Обмен данными между мобильными устройствами без организации соединений

Е.А. Трушкина, Д.Е. Намиот

***Аннотация*—В статье рассматривается один подход к организации обмена данными между близко расположенными мобильными устройствами, который не требует организации прямого соединения между устройствами и не использует сторонних (облачных) хранилищ. Статья написана по результатам выпускной квалификационной работы, выполненной на факультете ВМК МГУ имени М.В. Ломоносова. В работе предложено мобильное приложение для платформы Android, основанное на модели сетевой пространственной близости, которое позволяет обмениваться контактной информацией с другими мобильными устройствами, расположенными поблизости. В рамках этой модели, для определения пространственной близости используется ограниченная область распространения сигналов беспроводных сетей. В силу полного исключения работы с гео-координатами, такие сети не обязательно могут быть стационарными узлами с известным местоположением. Основной моделью использования является программное создание таких беспроводных сетей, специально для предоставления сервисов, использующих информацию о местоположении. При этом рекламная (представительская) информация таких узлов может быть использована для передачи пользовательских данных.**

***Ключевые слова*—сетевая пространственная близость, Bluetooth.**

I. Введение

Эта статья написана по результатам выпускной квалификационной работы, выполненной на факультете ВМК МГУ имени М.В. Ломоносова.

В работе была представлена оригинальная реализация (мобильный сервис) обмена данными между двумя близко расположенными мобильными устройствами на платформе Android. В целом такие сервисы достаточно известны. Два устройства могут использовать Bluetooth (Wi-Fi Direct) соединение для передачи данных. Можно воспользоваться каким-нибудь промежуточным облачным сервером, куда одно устройство (приложение) выгружает данные, а второе – скачивает. Для небольшого объема данных можно представить динамически созданный QR-код на одном устройстве, который прочитает приложение на другом устройстве. Последний вариант, по крайней мере, не требует сетевых соединений.

В данном случае обмен информацией был построен на модели сетевой пространственной близости. Это новая архитектура для сервисов, использующих информацию о местоположении. В рамках данной архитектуры полностью исключается работа с гео-координатами, которая заменяется на определение пространственной близости. Идея состоит в том, что когда мы в гео-сервисах описываем данные (услуги) в привязке к гео-координатами, подавляющее большинство гео-запросов состоит именно в подборе данных (услуги) на основе пространственной близости: сервис вычисляет расстояние, используя координаты клиента и координаты, ассоциированные с данными. Далее, если это расстояние не превышает некоторого предела (естественно, зависящего от сервиса), соответствующие данные возвращаются клиенту. То есть гео-координаты на самом деле используются только для вычисления пространственной близости. Модель сетевой пространственной близости как раз и предлагает заменить работу с гео-координатами непосредственным вычислением пространственной близости. В качестве способа измерения пространственной близости предлагается использовать ограниченную область распространения сигнала беспроводных сетей.

Соответственно, данные (услуги, сервисы) вместо гео-координат предполагается ассоциировать с существующими или специально созданными узлами беспроводных сетей. Доступность (видимость) беспроводного узла со стороны клиента и будет свидетельствовать о пространственной близости. При этом, поскольку гео-координаты ни в какой форме не участвуют в этом процессе, можно динамически создавать беспроводные узлы специально для задач фиксации близости (например, программно создать точку Bluetooth). Истинное местоположение таких узлов неизвестно, но важно, что поблизости от них можно предоставлять услуги (данные), привязанные к их текущему местоположению. Если мы будем использовать несколько беспроводных узлов, то пересечение их областей видимости позволяет моделировать гео-решетку (рис. 1). Собственно говоря, этот факт позволяет моделировать, на практике, все существующие гео-сервисы.

Статья получена 9 июля 2022.
Трушкина Е.А. – МГУ имени М.В. Ломоносова (email: trushkinaevg@gmail.com).
Намиот Д.Е. – МГУ имени М.В. Ломоносова (email: dnamiot@gmail.com)

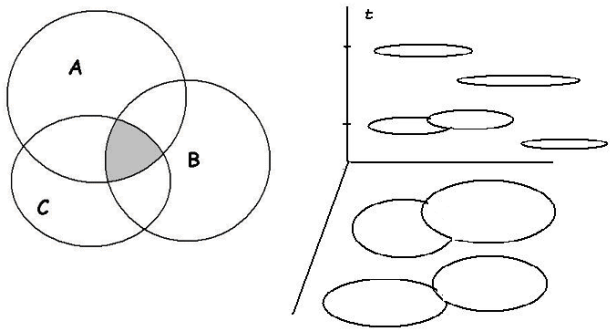

Рис.1. Области видимости и их изменение во времени

При этом, определенные таким образом сервисы, будут как динамическими (беспроводной узел может быть включен или выключен), так и мобильными (беспроводной узел, определяющий сервис, может перемещаться) [1].

Следующий момент – это то, что означает определение “видимости” беспроводного устройства. С технической точки зрения – это получение рекламной информации беспроводного узла. Например, получение SSID беспроводного узла – это и есть фиксация его видимости. То есть мы имеем дело с классической киберфизической системой: физический процесс распространения сигнала и его программное измерение. Рекламная информация беспроводного узла может быть кастомизирована. Это означает, что вместе с фиксацией близости можно получить некоторую сервисную информацию. Это позволяет реализовывать сервисы, которые вообще не имеют серверной (облачной) составляющей. Клиент, попавший в зону сервиса, одновременно получит и сервисную информацию [2].

Отметим, что по такому принципу, например, в 2020 году была реализована совместная спецификация Google и Apple Covid tracker [5].

Модель сетевой пространственной близости была разработана на факультете ВМК МГУ имени М.В. Ломоносова [3,4] и использовалась во многих квалификационных работах и магистерских диссертациях [6, 7].

В данной работе рассматриваются модели информационного вещания посредством методов сетевой пространственной близости для мобильных устройств, их практическое применение, а также разработанное в рамках выпускной работы мобильное приложение «Quack!» для ОС Android, которе представляет собой так называемый цифровой бэдж, что является, по сути, заменой бумажной карточки: такой тип приложений является физическим расширением социальных сетей.

Целью работы является анализ моделей, методов и технологий сетевой пространственной близости и их применение, а также разработка мобильного, позволяющего обмениваться контактами peer2peer без подключения к Интернет и использования какой-либо серверной инфраструктуры, как доказательство рассматриваемого нами подхода.

Разрабатываемое мобильное приложение поддерживает следующие функции:

- сканирование контактов вокруг мобильного устройства и отображение их в виде списка с возможностью просмотра детальной информации по каждому контакту;
- возможность заполнить и сохранить контактную информацию о себе в виде профиля;
- вещание своего профиля с контактной информацией посредством технологий сетевой пространственной близости.

Работа содержит примеры программного кода на языке Kotlin для ОС Android, которые иллюстрируют концепции и способы разработки полноценного приложения с поддержкой технологий сетевой пространственной близости.

II. Обзор технологий

A. Концепция сетевой пространственной близости

Сетевая близость – это свойство пространственной близости, которая определяется видимостью одного сетевого устройства другим. Радиус действия сетевого устройства ограничен, поэтому его видимость другими устройствами является некоторой характеристикой местоположения этих устройств. В зависимости от этого местоположения пользователю предоставляется контекстно-зависимая информация и дополнительная функциональность. Системы, основанные на принципе сетевой близости, используют сетевые интерфейсы мобильных устройств как датчики расстояния.

С программной точки зрения, обнаружение беспроводного устройства - это получение его рекламной информации. Именно так работает, например, поиск соседних узлов в беспроводных сетях. Поиск соседних узлов - это определение всех узлов в сети, с которыми данный узел может напрямую связываться. Очевидно, что узлы должны использовать (передавать) некоторую идентификацию во время процесса обнаружения. И идея сетевой пространственной близости состоит именно в том, чтобы использовать эту идентификационную информацию (или некоторые надстройки для этой информации) для передачи пользовательских данных.

Ограниченная область распространения сигнала беспроводных сетей как раз и служит основанием для определения близости. В этом отличие, например, от систем позиционирования, которые используют информацию о беспроводных сетях. Все они, в той или иной форме, используют информацию о предварительной разметке сигналов беспроводных сетей относительно узлов, с известным местоположением. Отсутствие же требований к определению местоположения позволяет использовать для определения сетевой пространственной близости не

только любые существующие узлы беспроводных сетей (точки доступа Wi-Fi, теги Bluetooth и т.д.), но и создавать такие узлы специально для задач определения близости. В последнем случае можно произвольно задавать идентификацию таких узлов (их рекламное представление), а также обновлять ее динамически. Например, устройство Bluetooth Low Energy в режиме рекламы (рекламодатель) периодически передает рекламную информацию в трех каналах представления и слушает, ожидая ответа от других устройств. С другой стороны, устройство в режиме сканирования (так называемый сканер), периодически сканирует каналы представления (рекламные каналы) и слушает рекламную информацию от других устройств. Инициаторы отличаются только тем, что они могут отвечать лишь на конкретные типы рекламных пакетов.

Таким образом, в модели сетевой пространственной близости гео-координаты заменяются идентификацией узлов беспроводных сетей. Соответственно, проверка близости, классически представленная как сравнение координат, в новой модели представляется как проверка видимости (доступности) определенных узлов беспроводных сетей. А эта проверка, в свою очередь, заключает в фиксации получения рекламы (идентификации) беспроводного узла. Помимо рекламы беспроводного узла, которая фиксирует факт его доступности (видимости), могут учитываться и другие его доступные характеристики. Например, сила сигнала (RSSI), направление (для Bluetooth 5) и т.д. Эти характеристики могут учитываться в условиях для предоставления сервисов, учитывающих информацию о местоположении как дополнительные условия.

B. Анализ протокола Wi-Fi Direct

Wi-Fi Direct – это новая технология, разрабатываемая Wi-Fi Alliance и направленная на улучшение прямой связи между устройствами в Wi-Fi. Учитывая широкую базу устройств с возможностями Wi-Fi, а также тот факт, что она может быть полностью реализована программно поверх традиционных источников Wi-Fi, ожидается, что эта технология будет иметь значительное влияние.

Более чем через десять лет после своей разработки стандарт IEEE 802.11 стал одним из самых распространенных способов доступа в Интернет. Однако для того, чтобы продолжить свой поразительный успех, технология Wi-Fi должна развиваться и охватывать все большее количество вариантов использования. Учитывая широкое распространение Wi-Fi во многих типах устройств, естественным путем развития технологии является обеспечение связи между устройствами, т.е. без необходимости присутствия точки доступа (AP), что традиционно обеспечивается другими технологиями [17]. Именно с этой целью и была разработана технология Wi-Fi Direct, созданная Wi¬Fi Alliance.

Прямое соединение между устройствами уже было возможно в первоначальном стандарте IEEE 802.11 с помощью режима работы ad hoc. Однако он так и не получил широкого распространения на рынке и, следовательно, имеет ряд недостатков, когда сталкивается с современными требованиями, например, отсутствие эффективной поддержки энергосбережения или расширенных возможностей QoS. Другой актуальной технологией в области коммуникаций между устройствами Wi-Fi является 802.11z, также известная как Tunneled Direct Link Setup (TDLS), которая обеспечивает прямую связь между устройствами, но требует, чтобы станции были связаны с одной и той же точкой доступа.

В отличие от предыдущих технологий, технология Wi-Fi Direct использует другой подход для улучшения связи между устройствами. Вместо того чтобы использовать режим работы ad-hoc, Wi-Fi Direct основывается на успешном режиме инфраструктуры IEEE 802.11 и позволяет устройствам договариваться о том, кто возьмет на себя функции, подобные точке доступа. Таким образом, устаревшие устройства Wi-Fi могут без проблем подключаться к устройствам Wi-Fi Direct. Принимая такое решение, Wi-Fi Direct немедленно наследует все улучшенные механизмы QoS, энергосбережения и безопасности разработанные для инфраструктурного режима Wi-Fi за последние годы.

Устройство однозначно ведет себя либо как точка доступа, либо как клиент, каждая из этих ролей включает в себя различный набор функциональных возможностей. Основная новизна Wi-Fi Direct заключается в том, что эти роли задаются как динамические, и, следовательно, устройство Wi-Fi Direct должно реализовывать как роль клиента, так и роль точки доступа (иногда называемой Soft-AP). Таким образом, эти роли являются логическими ролями, которые могут даже выполняться одновременно одним и тем же устройством, например, путем использования различных частот (если устройство имеет несколько физических радиостанций) или разделения канала по времени с помощью методов виртуализации. Для того чтобы установить связь, устройства P2P должны договориться о роли, которую каждое устройство возьмет на себя.

Устройства Wi-Fi Direct, формально известные как устройства P2P, взаимодействуют путем создания групп P2P, которые функционально эквивалентны традиционным инфраструктурным сетям Wi-Fi. Устройство, реализующее AP-подобную функциональность в P2P Group, называется P2P Group Owner (P2P GO), а устройства, действующие как клиенты, называются P2P Clients. Учитывая, что эти роли не являются статичными, когда два устройства P2P обнаруживают друг друга, они "договариваются" о своих ролях (P2P Client и P2P GO) для создания P2P Group. После создания группы P2P другие клиенты P2P могут присоединиться к ней, как в традиционной сети Wi-Fi. Устаревшие клиенты также могут взаимодействовать с P2P GO, если они не являются устройствами 802.11b и поддерживают необходимые

механизмы безопасности. Таким образом, устаревшие устройства формально не принадлежат к группе P2P и не поддерживают расширенные функциональные возможности, определенные в Wi-Fi Direct, но они просто "видят" P2P GO как традиционную точку доступа.

Логическая природа ролей P2P поддерживает различные архитектурные развертывания, два из которых показаны. Например, одна группа может быть создана мобильным телефоном, разделяющим свое 3G-соединение с двумя ноутбуками; для этой первой группы телефон действует как P2P GO, а два ноутбука - как P2P клиенты. Чтобы расширить сеть, один из ноутбуков создает вторую P2P-группу с принтером; для этой второй группы ноутбук действует как P2P GO. Для того чтобы действовать одновременно как P2P-клиент и как P2P GO, ноутбук обычно чередует эти две роли, разделяя по времени Wi-Fi интерфейс; позже мы представим протокол NoA, который может быть использован для этой цели

Как и традиционная точка доступа, P2P GO объявляет о себе с помощью маячков и должен поддерживать услуги энергосбережения для своих клиентов. P2P GO также обязан запускать сервер протокола динамической конфигурации хоста (DHCP) для предоставления P2P-клиентам IP-адресов. Кроме того, только P2P GO имеет право подключать устройства в своей P2P группе к внешней сети (например, 3G сети или инфраструктурной WLAN) и для этого соединения не допускается использование мостов. Поэтому подключение должно осуществляться как правило на сетевом уровне.

Как только два устройства P2P нашли друг друга друг друга, они начинают фазу GO Negotiation. Для этого используется трехстороннее рукопожатие, а именно запрос GO Negotiation Request/Response/Confirmation, в ходе которого два устройства договариваются о том, какое устройство будет действовать как P2P GO, и о канале, на котором будет работать группа, который может находиться в диапазоне 2,4 ГГц или 5 ГГц. Чтобы договориться об устройстве, которое будет действовать как P2P GO, устройства P2P посылают числовой параметр, значение GO Intent, в рамках трехстороннего рукопожатия, и устройство, объявившее наибольшее значение, становится P2P GO. Для предотвращения конфликтов, когда два устройства объявляют одинаковые GO Intent, в GO Negotiation Request включен бит tiebreaker, который случайным образом устанавливается каждый раз, когда посылается GO Negotiation Request.

После того, как устройства обнаружили друг друга и договорились о соответствующих ролях, следующей фазой является установление безопасной связи с помощью Wi-Fi Protected Setup, которая обозначается как фаза WPS Provisioning, и, наконец, обмен DHCP для настройки IP-конфигурации (фаза Address config).

Устройство P2P может автономно создать группу P2P, где оно сразу же становится P2P GO, устанавливая канал и начиная вещание. Другие устройства могут обнаружить созданную группу, используя традиционные механизмы сканирования, а затем непосредственно перейти к фазам WPS Provisioning и Address Configuration. По сравнению с предыдущим случаем, в этом случае фаза обнаружения упрощается, так как устройство, создающее группу, не переходит из одного состояния в другое, и фаза GO Negotiation не требуется.

Во время процесса формирования P2P устройства могут объявить группу постоянной, используя флаг в атрибуте P2P Capabilities, предварительно отправленный в кадры маяка, ответы зонда и кадры GO negotiation. Таким образом, устройства, формирующие группу, сохраняют сетевые учетные данные и назначенные роли P2P GO и Client для последующих повторных созданий P2P группы. В частности, после фазы обнаружения, если устройство P2P распознает, что в прошлом оно уже сформировало постоянную группу с соответствующим аналогом, любое из двух устройств P2P может использовать процедуру приглашения (двухстороннее рукопожатие) для быстрого повторного создания группы.

Важной особенностью Wi-Fi Direct является возможность поддержки обнаружения услуг на канальном уровне. Таким образом, до создания группы P2P, устройства P2P могут обмениваться запросами для обнаружения набора доступных услуг и, основываясь на этом, решать, продолжать ли формирование группы или нет. Это представляет собой значительный сдвиг по сравнению с традиционными сетями Wi-Fi, где предполагается, что единственной услугой, в которой заинтересованы клиенты, является подключение к Интернету.

Для того чтобы реализовать вышеизложенное, запросы на обнаружение услуг, генерируемые протоколом более высокого уровня, например, UPnP или Bonjour [11], передаются на канальный уровень с помощью протокола Generic Advertisement Protocol (GAS), определенного 802.11u [18]. GAS - это протокол запросов/ответов канального уровня, реализованный с помощью публичных кадров действий, который позволяет двум неассоциированным устройствам 802.11 обмениваться запросами, принадлежащими протоколу более высокого уровня (например, протоколу обнаружения услуг). GAS реализуется с помощью общего контейнера, который обеспечивает фрагментацию и повторную сборку, а также позволяет устройству-получателю идентифицировать передаваемый протокол более высокого уровня.

C. Анализ протокола Bluetooth Low Energy

Появившийся в 2010 году, Bluetooth Low Energy (BLE, иногда называется как Bluetooth Smart) был создан для разработки радиостандарта с минимально возможным энергопотреблением, специально оптимизированного для низкой стоимости, низкой пропускной способности, низкой мощности и низкой сложности [19]. Хотя BLE является частью протокола

Bluetooth, начиная с Bluetooth 4.0, эти две технологии имеют мало общего, кроме частотного диапазона.

BLE быстро развивается по сравнению с другими стандартами беспроводной связи. В 2014 году на долю BLE приходилось 85% беспроводных радиоприемников, используемых в коммерческих устройствах [20], причем успех этой технологии обеспечили такие крупные игроки, как Apple и Samsung. Он имеет более компактный стек протоколов, чем у классического Bluetooth, нет затрат на лицензирование, и нет платы за доступ к спецификации. Поэтому он становится стандартом де-факто для беспроводной связи между небольшими устройствами.

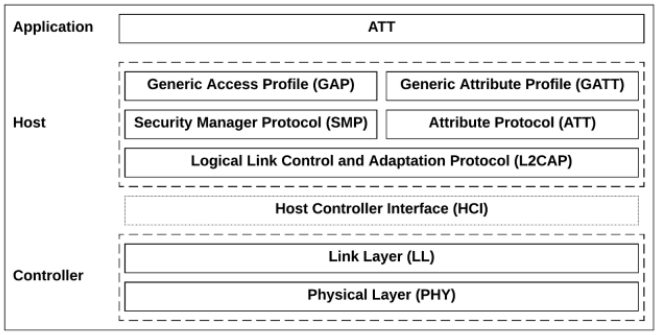

Рис. 2. Стек протоколов BLE

Стек протоколов BLE является основным фактором энергоэффективности и одним из ключей к успеху. Хотя он имеет такое же количество уровней, как и классический Bluetooth, их реализация намного проще. Стек протоколов разделен на 3 части: контроллер, хост и приложение. Ниже приводится краткое описание двух нижних частей и содержащихся в них уровней.

Контроллер: Эта группа включает в себя самые нижние уровни в стеке протоколов и осуществляет фактический обмен данными между устройствами BLE.

Физический уровень обеспечивает фактическую радиосвязь в диапазоне 2,4 ГГц ISM (Industrial, Scientific, and Medical). Он делит этот диапазон на 40 каналов, 37 из которых используются для данных соединения, а остальные 3 - для адвертайзинга. BLE использует очень простую технику перескока частоты, чтобы минимизировать влияние радиопомех, потенциально присутствующих в этом диапазоне, которые могут включать WiFi и классический Bluetooth.

Переход на новый канал при каждом событии соединения осуществляется по следующей формуле:

Новый канал = (Текущий канал + Приращение прыжка) mod 37

Приращение прыжка устанавливается ведущим устройством на случайное значение между 5 и 16. Это гарантирует, что каждый канал будет использоваться ровно один раз в каждом цикле, поскольку 37 - простое число.

Канальный уровень взаимодействует непосредственно с физическим уровнем и является единственным уровнем в стеке с жесткими ограничениями реального времени, поскольку он обрабатывает все строгие требования протокола по времени. Как таковой, он обычно реализуется как смесь пользовательского аппаратного и программного обеспечения и отделен от других уровней стандартным интерфейсом. Реализация канального уровня каждым производителем является уникальной, закрытой системой, и в настоящее время не существует реализации с открытым исходным кодом.

Некоторые функции канального уровня обычно реализуются аппаратно, например, генерация случайных чисел и шифрование AES. Программная часть управляет такими свойствами, как роли и состояние канала радиосвязи. Устройство может работать в одной из этих ролей:

- Адвертайзер - просто выполняет адвертайзинг пакетов;
- Сканер - регулярно прослушивает рекламные пакеты.;
- Ведущий - инициирует соединение и затем управляет им;
- Ведомое устройство - принимает запрос на соединение и затем следует таймингу ведущего устройства;

Кроме ограничений реализации, в протоколе нет ничего, что могло бы помешать устройству взять на себя любую из этих ролей, и даже более одной одновременно.

Ведомое устройство может посылать 4 типа пакетов адвертайзинга, определяемых этими тремя свойствами: Connectable, Scannable и Directed. Connectable означает, что сканер может инициировать соединение после получения рекламного пакета. Сканируемый означает, что сканер может отправить запрос на сканирование после получения пакета. Направленный означает, что он нацелен на определенный сканер. Направленный рекламный пакет не может нести полезную нагрузку (пользовательские данные), в то время как ненаправленный может. Направленные пакеты, таким образом, могут только приглашать к соединению, и поэтому они всегда соединяемы.

Для роли сканера спецификация BLE определяет два типа процедур: пассивное сканирование и активное сканирование. Пассивное сканирование означает, что сканер просто прослушивает широковещательные рекламные пакеты, при этом рекламодатель не знает, были ли получены пакеты или нет. Активное сканирование, с другой стороны, позволяет сканеру передавать пакет Scan Request после получения рекламного пакета. Этот Scan Request, в свою очередь, побуждает рекламодателя ответить на него метким названием Scan Response, что позволяет отправлять дополнительные данные по адвертайзингу.

Этот метод, однако, не позволяет сканеру отправлять какие-либо данные адвертайзеру, поскольку спецификация диктует фиксированную полезную нагрузку для пакета Scan Request.

Интерфейс хост-контроллер: Спецификация Bluetooth допускает физическое разделение хоста и контроллера на разных чипах. Это часто имеет смысл для больших устройств, чтобы иметь выделенный процессор для контроллера, из-за его жестких требований реального времени и доступа к физическому уровню. Интерфейс хост-контроллера обеспечивает связь между хостом и контроллером через последовательное соединение. Он определяется как набор команд и событий, которые разделяются между двумя частями, а также несколько транспортов для фактической связи (например, UART, USB, SDIO и т.д.). Если обе части находятся на одном кристалле (так называемая система на кристалле, или SoC), эта часть не используется.

Хост: Эта группа состоит из следующих уровней:

Протокол управления логическим каналом и адаптации служит мультиплексором для верхних уровней, инкапсулируя их пакеты в стандартный формат пакетов BLE. Он также обрабатывает фрагментацию и рекомбинацию, разбивая большие сообщения на фрагменты, которые соответствуют максимальному размеру полезной нагрузки для передачи, а затем рекомбинируя их на принимающей стороне.

Протокол атрибутов управляет аспектами клиент-сервер протокола BLE. Клиент отправляет запрос на сервер для чтения или записи атрибута, а сервер отвечает запрошенным значением или разрешением на запись.

Менеджер безопасности, как следует из названия, обрабатывает аспекты безопасности соединения (например, начальный обмен ключами и зашифрованную передачу) и скрывает публичный адрес Bluetooth, если требуется, чтобы избежать отслеживания устройства.

Профиль Generic Attribute Profile основывается на протоколе Attribute Protocol, добавляя иерархию и абстракцию данных, которые определяют, как данные организуются и обмениваются между приложениями. По сути, он обеспечивает стандартизированную структуру для обмена информацией между приложениями одного типа. Например, существуют официальные профили для измерения пульса и температуры тела, что облегчает разработчикам создание приложений, работающих с готовыми датчиками.

Профиль общего доступа обеспечивает основу для обнаружения устройствами других устройств, передачи данных, установления безопасных соединений и выполнения других низкоуровневых операций согласованным образом. В частности, этот элемент определяет роли, которые устройство может играть в сети, режимы, в которых оно может работать в рамках этих ролей, и процедуры, доступные в каждом режиме.

Четыре роли, которые может принимать устройство:

- Роль вещатель соответствует роли адвертайзера канального уровня и используется, когда устройства просто периодически посылают рекламные пакеты с данными, как, например, термометр, передающий температуру. Эти данные свободно доступны любому устройству, которое слушает, и устройства в этой роли не могут получать никаких данных.

- Роль наблюдателя соответствует роли сканера канального уровня и используется приложениями для сбора данных, посылаемых широковещательными устройствами.

- Роль центрального устройства соответствует роли ведущего устройства канального уровня и позволяет устройству инициировать и устанавливать соединения с несколькими периферийными устройствами. Протокол BLE является асимметричным, поскольку требования к вычислительным ресурсам и мощности для этой роли выше, чем для роли периферийного устройства, особенно если она управляет несколькими устройствами.

- Роль периферийного устройства соответствует роли ведомого устройства канального уровня. Эта роль сначала отвечает за рекламу своего присутствия, пока Центральное устройство не соединится с ней, а затем за поддержание времени, необходимого для регулярной связи с Центральным устройством. Хотя требования к обработке и памяти устройства минимальны, его строгие временные требования предполагают наличие постоянного источника питания.

Режимы – это состояния, в которые устройство может переходить в рамках роли для выполнения определенной процедуры. Процедуры - это последовательности действий, которые устройство может предпринять в том или ином режиме для выполнения задачи, такой как вещание, наблюдение или установление соединения.

III. Обзор существующих работ

Классическая модель работы мобильных приложений, которые используют информацию о позиционировании, состоит в отправке полученных на клиентской стороне данных на сервер для последующей обработки [21]. Отказ от серверной инфраструктуры в предлагаемом подходе явным образом обеспечивает приватность. В данном случае, например, рассылка профиля в социальной сети будет доступна только тем мобильным абонентам, которые в данный момент времени видят (могут видеть) владельца профиля. А какая-либо история таких рассылок просто отсутствует.

Компания Facebook подала заявку на патент [22] на способ предсказания местоположения пользователя социальной сети по данным о сигналах беспроводных сетей (Wi-Fi, Bluetooth, сотовых сетей и NFC). Суть способа состоит в том, что приложение, установленное на устройстве пользователя и связанное с аккаунтом пользователя в социальной сети, в фоновом режиме собирает данные о сигналах беспроводных сетей, получаемых устройством, и отправляет их на удаленный сервер вместе с геопозицией устройства. Сервер собирает эти данные в базу и проводит корреляции между идентификаторами сетей и геопозицией устройства. Таким образом, имея базу идентификаторов сетей и их геопозиций, можно определять

местоположение другого устройства, которое также отправляет в фоновом режиме идентификаторы беспроводных сетей. Это позволит определять местоположение устройств пользователей, которые запретили приложению социальной сети доступ к геопозиции. В заявке предполагается, что к местоположению можно будет привязывать лейблы, события или социальные действия. Как видно, здесь опять присутствует центральное хранилище.

Наиболее очевидным способом применения сетевой близости является использование системы навигации внутри помещений. Как правило, инфраструктура для таких систем создается специально. Для этих целей хорошо подходят Bluetooth-маячки, работающие по протоколу BLE. Эти маячки имеют компактные размеры и способны к длительной автономной работе, система из таких маячков позволяет определять местоположение с точностью до 1-2 м [23]. Маячки размещаются на стационарных позициях внутри помещения, их координаты заносятся в базу данных. Мобильное приложение сканирует находящиеся в пределах сетевой близости маячки и отправляет эту информацию на сервер, где происходит вычисление координат устройства, либо вычисление местоположения происходит на самом устройстве по заранее загруженной карте маячков. Существуют различные техники для вычисления местоположения по сигналам беспроводных сетей: простая замена концепции местоположения концепцией близости, трилатерация [24] и фингерпринтинг (информация о сетевом окружении). Помимо использования BLE для определения местоположения можно использовать другие технологии. Использование сигналов сетей Wi-Fi позволяет определять местоположение с точностью до нескольких десятков метров. В целом, вопрос использования сигналов беспроводных сетей для определения местоположения устройства является достаточно хорошо проработанным.

Контексто-зависимые приложения, основанные на сетевой близости, используются в маркетинге для доставки на устройство пользователя рекламных сообщений, привязанных к местоположении. В работе [25] представлена система для рекламы в торговых центрах, основанная на сетевой близости. Специальное приложение, устанавливаемое на смартфон посетителя торгового центра, сканирует Bluetooth-теги, привязанные к магазинам торгового центра. По силе сигнала (RSSI) приложение выбирает ближайший магазин, соединяется с Bluetooth-устройством этого магазина и загружает с этого устройства рекламу, которую отображает пользователю. Здесь необходимо отметить два момента: во-первых, зависимость силы сигнала от расстояния не всегда такая простая, во-вторых, и это главное, здесь речь идет опять-таки о соединении.

В работе [26] предлагается модель Bluetooth Data Points, не требующая установления соединения с Bluetooth-устройством - контент, привязанный к идентификатору Bluetooth-устройства, хранится в сети Интернет, а факт обнаружения узла Bluetooth является триггером для того, чтобы приложение запросило и отобразило этот контент. В качестве узла Bluetooth, к идентификатору которого привязывается контент, в такой системе может выступать обычный мобильный телефон. В работе [27] предложено привязывать правила доставки сообщений не только к отдельным Bluetooth- тегам, но и к слепкам, содержащим список одновременно обнаруженных Bluetooth-устройств и дополнительную контекстную информацию. При срабатывании правила на устройство пользователя отправляется push-уведомление, содержащее как сам контент, так и указание на способ обработки данного контента. Аналогичные системы можно строить на базе точек доступа Wi-Fi. Так, в работе [28] предложен сервис SpotEx, который по слепкам обнаруженных точек доступа Wi-Fi на основе заложенных в нем правил отправляет мобильному приложению локально-значимую информацию. Перечисленные системы можно использовать не только в розничной торговле, но и в других сферах, создавая сервисы, основанные на местоположении.

Сетевая близость также используется для подтверждения нахождения пользователя в той или иной локации, как это делается в системах для контроля за посещаемостью занятий студентами, разработанных для того, чтобы отмечать присутствие студента в конкретной аудитории в конкретное время.

Концепцию сетевой близости применяют для создания мобильных интерфейсов к объектам физического мира. Так, компания Google на основе протокола BLE разрабатывает Physical Web [29] - технологию для создания web-интерфейса к объектам физического мира.

Теги могут быть прикреплены к рекламным постерам, витринам, дисплеям. Пользователь может отсканировать тег при помощи мобильного приложения, связанного с его аккаунтом в социальной сети. Мобильное приложение отображает таргетированную рекламу товара или услуги или иную информацию на основе данных профиля пользователя. Пользователь может мгновенно заказать понравившийся товар или услугу.

Прикрепленные к товарам или полкам магазина NFC-теги могут содержать идентификатор товара. Специальное мобильное приложение считывает идентификатор товара и запрашивает информацию о нем (цена, остаток на складе) у сервера, который связан с ERP-системой продавца.

Сетевая близость позволяет не только создавать интерфейсы к физическим объектам, но и исследовать характер взаимодействия пользователя с объектом.

Контекстом мобильного приложения могут служить и окружающие люди. Многие социальные сети

используют данные о геопозиции пользователя для показа ему друзей или возможных знакомых, находящихся поблизости, интересных событий, публикаций и фотографий, привязанных к местоположению, геотаргетированной рекламы. Выделяется особый тип социальных сетей - геосоциальные сети, или социальные сети, основанные на местоположении. В настоящее время эти приложения используют GPS данные мобильного устройства и требуют доступа к геопозиции устройства. Вместе с тем, модель сетевой пространственной близости предлагает здесь новый подход – физическое расширение социальных сетей [30].

В данной статье описывается мобильное приложение, при помощи которого пользователи посредством технологий сетевой близости смогут делиться контактной информацией – по типу vcard или “виртуальной визиткой” – напрямую между устройствами, без использования какой-либо серверной инфраструктуры, без установки соединения и без сканирования QR кодов. Исследованию способов построения такого программного решения на современных мобильных операционных системах и посвящен следующий раздел.

IV. Мобильное приложение «Quack!»

Приложение состоит из трех закладок:

Lake - список всех собранных вокруг контактов (рис. 3)
Nest - экран профиля пользователя (рис.4)
More - дополнительный экран с информацией об авторе и ссылке на Github с репозиторием проекта (рис. 5)

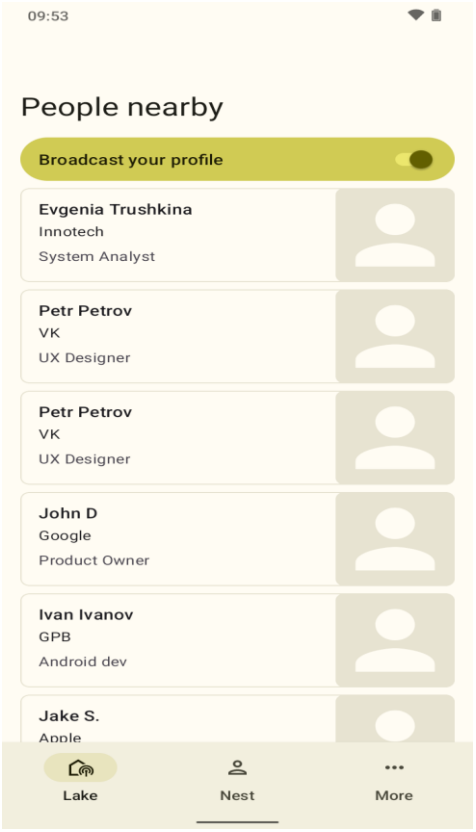

Рис. 3. Экран списка контактов поблизости

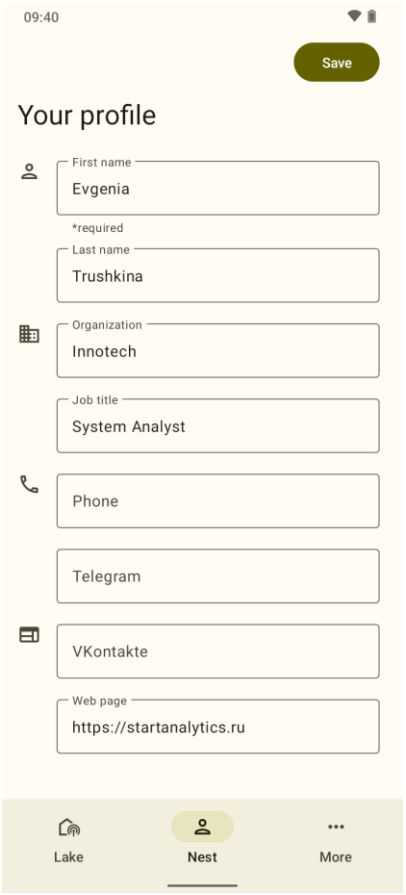

Рис. 4. Экран профиля пользователя

Рис. 5. Экран “Еще”

На экране “Lake” пользователь может выбрать интересующий его контакт и перейти на экран детальной информации о контакте. На экране детальной информации (рис. 6.) о контакте пользователь может просмотреть всю информацию о контакте, который другой пользователь заполнял на вкладке “Nest”, также переслать полученную информацию в любое другое приложение (шаринг).

Чтобы начать делиться своим контактом пользователь должен его заполнить на вкладке “Nest” и переключить переключатель “Broadcast your profile” на вкладке “Lake” в положение “ON”. Если пользователь нажимает на переключатель до заполнения профиля, то приложение предложит пользователю перейти на экран заполнения профиля. Также немаловажна проверка на разрешение на сканирование, доступность и состояние модуля Bluetooth на устройстве (рис. 7).

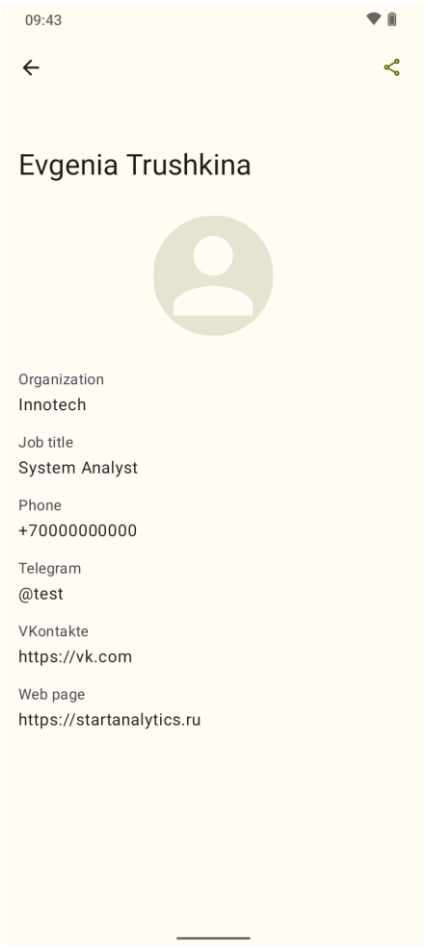

Рис. 6. Экран детальной информации о контакте

Само приложение написано на языке программирования Kotlin [31], в своей основе использует принципы чистой архитектуры (clean architecture), когда все приложение разделяется на независимые слои (в нашем случае: data, domain, presentation), и внедрения зависимостей (dependency injection, используется библиотека Hilt). Для работы в многопоточной среде выбраны инструменты Kotlin Coroutines и Kotlin Flow. Презентационный слой построен с использованием шаблона Model-View-ViewModel [32].

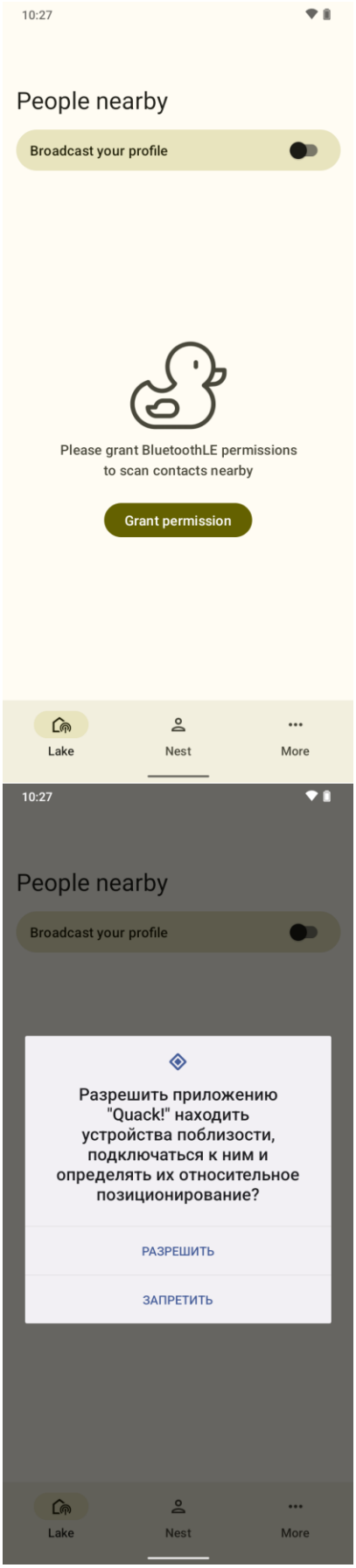

Рис. 7. Проверка разрешений в ОС Android

Самыми сложными и значимыми частями приложения являются модуль вещания и модуль сканирования. Они рассмотрены в следующем разделе. Исходный код приложений можно изучить по ссылке в Gitlab [15]. Само приложение доступно в Google Play по ссылке [16].

V. О программной реализации

A. *Разработка программного модуля вещания*

Для того, чтобы начать процесс вещания нужно получить экземпляр класса *BluetoothLeAdvertiser*. Ссылку на него мы можем получить из *BluetoothManager*, для его получения воспользуемся методом *getSystemService()* у *Context*:

```
  context.getSystemService(Context.BLUET
OOTH_SERVICE) as BluetoothManager
  Получение BluetoothLeAdvertiser:
  val       bluetoothLeAdvertiser       =
bluetoothManager?.adapter?.bluetoothLeAd
vertiser
  if (bluetoothLeAdvertiser == null) {
      Log.e("BleProximityRepository",
"bluetoothLeAdvertiser       is       not
available")
      throw BLENotAvailableException()
  }
  Далее для запуска вещания:
  val data = AdvertiseData.Builder()
      .setIncludeDeviceName(false)
      .setIncludeTxPowerLevel(false)
      .addServiceData(

uuidBleServiceHolder.getUuid(),

strData.toByteArray(Charsets.UTF_8)
      )
      .build()

  bluetoothLeAdvertiser.startAdvertising
(advertiseSettings,                 data,
advertisingCallback)
```

Разберем подробнее данный пример. Прежде всего, нужно подготовить данные к отправке и создать *AdvertiseData* при помощи *AdvertiseData.Builder()*. В методе *addServiceData()* мы указываем сгенерированный заранее UUID (вида “0DD0642A-D822-4DF5-AC1E-BCA17263F97D”) и данные для рекламы. При этом в методы *setIncludeDeviceName* и *setIncludeTxPowerLevel* мы передаем значение “false”, чтобы Android SDK не добавлял к advertisement пакету информацию об имени устройства и силе сигнала, так как это увеличит размер пакета, и мы выйдем за предел в 31 байт.

В *AdvertiseSettings* мы указываем настройки вещания:

```
  AdvertiseSettings.Builder()

.setAdvertiseMode(AdvertiseSettings.ADVE
RTISE_MODE_LOW_LATENCY)

.setTxPowerLevel(AdvertiseSettings.ADVER
```

```
TISE_TX_POWER_HIGH)
          .setConnectable(false)
          .build()
```

В методе *setAdvertiseMode()* мы указываем, что данные нужно передавать с низкой латентностью, а в *setTxPowerLevel()* - передавать данные, используя максимальную мощность Bluetooth передатчика. Так как приложение сканирует и передает данные только когда приложение запущено и не находится в фоне, то данные настройки улучшат качество и скорость работы передачи данных и не повлияют на разряд батареи. В случае, если бы мы хотели сделать так, чтобы приложение продолжило слушать и вещать данные, в этом случае следовало бы указать менее энергозатратные настройки и использовать *Foreground Service*.

AdvertiseCallback, которые также указывается в методе *startAdvertising()* для запуска вещания, представляет собой набор функций обратного вызова следующего вида:

```
  private     val     advertisingCallback:
AdvertiseCallback      =      object      :
AdvertiseCallback() {
      override                              fun
onStartSuccess(settingsInEffect:
AdvertiseSettings) {

Log.d("BleProximityRepository",
"Advertising onStartSuccess")

super.onStartSuccess(settingsInEffect)
      }

      override                              fun
onStartFailure(errorCode: Int) {
          val errorMsg = when(errorCode)
{

ADVERTISE_FAILED_ALREADY_STARTED       ->
"ADVERTISE_FAILED_ALREADY_STARTED"

ADVERTISE_FAILED_FEATURE_UNSUPPORTED   ->
"ADVERTISE_FAILED_FEATURE_UNSUPPORTED"

ADVERTISE_FAILED_INTERNAL_ERROR        ->
"ADVERTISE_FAILED_INTERNAL_ERROR"

ADVERTISE_FAILED_TOO_MANY_ADVERTISERS  ->
"ADVERTISE_FAILED_TOO_MANY_ADVERTISERS"

ADVERTISE_FAILED_DATA_TOO_LARGE        ->
"ADVERTISE_FAILED_DATA_TOO_LARGE"
              else -> "Unknown error"
          }


Log.e("BleProximityRepository",
"Advertising onStartFailure: $errorMsg")

super.onStartFailure(errorCode)
      }
  }
```

Метод *onStartSucces()* срабатывает в случае, если вещание было успешно запущено, а *onStartFailure()* - в случае, если старт не произошел из-за ошибки (например, из-за превышения лимита в 31 байт на передачу данных - *ADVERTISE_FAILED_DATA_TOO_LARGE)*.

Так как мы выбрали модель вещания “Карусель”, мы должны отправлять данные в в цикле пакет за пакетом, указывая в каждом пакете идентификатор устройства и порядковый номер пакета. Таким образом, другое устройство сможет собрать из полученных пакетов контакт пользователя.

Упрощенно функция вещания выглядит так:

```
  @SuppressLint("MissingPermission")
  override           suspend           fun
startContactBroadcasting(contact:
Contact) =
  withContext(ioDispatcher) {
      //..
      val deviceId = getRandomDeviceId()

      while (isBroadcastingTurnedOn) {
          val    optimizedStrContact    =
optimizeStrContact(contact)   //   полное
сообщение в виде текста в формате
разработанного протокола
          val         strLength         =
optimizedStrContact.length
          val       totalPackages       =
ceil((strLength.toDouble()              /
BLE_PACKET_DATA_LENGTH)).toInt()

Log.d("BleProximityRepository",
"Prepared   data   for   broadcast.\nData:
$strContact\nOptimized             Data:
$optimizedStrContact\nPackages:
$totalPackages")
          val end = Char(totalPackages)
          for     (i     in     0     until
totalPackages) {
              val    start    =    (i    +
CHARSET_START).toChar()
              val   step   =   if   (i   ==
totalPackages - 1) {
                  val        rest        =
(totalPackages        -        1)        *
BLE_PACKET_DATA_LENGTH
                  if (strLength >= rest)
{
                      strLength - rest
                  } else {
                      rest - strLength
                  }
              } else {
                  BLE_PACKET_DATA_LENGTH
              }
              val        strData        =
"${deviceId}${start}${end}${optimizedStr
Contact.substring(i                     *
BLE_PACKET_DATA_LENGTH,        i        *
BLE_PACKET_DATA_LENGTH + step)}"
```

```
            val         data         =
AdvertiseData.Builder()

.setIncludeDeviceName(false)

.setIncludeTxPowerLevel(false)
                  .addServiceData(

uuidBleServiceHolder.getUuid(),

strData.toByteArray(Charsets.UTF_8)
                  )
                  .build()

bluetoothLeAdvertiser.startAdvertising(a
dvertiseSettings,                  data,
advertisingCallback)

Log.d("BleProximityRepository", "Started
broadcasting data part: $strData")

delay(BROADCAST_PERIOD_BETWEEN_PACKAGES)

bluetoothLeAdvertiser.stopAdvertising(ad
vertisingCallback)

Log.d("BleProximityRepository", "Stopped
broadcasting data part")
          }

delay(BROADCAST_PERIOD_BETWEEN_CIRCLES)
      }
  }
```

В данном случае мы разбиваем полное сообщение с контактом на части, и последовательно их отправляем в цикле. Для этого мы запускаем вещание одного пакета методом *startAdvertising()*, отправляем его в течение 200 миллисекунд *(delay(BROADCAST_PERIOD_BETWEEN_PACKAGES))* и останавливаем вещание методом stopAdvertising(). Далее в цикле переходим к вещанию следующего пакета.

B. Разработка программного модуля сканирования

Для сканирования advertise пакетов по технологии BLE в Android SDK используется класс *BluetoothLeScanner*, ссылку на который можно также получить из экземпляра класса *BluetoothAdapter*:

```
  val          bluetoothAdapter          =
bluetoothManager?.adapter
  val         bluetoothLeScanner         =
bluetoothAdapter?.bluetoothLeScanner
```

Перед запуском сканирования также следует проверить, доступен ли сам сканнер на устройстве:

```
  if   (bluetoothLeScanner   ==   null   ||
!bluetoothAdapter.isEnabled)  {
      Log.e("BleProximityRepository",
"bluetoothLeScanner is not available")
  }
```

Далее для запуска скакнирования у bluetoothLeScanner вызываем метод “*startScan*”:

```
  @SuppressLint("MissingPermission")
  override          suspend          fun
startContactsScan() {
      withContext(ioDispatcher) {
          val     bluetoothAdapter      =
bluetoothManager?.adapter
          val     bluetoothLeScanner     =
bluetoothAdapter?.bluetoothLeScanner
          if (bluetoothLeScanner == null
|| !bluetoothAdapter.isEnabled)  {

Log.e("BleProximityRepository",
"bluetoothLeScanner is not available")
              throw
BLENotAvailableException()
          }

          val           filter           =
ScanFilter.Builder()

.setServiceData(uuidBleServiceHolder.get
Uuid(), null)
              .build()

         val   bluetoothLeScanSettings   =
scanScanSettings.Builder()

.setScanMode(ScanSettings.SCAN_MODE_LOW_
LATENCY)
             .build()


bluetoothLeScanner.startScan(listOf(filt
er),            bluetoothLeScanSettings,
scanCallback)

Log.d("BleProximityRepository",
"startContactsScan")
      }
  }
```

Здесь разберем более подробно “*bluetoothLeScanner.startScan(listOf(filter), bluetoothLeScanSettings, scanCallback*)”. Для вызова метода требуется указать список фильтров для advertisement пакетов, в нашем случае мы будем фильтровать по UUID сервиса, чтобы не обрабатывать сообщения, которые не относятся к нашему протоколу. Следующим параметром является bluetoothLeScanSettings:

```
  val      bluetoothLeScanSettings      =
scanScanSettings.Builder()

.setScanMode(ScanSettings.SCAN_MODE_LOW_
LATENCY)
      .build()
```

Здесь мы указываем, что сканирование следует проводить часто с низким уровнем латентности.

Последним и самым главным параметром метода

является *scanCallback*, через него мы передаем функции обратного вызова для обработки прослушиваемых пакетов:

```
  private val protocolCache: MutableMap<String, MutableSet<BleContactPartModel>> = ConcurrentHashMap()

  private val scanCallback: ScanCallback = object : ScanCallback() {
      override fun onScanResult(callbackType: Int, result: ScanResult) {
          val bytes = result.scanRecord?.getServiceData(uuidBleServiceHolder.getUuid())
          bytes?.let {
              val data = String(bytes, Charsets.UTF_8)

Log.d("BleProximityRepository", "Found data: $data")
              try {
                  val deviceId = data.substring(0, 2)
                  var seq = protocolCache[deviceId]
                  if (seq == null) {
                      seq = sortedSetOf(bleContactPartComparator)

protocolCache[deviceId] = seq
                  }
                  seq.add(

BleContactPartModel(
                          deviceId = deviceId,
                          index = data[2].code - CHARSET_START,
                          count = data[3].code - CHARSET_START,
                          value = data.substring(4)
                      )
                  )
              } catch (e: Exception) {

Log.e("BleProximityRepository", "Incorrect BLE package: $data - skipping")
              }
          }


super.onScanResult(callbackType, result)
      }
  }
```

В *scanCallback* мы слушаем поступающие пакеты и распределяем их в коллекции типа *Map protocolCache* по полученным *deviceId* из нашего протокола. Далее сообщения сортируются.

Таким образом, во время сканирования мы заполняем *protocolCache*, в котором для каждого *deviceId* есть своя цепочка сообщений.

Один раз в три секунды мы проверяем состояние *protocolCache*, собираем из отдельных сообщений полную цепочку и формируем объект контакта пользователя:

```
  private const val SCAN_PERIOD = 3_000L

  override suspend fun getScannedContacts() =
      withContext(ioDispatcher) {
          delay(SCAN_PERIOD)
          val contacts = collectContacts()

Log.d("BleProximityRepository", "getScannedContacts: $contacts")
          contacts
      }

  private fun collectContacts(): List<Contact> {
      val result = mutableListOf<Contact>()
      for (set in protocolCache.values) {
          // маппинг данных и добавление сформированного Contact в коллекцию result
      }

      return result
  }”
```

VI. Заключение

В работе представлено спроектированное и реализованное мобильное приложение, которое позволяет обмениваться контактной информацией между близко расположенными мобильными устройствами. Приложение не требует открытия прямых соединений между устройствами и не использует никаких сторонних сервером или облачных хранилищ. Использованная архитектура сетевой пространственной близости вообще не требует поддержки от телекоммуникационных операторов, и представленное приложение будет работать даже на мобильных устройствах без SIM-карт. Код разработанного приложения выложен в открытый доступ и доступен как для непосредственного использования, так и для возможных расширений. Отметим, например, что помимо представленного в работе C2C приложения, на тех же идеях может быть построен открытый API для обмена данными в B2C и B2B приложениях.

Благодарности

Работа выполнялась в рамках курса по Интернету Вещей [8]. Хотелось бы здесь отметить тех, кто способствовал появлению курса и программы. Это, во-первых, проф. М.А. Шнепс-Шнеппе. Все началось с одной из первых

работ на русском языке, посвященной инфраструктуре среды обитания (будущий Умный Город) [11], и продолжилось работами по стандартизации M2M [9, 10]. Большое влияние на курс оказали также работы по транспортной тематике, инициированные одним из самых активных авторов журнала INJOIT В.П. Куприяновским [12,13,14].

Библиография

[1] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On proximity-based information delivery." International Conference on Distributed Computer and Communication Networks. Springer, Cham, 2018.

[2] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On Content Models for Proximity Services." 2019 24th Conference of Open Innovations Association (FRUCT). IEEE, 2019.

[3] Namiot, Dmitry, Manfred Sneps-Sneppe, and Romass Pauliks. "On mobile applications based on proximity." 2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW). IEEE, 2019.

[4] D. Namiot, "Context-Aware Browsing -- A Practical Approach," 2012 Sixth International Conference on Next Generation Mobile Applications, Services and Technologies, 2012, pp. 18-23, doi: 10.1109/NGMAST.2012.13.

[5] Michael, Katina, and Roba Abbas. "Behind COVID-19 contact trace apps: The Google–Apple partnership." IEEE Consumer electronics magazine 9.5 (2020): 71-76.

[6] Volosnikova, Polina, and Dmitry Namiot. "On using web interfaces for network proximity services." International Journal of Open Information Technologies 8.6 (2020): 83-90.

[7] Namiot, Dmitry, and Ivan Makarychev. "On the alternative model of location marking on social networks." International Journal of Open Information Technologies 8.2 (2020): 74-90.

[8] Namiot, Dmitry, Manfred Sneps-Sneppe, and Yousef Ibrahim Daradkeh. "On internet of things education." 2017 20th conference of open innovations association (FRUCT). IEEE, 2017.

[9] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On m2m software." International Journal of Open Information Technologies 2.6 (2014): 29-36.

[10] Sneps-Sneppe, Manfred, and Dmitry Namiot. "About M2M standards and their possible extensions." 2012 2nd Baltic Congress on Future Internet Communications. IEEE, 2012.

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

[12] Куприяновская Ю. В. и др. Умный контейнер, умный порт, BIM, Интернет Вещей и блокчейн в цифровой системе мировой торговли //International Journal of Open Information Technologies. – 2018. – Т. 6. – №. 3. – С. 49-94.

[13] Соколов И. А. и др. Прорывные инновационные технологии для инфраструктур. Евразийская цифровая железная дорога как основа логистического коридора нового Шелкового пути //International Journal of Open Information Technologies. – 2017. – Т. 5. – №. 9. – С. 102-118.

[14] Цифровая трансформация экономики, железных дорог и умных городов. Планы и опыт Великобритании / В. П. Куприяновский, Г. В. Суконников, С. А. Синягов [и др.] // International Journal of Open Information Technologies. – 2016. – Т. 4. – № 10. – С. 22-31. – EDN WXBAYJ.

[15] Quack code https://github.com/AgnaWiese/Quack

[16] Quack application https://play.google.com/store/apps/details?id=ru.trushkina.quack

[17] Camps-Mur, Daniel, Andres Garcia-Saavedra, and Pablo Serrano. "Device-to-device communications with Wi-Fi Direct: overview and experimentation." IEEE wireless communications 20.3 (2013): 96-104.

[18] Khan, Muhammad Asif, et al. "Wi-Fi Direct Research-Current Status and Future Perspectives." Journal of Network and Computer Applications 93 (2017): 245-258.

[19] Kevin Townsend, Robert Davidson, Akiba, and Carles Cufi. Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking. O'Reilly, Sebastopol, CA, 2014

[20] Bluetooth Smart's Rise From Obscurity to Mainstream — EE Times. http:www.eetimes.com/author.asp?section_id=36&doc_id=1321690

[21] Minch R. P. Privacy issues in location-aware mobile devices // Proceedings of the 37th Annual Hawaii International Conference on System Sciences, 2004. - IEEE, 2004.

[22] Haski J. et al. Location prediction using wireless signals on online social networks : заяв. пат. 15838289 США. - 2018.

[23] Гриняк В. М., Гриняк Т. М., Цыбанов П. А. Позиционирование внутри помещений с помощью Bluetooth-устройств //Территория новых возможностей. Вестник Владивостокского государственного университета экономики и сервиса. - 2018. - №. 2 (41). - С. 137-147.

[24] Кузьмин М. Ф. Проектирование и разработка мобильного приложения для навигации на парковках с применением Bluetooth-маячков //Г орный информационно-аналитический бюллетень (научно-технический журнал). - 2017. - №. 3. - С. 383-393.

[25] Deordica B., Alexandru M. Advertisement using Bluetooth Low Energy //Review of the Air Force Academy. - 2014. - №. 2. - С. 6570

[26] Namiot, Dmitry, and Manfred Sneps-Sneppe. "The physical web in smart cities." 2015 Advances in Wireless and Optical Communications (RTUWO). IEEE, 2015..

[27] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On mobile Bluetooth tags." arXiv preprint arXiv:1502.05321 (2015).

[28] Namiot, Dmitry, and Manfred Sneps-Sneppe. "Context-aware data discovery." 2012 16th International Conference on Intelligence in Next Generation Networks. IEEE, 2012.

[29] Sneps-Sneppe, Manfred, and Dmitry Namiot. "On physical web models." 2016 International Siberian Conference on Control and Communications (SIBCON). IEEE, 2016.

[30] Namiot, Dmitry, and Ivan Makarychev. "On the alternative model of location marking on social networks." International Journal of Open Information Technologies 8.2 (2020): 74-90.

[31] Moskala, Marcin, and Igor Wojda. Android Development with Kotlin. Packt Publishing Ltd, 2017.

[32] Anderson, Chris. "The model-view-viewmodel (mvvm) design pattern." Pro Business Applications with Silverlight 5. Apress, Berkeley, CA, 2012. 461-499.

Data exchange between mobile devices without organizing the connections

Evgenia Trushkina, Dmitry Namiot

Abstract— **The article discusses one approach to organizing data exchange between closely spaced mobile devices, which does not require a direct connection between devices and does not use third-party (cloud) storage. The article was written based on the results of the final qualifying work performed at the faculty of the CMC of Lomonosov Moscow State University. The paper proposes a mobile application for the Android platform, based on the network spatial proximity model, which allows you to exchange contact information with other mobile devices located nearby. Within the framework of this model, a limited area of propagation of wireless network signals is used to determine spatial proximity. Due to the complete exclusion of working with geo-coordinates, such networks may not necessarily be stationary nodes with a known location. The main usage model is the software creation of such wireless networks, specifically for the provision of services that use location information. At the same time, advertising (representative) information of such nodes can be used to transfer user data.**

Keywords - **network spatial proximity, Bluetooth.**

REFERENCES

[1] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On proximity-based information delivery." International Conference on Distributed Computer and Communication Networks. Springer, Cham, 2018.

[2] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On Content Models for Proximity Services." 2019 24th Conference of Open Innovations Association (FRUCT). IEEE, 2019.

[3] Namiot, Dmitry, Manfred Sneps-Sneppe, and Romass Pauliks. "On mobile applications based on proximity." 2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW). IEEE, 2019.

[4] D. Namiot, "Context-Aware Browsing -- A Practical Approach," 2012 Sixth International Conference on Next Generation Mobile Applications, Services and Technologies, 2012, pp. 18-23, doi: 10.1109/NGMAST.2012.13.

[5] Michael, Katina, and Roba Abbas. "Behind COVID-19 contact trace apps: The Google–Apple partnership." IEEE Consumer electronics magazine 9.5 (2020): 71-76.

[6] Volosnikova, Polina, and Dmitry Namiot. "On using web interfaces for network proximity services." International Journal of Open Information Technologies 8.6 (2020): 83-90.

[7] Namiot, Dmitry, and Ivan Makarychev. "On the alternative model of location marking on social networks." International Journal of Open Information Technologies 8.2 (2020): 74-90.

[8] Namiot, Dmitry, Manfred Sneps-Sneppe, and Yousef Ibrahim Daradkeh. "On internet of things education." 2017 20th conference of open innovations association (FRUCT). IEEE, 2017.

[9] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On m2m software." International Journal of Open Information Technologies 2.6 (2014): 29-36.

[10] Sneps-Sneppe, Manfred, and Dmitry Namiot. "About M2M standards and their possible extensions." 2012 2nd Baltic Congress on Future Internet Communications. IEEE, 2012.

[11] Volkov, A. A. O zadachah sozdanija jeffektivnoj infrastruktury sredy obitanija / A. A. Volkov, D. E. Namiot, M. A. Shneps-Shneppe // International Journal of Open Information Technologies. – 2013. – T. 1. – # 7. – S. 1-10.

[12] Kuprijanovskaja Ju. V. i dr. Umnyj kontejner, umnyj port, BIM, Internet Veshhej i blokchejn v cifrovoj sisteme mirovoj torgovli //International Journal of Open Information Technologies. – 2018. – T. 6. – #. 3. – S. 49-94.

[13] Sokolov I. A. i dr. Proryvnye innovacionnye tehnologii dlja infrastruktur. Evrazijskaja cifrovaja zheleznaja doroga kak osnova logisticheskogo koridora novogo Shelkovogo puti //International Journal of Open Information Technologies. – 2017. – T. 5. – #. 9. – S. 102-118.

[14] Cifrovaja transformacija jekonomiki, zheleznyh dorog i umnyh gorodov. Plany i opyt Velikobritanii / V. P. Kuprijanovskij, G. V. Sukonnikov, S. A. Sinjagov [i dr.] // International Journal of Open Information Technologies. – 2016. – T. 4. – # 10. – S. 22-31. – EDN WXBAYJ.

[15] Quack code https://github.com/AgnaWiese/Quack

[16] Quack application https://play.google.com/store/apps/details?id=ru.trushkina.quack

[17] Camps-Mur, Daniel, Andres Garcia-Saavedra, and Pablo Serrano. "Device-to-device communications with Wi-Fi Direct: overview and experimentation." IEEE wireless communications 20.3 (2013): 96-104.

[18] Khan, Muhammad Asif, et al. "Wi-Fi Direct Research-Current Status and Future Perspectives." Journal of Network and Computer Applications 93 (2017): 245-258.

[19] Kevin Townsend, Robert Davidson, Akiba, and Carles Cufi. Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking. O'Reilly, Sebastopol, CA, 2014

[20] Bluetooth Smart's Rise From Obscurity to Mainstream — EE Times. http:www.eetimes.com/author.asp?section_id=36&doc_id=1321690

[21] Minch R. P. Privacy issues in location-aware mobile devices // Proceedings of the 37th Annual Hawaii International Conference on System Sciences, 2004. - IEEE, 2004.
[22] Haski J. et al. Location prediction using wireless signals on online social networks : zajav. pat. 15838289 SShA. - 2018.
[23] Grinjak V. M., Grinjak T. M., Cybanov P. A. Pozicionirovanie vnutri pomeshhenij s pomoshh'ju Bluetooth-ustrojstv //Territorija novyh vozmozhnostej. Vestnik Vladivostokskogo gosudarstvennogo universiteta jekonomiki i servisa. - 2018. - #. 2 (41). - S. 137-147.
[24] Kuz'min M. F. Proektirovanie i razrabotka mobil'nogo prilozhenija dlja navigacii na parkovkah s primeneniem Bluetooth-majachkov //G ornyj informacionno-analiticheskij bjulleten' (nauchno-tehnicheskij zhurnal). - 2017. - #. 3. - S. 383-393.
[25] Deordica B., Alexandru M. Advertisement using Bluetooth Low Energy //Review of the Air Force Academy. - 2014. - #. 2. - S. 6570
[26] Namiot, Dmitry, and Manfred Sneps-Sneppe. "The physical web in smart cities." 2015 Advances in Wireless and Optical Communications (RTUWO). IEEE, 2015..
[27] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On mobile Bluetooth tags." arXiv preprint arXiv:1502.05321 (2015).
[28] Namiot, Dmitry, and Manfred Sneps-Sneppe. "Context-aware data discovery." 2012 16th International Conference on Intelligence in Next Generation Networks. IEEE, 2012.
[29] Sneps-Sneppe, Manfred, and Dmitry Namiot. "On physical web models." 2016 International Siberian Conference on Control and Communications (SIBCON). IEEE, 2016.
[30] Namiot, Dmitry, and Ivan Makarychev. "On the alternative model of location marking on social networks." International Journal of Open Information Technologies 8.2 (2020): 74-90.
[31] Moskala, Marcin, and Igor Wojda. Android Development with Kotlin. Packt Publishing Ltd, 2017.
[32] Anderson, Chris. "The model-view-viewmodel (mvvm) design pattern." Pro Business Applications with Silverlight 5. Apress, Berkeley, CA, 2012. 461-499.