера Apache на платформу S60. На наш взгляд, это одно из наиболее перспективных направлений не только для задач сетевой близости, а вообще, для взаимодействия с любыми сенсорами телефона из веб-приложений. Идея состоит в том, что опрос значений сенсоров может выполняться CGI-скриптом или его аналогом (Java сервлетом, например), а данные выводить в формате JSONP. Для мобильного разработчика подключение такого опроса выглядит как включение некоторого JavaScript файла. Например:

<script type="text/javascript" src="http://localhost: 8080/getNetworks?callback=f" ></script>

Где *f* – имя функции, которая вызывается при завершении опроса и куда передается, например, описанный выше JSON массив с информацией о сетевых узлах. Здесь явно эксплуатируется тот факт, что, несмотря на то, что мы обычно говорим о программном интерфейсе по доступу к сенсорам, на практике, для большинства задач, это только чтение данных. Вместо API (Application Program Interfaces) следовало бы говорить о DPI (Data Program Interfaces). В данном случае (network proximity), это, очевидно именно чтение данных (опрос видимых сетей). Впрочем, подобное верно и для других задач контекстно-зависимого программирования. Поэтому, попытки создать некоторый всеобъемлющий API, подобно упомянутому выше Web Bluetooth из Firefox OS, на самом деле сильно замедляют процесс, поскольку создают, например, не существующие для данного класса задач проблемы с безопасностью и уделяют большое внимание никак не используемым возможностям.

Отметим также, что такого рода подключение опять-таки позволяет говорить о едином источнике данных. При доступе к такого рода сайту с устройства, где локальный веб-сервер отсутствует, соответствующий фрагмент просто не будет обрабатываться (выводиться пользователю).

IV. Заключение

В работе рассматривается вопрос использования информации о сетевом окружении при создании динамических веб-страниц. Предложено несколько подходов к реализации мобильного браузера, который может оперировать данными о сетевом окружении (сетевой близости) для представления пользователям информации, привязанной к текущему контексту. Также рассмотрены детали возможные детали реализации.

Библиография

[1] Holdener, Anthony T. HTML5 Geolocation. O'Reilly Media, Inc., 2011.

[2] YOU, L., & PENG, Q. (2010). Research and implementation of geocoding based on Google Maps API [J]. Journal of Hubei University (Natural Science), 2, 161-164.

[3] Schiller J., Voisard A. (ed.). Location-based services. – Elsevier, 2004.

[4] Дворкина Н.Б., Намиот Д.Е. ИСПОЛЬЗОВАНИЕ OPENCELLID API В МОБИЛЬНЫХ СЕРВИСАХ //Прикладная информатика 2010 No. 5 стр. 92-101.

[5] Громаков Ю.А., Северин А.В., Шевцов В.А. Технологии определения местоположения в GSM и UMTS. М.: Эко – Трендз, 2005.

[6] Бильчук А., Намиот Д. Geo Messages для Java телефонов //International Journal of Open Information Technologies. – 2013. – Т. 1. – №. 1. – С. 16-20.

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

[8] 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.

[9] Deng, Zhongliang, et al. "Situation and development tendency of indoor positioning." Communications, China 10.3 (2013): 42-55.

[10] Sharma, P., Xu, Z., Banerjee, S., & Lee, S. J. (2006). Estimating network proximity and latency. ACM SIGCOMM Computer Communication Review, 36(3), 39-50.

[11] Namiot, D. (2013). Network Proximity on Practice: Context-aware Applications and Wi-Fi Proximity. International Journal of Open Information Technologies, 1(3), 1-4.

[12] Cheng, Y. C., Chawathe, Y., LaMarca, A., & Krumm, J. (2005, June). Accuracy characterization for metropolitan-scale Wi-Fi localization. In Proceedings of the 3rd international conference on Mobile systems, applications, and services (pp. 233-245). ACM.

[13] 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

[14] 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.

[15] А.А. Волков, Д.Е. Намиот, М.А. Шнепс-Шнеппе. О задачах создания эффективной инфраструктуры среды обитания //International Journal of Open Information Technologies. – 2013. – Т. 1. –No. 7. –С. 1-10.

[16] Namiot D., Schneps-Schneppe M. Smart cities software from the developer's point of view // 6 th International Conference on Applied Information and Communication Technologies (AICT2013). — LUA Jelgava, Latvia, 2013. — P. 230–237.

[17] Намиот Д., Сухомлин В. О проектах лаборатории ОИТ //International Journal of Open Information Technologies. – 2013. – Т. 1. – №. 5. – С. 18-21.

[18] HTML5 Geolocation http://www.w3schools.com/html/html5_geolocation.asp Retrieved: Jan, 2014

[19] Amatya, S., & Kurti, A. (2014). Cross-Platform Mobile Development: Challenges and Opportunities. In ICT Innovations 2013 (pp. 219-229). Springer International Publishing.

[20] Firefox Bluetooth Web API: https://wiki.mozilla.org/WebAPI/WebBluetooth Retrieved: Jan, 2014

[21] Billock, G., Hawkins, J., & Kinlan, P. (2012). Web Intents. Internet: http://dvcs. w3. org/hg/web-intents/raw-file/tip/spec/Overview. html.

[22] Namiot, D., & Sneps-Sneppe, M. (2012, April). Proximity as a service. In Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on (pp. 199-205). IEEE.

[23] Web Activities http://blog.mozilla.org/labs/2011/07/web-apps-update-experiments-in-web-activities-app-discovery/ Retrieved: Jan, 2014

[24] Local Mobile Web server: https://research.nokia.com/page/231 Retrieved: Jan, 2014

Local web pages

Dmitry Namiot

Abstract— **This article describes the usage of context-aware programming for development and deployment of web sites (web pages) devoted to the description of any objects (events) with precisely delineated geographic scope. In this paper we propose mechanisms to create mobile web applications, the content of which is attracted to a particular geographic area. The accuracy of such a binding allows us to distinguish individual areas within the same indoor space.**

Keywords— **context aware, network proximity, indoor positioning, HTML5.**