Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра автоматизации систем вычислительных комплексов

ФЕДЧЕНКО Игорь Олегович

Информационные системы на базе push-уведомлений

ДИПЛОМНАЯ РАБОТА

Научный руководитель:
к.ф-м.н., ст. н. с.
Д.Е.Намиот

Москва, 2015

Аннотация

В данной работе речь пойдёт об исследовании существующих информационных систем, основанных на push-уведомлениях, и о реализации новой информационной системы, которая исправляет ряд недостатков, обнаруженных у аналогов – в частности, не требует от пользователя навыков программирования и автоматизирует создание уникального мобильного приложения.

В результате была разработана и предоставлена в широкий доступ информационная система, удовлетворяющая требованиям исходной задачи.

Содержание

1. Введение

3.3. Происхождение push-уведомлений

Уже давно в мире существует задача донесения информации от продавца потенциальному покупателю. Помимо массовой рекламы, в последнее время популярность приобрела персонализированная передача информации. В эпоху интернета мы продаём товары и делаем покупки с помощью компьютеров. Технические достижения, направленные на решение этого вопроса, обусловлены во многом нуждами коммерции и торговли. Ничего сверхъестественного в этом нет – даже в древних цивилизациях торговля служила двигателем науки и прогресса, и по сей день она остаётся одной из первопричин создания новых, более высокотехнологичных и научных решений, в том числе среди задач программирования, обработки и анализа данных и взаимодействия с персональными устройствами пользователей (потенциальных покупателей).

“В России с 1865 года (после появления Русского телефонного агентства) оперативное распространение новостей приобрело общегосударственный масштаб. В начале XX века появляется новый вид рекламы – эфирная. В 1920 году появляется новое мощное средство рекламирования – радиовещание. В конце XX века стала активно использоваться телевизионная реклама” [10]

Вскоре после этого человечество начало массово осваивать такую технологию, как интернет – и тут тоже не обошлось без рекламных нововведений: первые баннеры, всплывающие окна и автоматический переход на внешние ссылки. Однако с развитием аппаратуры и техники появились первые мобильные телефоны – и вскоре интернет-реклама стала не так продуктивна, как реклама на персональном устройстве, которое человек носит с собой, и здесь появляется необходимость в серьезных научные открытиях в области обработки и персонализации данных с целью привлечения потенциального покупателя. Регулярно придумывается огромное число новых рекламных подходов на стыке информационных технологий, программирования и последних маркетинговых исследований и открытий.

Несколько лет назад одним из самых активно используемых решений стала рассылка смс-сообщений. Однако совершенно очевиден тренд, в котором мобильные

абоненты избегают просмотра подобных сообщений, считая это спамом. Таким образом, появилась необходимость в новом инструменте оповещения пользователей, и одним из них стали push-уведомления.

1.2. Что такое push-уведомления

Push-уведомления – это краткие всплывающие уведомления, которые появляются на экране мобильного телефона и сообщают о важных событиях и обновлениях. [1] Они поддерживаются практически в каждой мобильной операционной системе (ОС). Сами по себе они являются средством представления информации пользователю и могут являться интерактивным элементом, позволяющим просмотреть информацию подробнее после осуществления клика по элементу. Они не зависят от наличия или отсутствия интернета, и могут отображаться, например, в зависимости от положения телефона в пространстве (например, используя GPS), или просто уведомлять пользователя телефона о текущем времени суток. Однако мы в данной работе рассматриваем push-уведомления в связке с интернет сервисом, обеспечивающим ведение интернет-рассылки.

Доставка сообщений в системах Push-уведомлений осуществляется по сети интернет, используя клиент-серверную модель. Мобильный клиент (subscriber) при помощи установленного приложения подписывается на рассылку push-уведомлений, а сервер публикаций (publisher), по наступлению определенного события, осуществляет рассылку подписавшимся клиентам. Таким образом, адресатом сообщения всегда является некоторое приложение. Важно отметить, что push–уведомления позволяют серверам рассылки уведомлять пользователей о наступлениях событий, даже если приложение в данный конкретный момент неактивно. Достаточно, чтобы приложение было установлено, и действовала подписка.

Отправка на сервере, как правило, осуществляется при наступлении некоторого события. Само событие отслеживается некоторым отдельным приложением, и может быть любым: получение сообщения от администратора, или нахождение некоторого количества клиентских устройств поблизости (может использоваться пассивный Wi-Fi или Bluetooth мониторинг).

Системы, позволяющие некоторым образом обрабатывать информацию, в совокупности с системами доставки этой информации и связанные с этим аппаратные, программные и человеческие ресурсы, называются *информационными системами [8]*.

1.3. Системы на основе push-уведомлений

Изначально системы создавались автономными: инициатором события мог быть либо администратор системы, либо выполнение предопределённого условия. Однако следующим этапом в развитии концепции рассылок на базе push-уведомлений явилось появление сервисов, предоставляющих интерфейс программирования приложений (application programming interface, API) для настройки собственных рассылок. Сервисы предоставляют своим пользователям возможность настройки контента рассылки, а так же загрузки клиентских приложений, которые будут являться подписчиками рассылки, реализованных для различных мобильных платформ (iOS, Android и другие).

Эти сервисы являются довольно популярными (перечень некоторых из них будет приведён далее в работе), однако у них есть один недостаток с точки зрения большинства коммерческих предприятий – это необходимость обладания навыками программирования. Когда у коммерческой организации или, тем более, у некоммерческого предприятия, возникает потребность в организации рассылки (в том числе это может быть рассылка для узкого круга лиц – например, для сотрудников компании), то необходимость нанимать программиста, имеющего навыки создания мобильных приложений, к тому же для нескольких мобильных операционных систем – это если не проблема, то по крайней мере издержки, которых хотелось бы избежать.

Поэтому возникает потребность в следующей ступени развития сервисов – это сервисы, которые, помимо представления API для создания и ведения рассылки, предоставляют мобильное приложение, работающее с данным API и устанавливаемое на мобильное устройство клиента-подписчика, а так же панель управления рассылкой push-уведомлений. Такие сервисы являются уже готовым бизнес-решением для организаций, упомянутых выше: при необходимости вести рассылку определенному кругу лиц, нужно просто зарегистрироваться, далее при необходимости произвести оплату, скачать клиентское приложение и настроить рассылку в личном кабинете панели управления. Такие сервисы являются уже можно назвать реализацией информационной системы. Более подробно перечень основных сервисов этого рода и взаимодействие с ними также рассматривается в следующих разделах.

Однако данная работа посвящена следующему шагу, аналогов которому пока что не существует, хотя ведутся разработки в этом направлении [2]. Идея состоит в том, чтобы

дать пользователям онлайн-сервиса возможность не просто создать рассылку, но и собрать собственное мобильное приложение, являющееся подписчиком. Настроить его дизайн, в некоторой степени определить дополнительный функционал, и скачать с последующей возможностью выложить в качестве фирменного приложения в сеть интернет. В рамках работы рассматривается создание такого приложения для операционной системы Android, поскольку она является наиболее широко используемой [3].

Таким образом, задача, которую мы поставили перед собой в рамках дипломной работы, является очень актуальной. Те, кто хотел бы иметь собственное, отличное от других приложение, позволяющее организовать рассылки на базе push-уведомлений, теперь смогут это сделать с минимальными усилиями и затратами как сил, так и ресурсов. Так же актуальность показывает количество инвестиций, которые привлёк аналогичный проект [2], разработка которого началась приблизительно в одно время с нашей работой.

Новизна данной работы состоит в том, чтобы создать сервис, не имеющий аналогов. Подобная идея на момент нашей начала работы над этим проектом не была никем реализована в полной мере, не смотря на явную актуальность проблемы и даже коммерческую выгоду. Во всяком случае, нам не удалось найти ни одного сервиса, который бы реализовывал описанную систему.

2. Постановка задачи

Задача данной дипломной работы заключается в построении информационной системы на основе push-уведомлений, которая позволяла бы без применения навыков программирования осуществлять пользователям тематическую рассылку текстовой и графической информации.

3. Обзор существующих решений

Существует довольно большое количество информационных систем, которые так или иначе пытаются предоставить инструмент одним пользователям производить рассылку информации другим пользователям посредством push-уведомлений. В целом структура и функционал таких систем сильно пересекаются, поэтому в данном разделе мы опишем, как эти системы устроены, какие именно возможности могут быть предоставлены пользователям, а также проведём сравнение некоторых популярных сервисов.

3.1. Основа существующих решений

Для доставки push-уведомлений от сервера-отправителя к клиенту-получателю в целях экономии ресурсов клиентских устройств все существующие системы использует промежуточное звено. Этим звеном является служба уведомлений (Push Service Provider). Вот несколько примеров основных служб:

1) GCM (Google Cloud Messaging) для мобильных устройств под ОС Android;
2) APNS (Apple Push Notification Service) для мобильных устройств под iOS;
3) MPNS (Microsoft Push Notification Service) для мобильных устройств под ОС Windows.

Эти службы поддерживают очень экономичное соединение с мобильными устройствами и осуществляют доставку push–уведомлений. ОС мобильного устройства перенаправляет уведомление соответствующему приложению. В приложениях под Windows Phone уведомления появляются в верхней части экрана. В приложениях под Android и iOS уведомления появляются в верхней части экрана в панели уведомлений.

Однако эти сервисы предоставляют именно API, под которое можно запрограммировать приложение, с ним взаимодействующее. Также необходимо самостоятельно создать сервис по управлению рассылкой. Как мы уже говорили, это не всегда удобно, что и обусловило появление информационных систем, рассматриваемых далее.

3.2. Перечень доступных возможностей

В первую очередь, во всех рассматриваемых решениях пользователю предоставляется возможность осуществлять 3 вида рассылок:

a. *Широковещательная рассылка*

   После указания контента рассылки, сервер запрашивает список зарегистрированных клиентов, после чего им осуществляется рассылка

b. *Персонализированная (индивидуальная) рассылка*:

   При указании контента рассылки публикатор также указывает уникальный номер получателя (из списка имеющихся у зарегистрированных пользователей), и по этому номеру осуществляется доставка

c. *Сегментированная рассылка*

   Нечто среднее между первыми двумя видами: публикатор должен указать условие на данные получателя. Сервер выбирает тех подписчиков, данные которых удовлетворяют условию, после чего им осуществляется доставка.

Информационные системы в этой области ставят перед собой следующие основные задачи:

- Предоставить пользователю инструмент для создания тематической рассылки
- Предоставить возможность осуществлять любой из 3ёх видов рассылок (см. выше), а для сегментированной рассылки предоставить как можно более гибкий способ выборки подписчиков для отправки
- Предоставить как можно более прозрачный способ подписи клиентского устройства на рассылку. В частности, нас будет интересовать возможность поместить в мобильное устройство пользователей приложение-подписчик на рассылку

Для решения задач из этого списка, пользователям предоставляются следующие инструменты:

1) Каждому пользователю-публикатору информационной системы выделяется виртуальный личный кабинет, в котором пользователь имеет консоль управления рассылками. Наиболее типичные возможности консоли: создание рассылки по

определённой тематике, отправка сообщения (одним из трёх видов push-рассылки), просмотр статистики открытых push-уведомлений и т.п.

2) Для осуществления различных видов рассылок используются метки, которые присваиваются каждому отдельному получателю. Это могут быть как произвольные тэги, так и универсальные характеристики, такие как версия операционной системы мобильного. Для персонализированной рассылки могут использоваться уникальные идентификаторы, генерируемые самим мобильным приложением, или же персональные параметры, настраиваемые пользователем – к примеру, день рождения владельца телефона.

   Как правило, также есть возможность указать заголовок сообщения и ссылку для перехода при нажатии получателем на сообщение, когда оно появляется в области уведомлений. Может указываться и изображение, появляющееся вместе с сообщением, и даже звуковой сигнал.

3) Для того, чтобы получатель вообще мог каким-то образом подписаться на рассылку, необходимо во-первых создать приложение, которое бы являлось подписчиком в техническом смысле, и во-вторых доставить это приложение конечному потребителю. Для этого применяются самые разнообразные решения (одно из которых предлагается в данной работе и является уникальным). В целом, возможные подходы можно разбить на следующие категории:

   - Предоставить программный код, который нужно внедрить в собственное мобильное приложение. Такую возможность предоставляют практически все сервисы, и это является наиболее гибким способом доставки, однако в то же время наиболее трудоёмким.
   - Предоставить готовое к загрузке «тестовое» приложение. В противовес предыдущему подходу, этот способ практически не требует усилий со стороны пользователя-публикатора информационной системы, однако также подход лишён гибкости в настройке приложения
   - Предоставить более или менее настраиваемый в личном кабинете шаблон мобильного приложения, который пользователь-публикатор может загрузить себе на компьютер. После этого, доработав при необходимости, приложение может быть собрано в подходящей для шаблона среде разработки и выложено как своё собственное

- Основной способ донести так или иначе созданное готовое мобильное приложение до пользователя – это выложить его в основной, специализированный для каждой мобильной операционной системы сборник приложений. Такие сборники всегда находятся в публичном доступе и дают разработчикам возможность загружать туда свои приложения. Также публикатор, желающий распространить своё приложение, может отправить его по электронной почте или выложить где-то в публичном доступе ссылку для загрузки приложения.

Встречаются и другие способы решения перечисленных задач, однако нет смысла перечислять их в общем обзоре. Более подробно мы их опишем в сравнительной таблице наиболее популярных информационных систем на основе push-уведомлений.

3.3. Сравнительная характеристика существующих решений

Здесь мы представим таблицу сравнения некоторых известных информационных систем, которые используются для проведения информационных рассылок на основе push-уведомлений. Выборка сервисов для сравнения – субъективный выбор автора, однако она достаточно репрезентативна, чтобы показать, какие задачи существующие системы решают, а какие нет.

Название сервиса	*Поддерживаемые устройства*	*Консоль управления рассылкой*	*Поддержка видов рассылок*	*Возможности настройки содержимого уведомлений*	*Средства создания мобильного приложения и доставки его пользователям-подписчикам*
Amazon Simple Notification *Service*	iOS, Android, Fire OS, Windows и другие	Есть, стандартная	Все виды	Можно указать тему сообщения, текст, звук, ссылку для перехода и изображение	Готовое приложение и программный код для встраивания в собственное приложение
Windows Azure	Windows 8, iOS,	Есть, стандарт	Все виды	Аналогично	Программный код для встраивания и

Notification Hubs	Windows Phone и Android	ная			тестовое приложение
Urban Airship Push-notifications	Аналогичный набор	Стандартная	Все виды	Аналогично	Программный код, встраиваемый в приложение
Для примера перечислим ещё несколько систем, не углубляясь в их функционал, поскольку он в большой мере аналогичен перечисленным ранее: • Сервис PUSH-уведомлений Parse • *Сервис PUSH-уведомлений Uniqush* – сервис с открытым исходным кодом • *Сервис PUSH-уведомлений PushOver* – имеются мобильное и настольное приложение, ориентированные на распространение среди пользователей подписчиков					
Jeapie	Аналогично	Стандартная	Все виды	Отличительной особенностью является визуальный конструктор уведомлений, который позволяет акцентировать внимание публикатора на визуальной составляющей уведомления, что является очень важным	Готовый исходный код для встраивания в приложения. Обещается шаблон приложения.

Push2Press	iOS, Android	Как отдельная CMS, так и плагин для Word Press.	Все виды	Стандартные	В отличие от большинства остальных, используя Titanium SDK, сервис позволяет сконструировать приложение практически полностью онлайн, после чего скачать готовый к сборке шаблон, и собрать его в настольной среде разработки
OpenPush	Всё как у большинства, но для распространения предоставляется готовый шаблон приложения. Мобильные ОС: iPhone, Android, Blackberry, Windows Phone и другие				
Nudge	Аналогично предыдущему				

Таким образом, можно подвести итоги. Для начала отметим те задачи, которые перечисленные сервисы позволяют решить:

1. Для создания рассылки на основе push-уведомлений необходимо создать одну-две учётные записи, после чего контент рассылки можно задать в консоли управления
2. Есть готовые решения для интеграции сервиса с мобильным приложением: от библиотеки, встраиваемой в исходный код приложения, до готового приложения, настроенного на взаимодействие с конкретным сервисом.
3. Сопровождение рассылки: будь то платный веб-сервис или сервис с открытым исходным кодом, всё хранение сообщений, гарантию их доставки и защиту информации он берёт на себя.
4. В целом, не нужно изобретать велосипед: уже имеется высокоуровневая надстройка рассылки, которую лишь нужно настроить пользователю.

Теперь отметим проблему, которая осталась нерешённой:

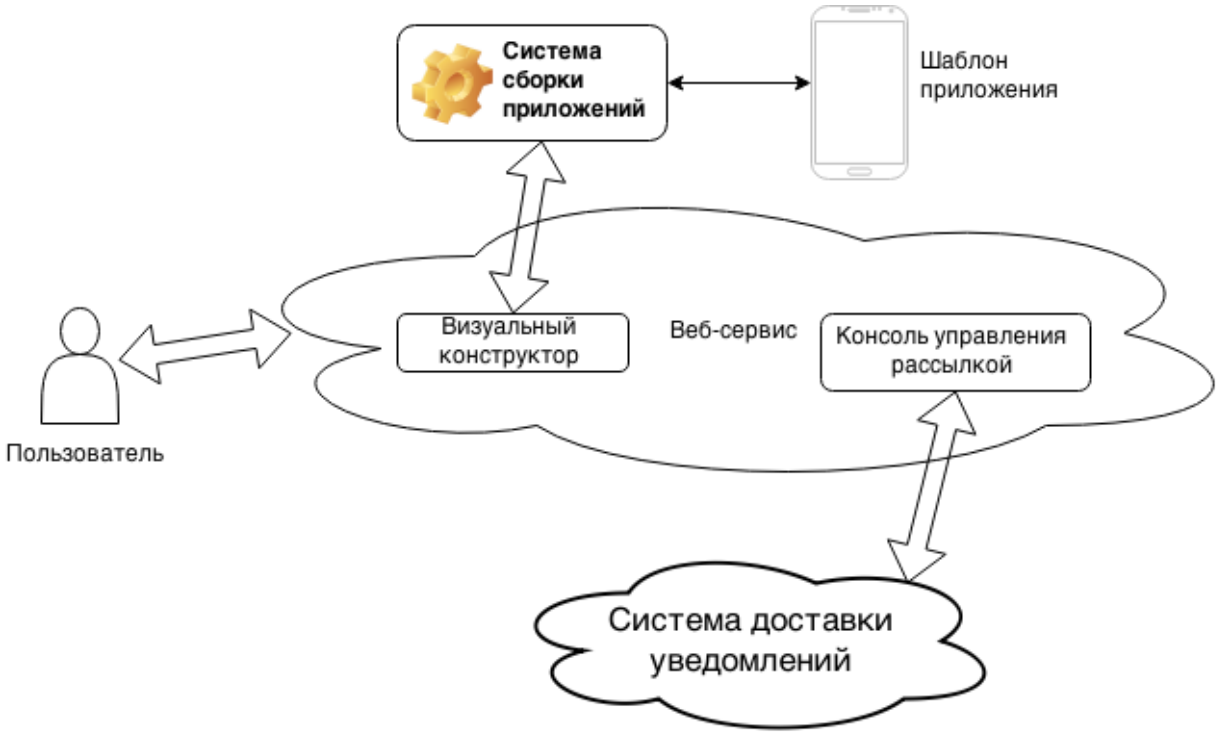
1. Со стороны клиента приложение может быть либо стандартным (готовое решение от сервиса), либо написанным пользователем сервиса самостоятельно с использованием библиотек сервиса или шаблонов. А если пользователь хочет уникальное приложение, однако он далёк от программирования? Если он не хочет или не имеет возможности устанавливать программное обеспечение для сборки мобильного приложения? Придётся искать специалиста.

Эту проблему мы и хотим решить в рамках дипломной работы.

4. Описание решения задачи

Вообще говоря, информационную систему, которая позволила бы решить поставленную задачу, можно разделить на три составляющие:

1) Веб-сервис, предоставляющий пользователю-публикатору взаимодействовать с системой. Это личный кабинет, включающий в себя как визуальный конструктор мобильного приложения, так и консоль управления рассылкой
2) Шаблон мобильного приложения, настраиваемый в визуальном конструкторе, а также система сборки приложения из этого шаблона
3) Система доставки уведомлений. Нужна система, которая позволяла бы осуществлять отправку уведомлений на конкретные собранные приложения.



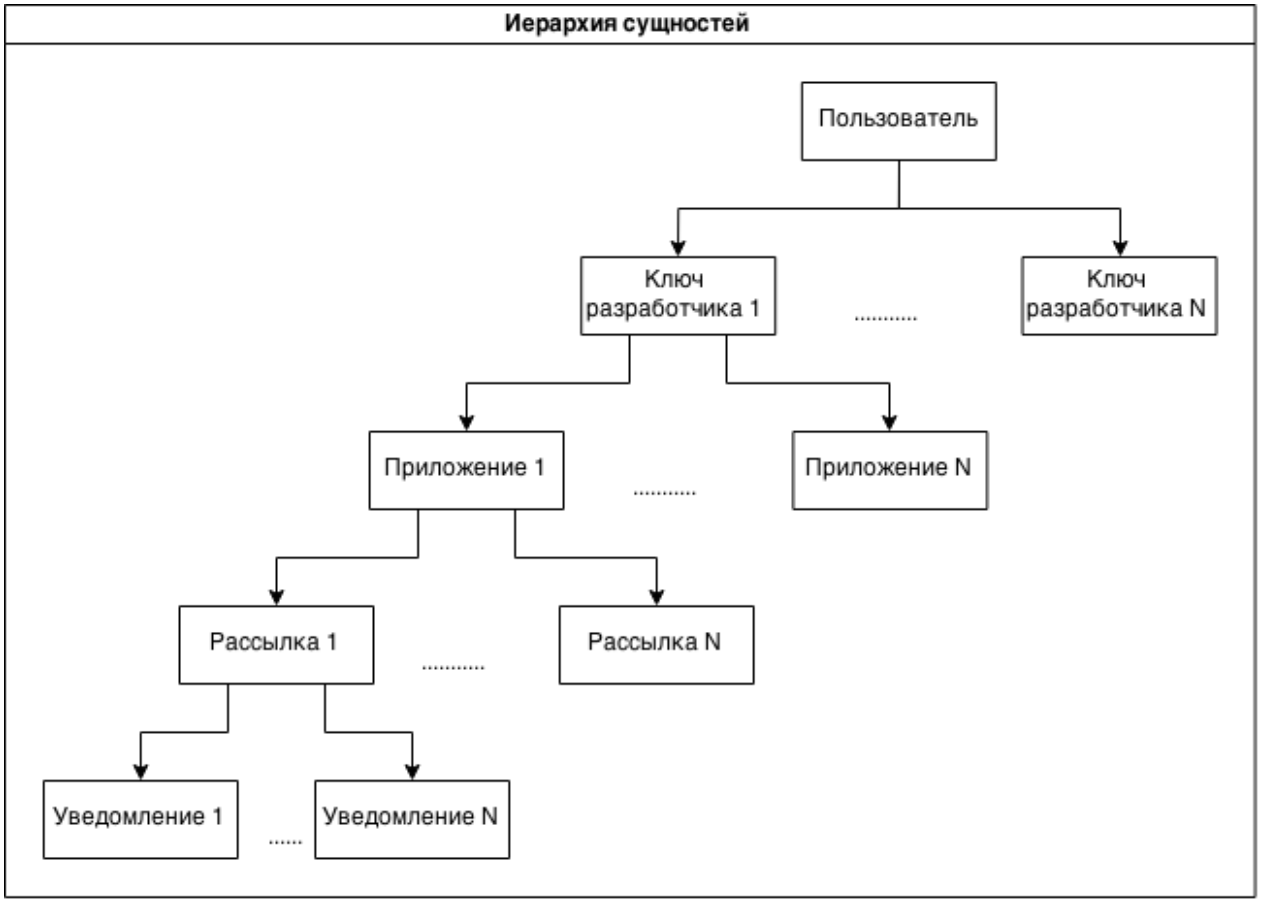
Далее мы опишем модель каждого из этих компонентов.

4.1. Описание веб-сервиса информационной системы

Веб-сервис нашей информационной системы будет построен следующим образом:

- Новый пользователь, желающий стать публикатором рассылки, должен зарегистрироваться, чтобы получить собственный аккаунт и войти в личный кабинет

- Отношения между мобильными приложениями, создаваемыми пользователем, и рассылками, которые он будет публиковать, характеризуются связями один-ко-многим, то есть сначала пользователь создаёт сущность приложения, которое будет распространяться среди пользователей, а затем для этого приложения он создаёт рассылки, на которые это приложение будет подписано.
- В каждой рассылке может отправляться, вообще говоря, не ограниченное число уведомлений.
- Для того, чтобы созданное приложение можно было публиковать в свободном доступе, оно должно быть подписано уникальным ключом разработчика. Ключи также могут создаваться пользователем в личном кабинете и привязываться к приложениям связью один ключ ко многим приложениям.
- Таким образом, связка ключ_разработчика -> приложение -> рассылки -> сообщения_в_рассылках образуют древовидную структуру:



Стандартный алгоритм действий рядового пользователя-публикатора после регистрации можно разделить на две фазы:

- Создание мобильного приложение для распространения между подписчиками
- Создание и управление рассылкой

Описание действий пользователя для создания приложения можно описать следующим образом:

1) Пользователь в визуальном конструкторе создаёт сущность мобильного приложения. Целью данной работы не является демонстрация возможностей по визуальному построению мобильных приложений, так что для обозначения уникальности приложения достаточно указать логотип приложения (подробнее мы это рассмотрим в подразделе, посвящённом шаблону мобильного приложения).
2) Как во время создания приложения, так и после этого, перейдя на страницу управления этим приложением, пользователь может привязать к приложению один из ключей разработчика (или перейти к мастеру создания ключей для создания такового).
3) Приложение может быть загружено как с привязанным ключом, так и без него. Без привязки ключа оно может быть загружено с целью отладки приложения – тогда оно будет подписано специальным, отладочным ключом, запрещающим публикацию. Если приложение скачано после привязки к нему ключа разработчика, оно готово к публикации, например, в Google Play Market.

Более строго процесс создания приложения пользователем описывается следующей блок-схемой «Алгоритм создания приложения» (см. ниже).

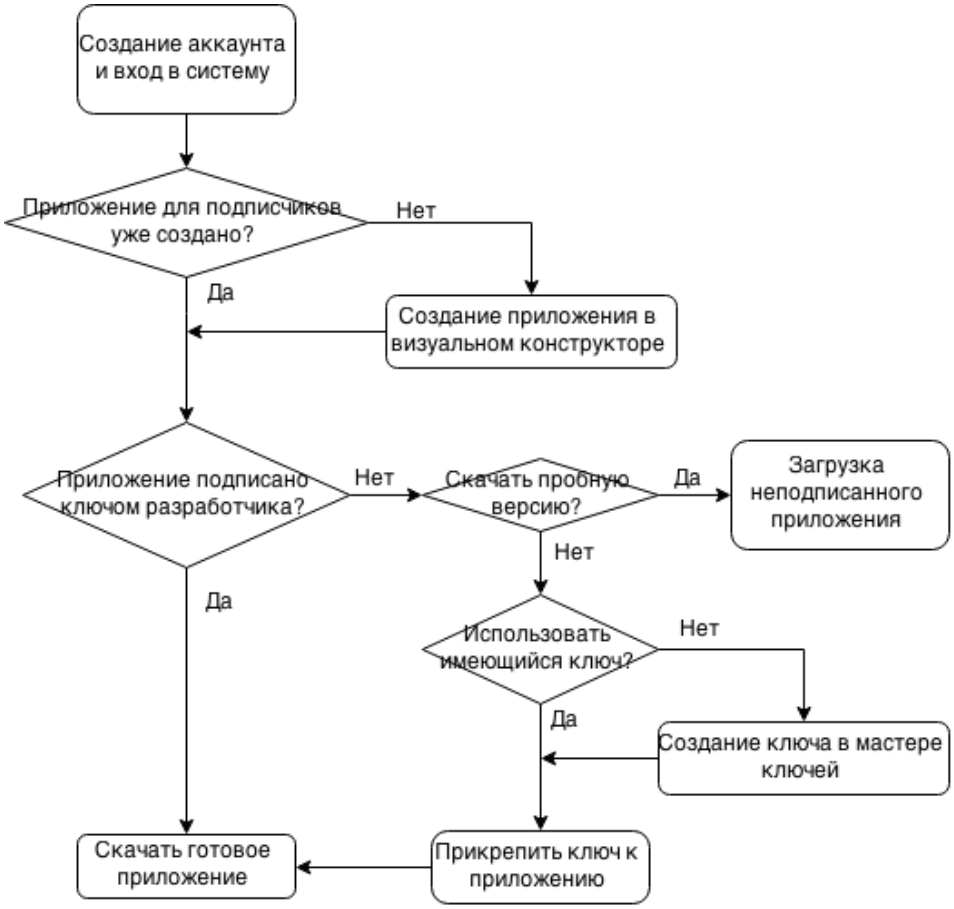
Далее, после создания приложения, пользователь может создать в нём рассылку. Действия необходимы следующие:

1) В управлении приложением необходимо создать новую рассылку – для этого нужно просто указать тему рассылки. Отметим, что тема может вводиться кириллицей – проблем с кодировкой не возникнет. Указанная тема рассылки будет указываться как заголовок всех уведомлений, отправленных в рамках этой рассылки.
2) В управлении созданной рассылкой можно отправлять уведомления. Для достижения целей, поставленных в данной работе, достаточно задавать в уведомлении текст уведомления и опциональную ссылку для перехода при открытии уведомления на устройстве получателя. Ссылка указывается, если

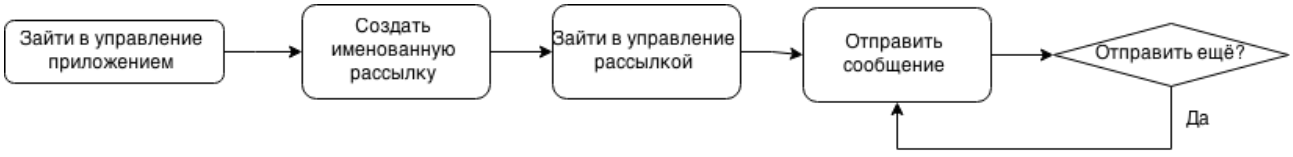
необходимо, чтобы при клике по уведомлению на устройстве пользователя открылся браузер и загрузил страницу по указанному адресу. Это удобно, когда нужно вставить в уведомление графический или звуковой контент. Если ссылка не указывается, при клике на уведомлении будет открыто наше приложение с демонстрацией логотипа при старте. Уведомление будет отправлено на все устройства с установленным в этот момент приложением сразу же после его отправки с этой страницы.

Блок-схема алгоритма создания рассылки и отправки уведомлений представлена ниже.

Алгоритм создания приложения



Алгоритм создания рассылки и отправки уведомления



Далее мы подробнее рассмотрим собираемое мобильного приложение и модель доставки сообщения после того, как оно было опубликовано в одной из рассылок.

4.2. Описание мобильного приложения

Поскольку построение приложения, предоставляющего весь необходимый функционал для любого публикатора, будь то интернет-магазин или некоммерческая организация, заведомо является очень тяжёлой (если вообще выполнимой) задачей, а создание визуального конструктора, позволяющего достаточно гибко определить входящий в приложение функционал, выходит за рамки данной дипломной работы, мы ограничимся простым приложением, которое было бы уникальным для целей публикатора, и позволяло бы оповещать пользователей о новостях рассылкой push-уведомлений.

Для этого достаточно приложения, которое:

a) При запуске отображает уникальный логотип, задаваемый публикатором в визуальном конструкторе
b) В основной части работы просто отображает историю приходивших уведомлений – чтобы подписчик мог посмотреть последние уведомления и при необходимости перейти по соответствующим ссылкам

Такое приложение вполне подойдёт публикаторам, которые хотели бы просто держать своих подписчиков в курсе последних новостей, и все подробности (в том числе графический и звуковой контент) размещали бы по связанной с уведомлением ссылке.

Для сборки приложения мы будем использовать метод программных тегов. Он заключается в том, что в программном коде приложения фрагменты, которые должны быть настроены непосредственно перед сборкой, помечаются двойными фигурными скобками и некоторым ключевым словом. Например, <image=”{{путь_к_логотипу}}”>, где «путь_к_логотипу» – ключевое слово, которое отличает этот тег ото всех остальных.

Псевдокод алгоритма, реализующего этот подход, следующий:

```
For название_тега, содержимое in список_тегов_с_содержимым_для_вставки:
    //в реальной программе проход осуществляется рекурсивно,
    ///однако мы опустим этот факт для наглядности
    For файл in папка_с_исходными_кодами:
        Заменить_подстроку(название_тега, содержимое)
```

EndFor

EndFor

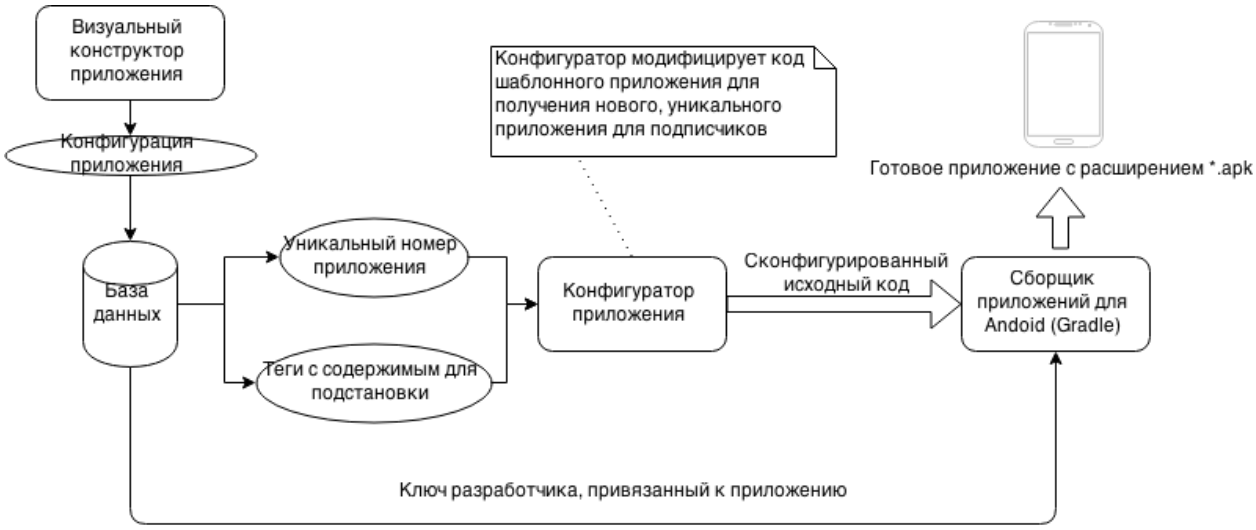
Для того, чтобы пользователь всегда мог просмотреть полученные новости, вне зависимости от наличия подключения к интернету, история уведомлений будет храниться в локальном хранилище, расположенном в файловой системе операционной системы устройства.

Сама система сборки осуществляет построение приложения в два этапа:

1) Нахождение всех тегов и подстановка вместо них конкретных значений. В нашем случае необходимо для каждого приложения указать его уникальный номер, по которому можно будет адресовать уведомления, а также все параметры, определяемые в визуальном конструкторе. То есть путь к изображению логотипа. Номер приложения используется при запуске процедуре его на сервере-публикаторе, рассылающим уведомления.
2) Когда исходный код приложения готов к сборке, система собирает из него готовое приложение, при необходимости подключая ключ разработчика, чтобы приложение получилось с возможностью публикации.

Для того, чтобы экономить место на жёстком диске файловой системы сервера, приложения в собранном виде не хранятся – достаточно хранить в базе данных все необходимые для сборки данные. Это также позволяет гарантировать, что пользователь всегда загружает актуальную версию своего приложения – в начале загрузки пользователем оно заново собирается, а по окончании удаляется с сервера.

Таким образом, схематично сборка осуществляется так:

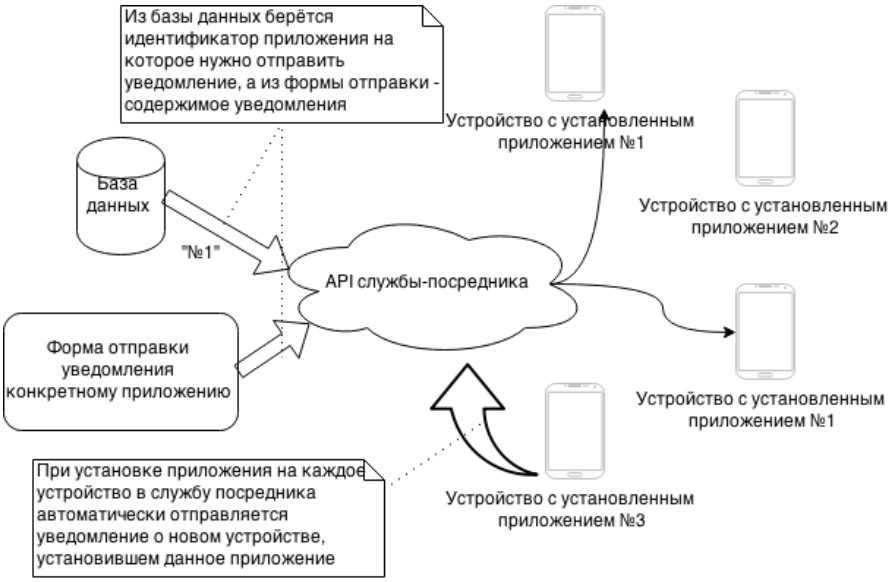


4.3. Модель доставки уведомлений

Для того, чтобы доставить собранному приложению уведомление, отправленное через личный кабинет, необходима некоторая служба-посредник, гарантирующая доставку уведомлений (то есть его хранение, отслеживание статуса доставки и т.п.). Для этого нужно либо реализовывать систему самостоятельно, либо воспользоваться одним из существующих решений, которые были перечислены в разделе обзора существующих решений.

Учитывая то, что в рамках дипломной работы большое количество отправляемых уведомлений не предполагается, а в реализации подобной службы никаких новшеств не предвидится, в качестве посредника было взято одно из существующих решений, наилучшим образом подходящее по тарифному плану – это сервис push-уведомлений Parse. Если рассматривать проект с коммерческой точки зрения, то при превышении лимита бесплатной отправки уведомлений с пользователей сервиса может взыматься плата, отвечающая тарифным планам выбранного нами сервиса.

Модель работы службы доставки уведомлений следующая: после того, как собранное приложение устанавливается на устройстве подписчика, оно регистрируется в сервисе Parse, указывая свой уникальный идентификатор. Далее, при отправке уведомления через наш веб-сервис, посредством внешнего API оно перенаправляется в сервис посредника с указанием уникального идентификатора приложения, которому его необходимо отправить. Далее сервис-посредник просматривает все зарегистрированные экземпляры приложений, выбирает из них те, которые соответствуют полученному идентификатору, и рассылает уведомление этим приложениям. Схема представлена ниже:



4.4. Описание возможных модификаций и расширений

Здесь мы перечислим примеры модификаций и расширений, которые могут быть позже реализованы:

1) Увеличение возможностей визуального конструктора. Используя методологию программных тегов, можно фактически встраивать любой код в мобильное приложение.
2) Добавление полей при отправке уведомления. Можно расширить перечень полей до соответствия набору сервиса посредника. Это может быть звуковой сигнал, или графический контент
3) По аналогии с ОС Android, можно создавать приложения для ряда других мобильных операционных систем. Нужно лишь доработать совместимые системы сборки.
4) Можно добавить автоматическое выкладывание приложений в публичный доступ, рассылку ссылок на приложение на заданный список электронных почт и так далее

4.5. Внедрение функционала, имеющегося в аналогичных системах

Поскольку в нашей работе проект изначально не позиционировался как коммерческий, большая часть функционала, выигрышно подчёркивающая удобство и потенциал сервиса в глазах пользователя, не была реализована. Акцент ставился на решении задачи (см. раздел «Постановка задачи»). Поэтому для того, чтобы систему можно было сравнить с другими, введём ряд допущений, основывающихся на том, что если продолжить работу над веб-сервисом с целью повышения его популярности, то большое количество функционала будет добавлено. Для каждого допущения, чтобы не быть голословными, также опишем приблизительную схему реализации, которая позволит внедрить интересующий функционал:

- Отложенная отправка уведомлений

Многие сервисы как push-, так и смс-уведомлений предоставляют возможность отложенной отправки уведомлений. Это удобно, например, когда необходимо отправить уведомления в нерабочее время, или если необходимо обеспечить точное время отправки.

Для внедрения подобного функционала можно использовать различные средства. Учитывая, что веб-сервис реализован на связке Python-Django, для этого идеально подойдёт пакет Celery – он позволяет выполнять задачи асинхронно, а также через заданные интервалы времени.

Аналогом такого решения было бы использование предустановленной службы cron в ОС Linux.

- Персонализация уведомлений

Персонализация уведомлений подразумевает использование в содержимом уведомления персональной информации получателя. Подобное часто обусловлено, например, маркетинговыми стратегиями.

Для реализации такого функционала необходимо, во-первых, получить средство сбора таковой информации от пользователя, и во-вторых средство указания в тексте уведомления некоторой переменной, принимающей значение, зависящее от полученной персональной информации.

Наилучшим источником персональной информации является приложение, устанавливаемое пользователем. В первую очередь, если при установке пользователь подтвердил разрешение приложению собирать персональную информацию, приложение способно собирать такую информацию в фоновом режиме и отсылать на сервер. Альтернативой является встраивание в приложение некоторой формы пользовательского ввода, в которой необходимую нам информацию пользователь укажет самостоятельно. Для хранения персональной информации можно использовать хранилище сервиса Parse и его API для загрузки/получения из него информации.

Для внедрения персональной информации в текст уведомления можно использовать уже рассмотренный нами метод тегов, заменяемых на некоторое содержимое. Например, {{name}} в тексте уведомления может автоматически заменяться на имя получателя, если таковое имеется, или некоторое предустановленное значение.

- Регулирование подписки со стороны пользователя

Не смотря на естественное желание публикатора донести информацию до как можно большего количества людей, не все обладатели приложения могут хотеть быть подписчиками той или иной рассылки. Напомним, что изначально для приложения создаётся набор рассылок, все уведомления из которых принимаются этим приложением.

Для того, чтобы пользователь мог отписаться от конкретной рассылки, достаточно дать пользователю возможность сообщить об этом из своего приложения – например, при длительном нажатии на уведомление высветить опцию «отписаться от данной рассылки». После этого можно воспользоваться вызовом к API сервиса Parse, добавив данную тему рассылки в список игнорируемых данным приложением.

- Высокая пропускная способность

В данный момент все уведомления одного пользователя отправляются синхронным образом – пока не отправится первое уведомление, невозможно отправить второе, и так далее. Это неизбежно влияет на пропускную способность сервиса. В данный момент такая реализация использовалась для того, чтобы затруднить быстрое исчерпание лимита бесплатных уведомлений, которые предоставляются сервисом Parse, однако в будущем можно запускать отправку уведомления в асинхронном режиме, например с помощью технологии AJAX, что позволит значительно повысить скорость отклика сервиса.

- Внешний API

Для целей данной работы достаточно управления рассылки из личного кабинета пользователя, однако нередко подобные действия удобнее производить из некоторого собственного приложения, или стороннего вебсайта, к которому у публикатора есть доступ. На этот случай имеет смысл предоставить возможность вызова некоторых функций API для, например, отправки уведомления в рамках уже имеющейся рассылки.

Поскольку наш веб-сервис построен по принципу REST, он уже полностью готов к интеграции в него REST API – то есть отправки специальных типов запросов по протоколу HTTP для изменения состояния сервиса. К примеру, можно указать, что при получении запроса HTTP PUT, содержащем приложение, рассылку и текст уведомления, должна производиться отправка соответствующего уведомления. Однако для того, чтобы авторизация пользователя, отправляющего подобный запрос, была надёжность, есть смысл изменить протокол взаимодействия с API сервиса на HTTPS, предоставляющий защищённое соединение.

4.6. Сравнение полученного решения с сервисом смс-рассылки

Как уже отмечалось ранее, предшественником и в некотором смысле конкурентом информационных систем на базе push-уведомлений являются системы, основанные на смс-рассылках. Помимо преимуществ полученного нами решения перед другими

аналогичными информационными системами (преимущества вытекают из формулировки поставленной в данной работе задачи), логично было бы провести сравнение с сервисом, осуществляющим смс-рассылку.

В сравнении будем рассматривать пункты, описанные в предыдущем подразделе, реализованными – это позволит сделать акцент в первую очередь не на функциональности (которую в таком случае можно реализовать приблизительно одинакового уровня), а на различии технологий push и смс самих по себе.

Отметим, что перечисленные достоинства одной системы являются недостатками другой системы. Общие достоинства и недостатки указывать в сравнении нет необходимости.

Достоинства смс-рассылок:

- Нет необходимости ставить специальное приложение
- Нет ограничения на операционную систему устройства – поддерживаются абсолютно все модели мобильных телефонов
- Для того, чтобы подписать владельца устройства к рассылке, нужен только его номер телефона
- Можно отправить уведомление «вслепую» по случайному номеру – иногда это бывает полезным
- На многих моделях телефонов для того, чтобы добавить публикатора в чёрный список, требуется установка отдельного приложения. А на некоторых это вообще не возможно (сомнительное достоинство, но всё же)
- Нет зависимости от интернета
- Простота организации – кроме подписки на сервисе рассылок вообще ни о чём больше не надо думать

Достоинства рассылок на основе push-уведомлений:

- Простой доступ к персональным данным пользователя
- Понятный способ подписаться/отписаться от рассылок как внутри приложения, так и посредством установки/удаления приложения
- Возможность организовать обратную связь средствами приложения (например, упомянутые ранее формы пользовательского ввода)
- Потенциально более широкие возможности по отображению уведомлений - например, иконка, предустановленная публикатором

- Требование к развитой операционной системе мобильного телефона позволяет гарантировать наличие таких данных, как, например, GPS-координаты. Отсюда всевозможные преимущества вроде возможности проведения геопозиционирования и отправки уведомлений в зависимости от местонахождения пользователя.
- Дешёвая стоимость отправки уведомления (в тысячи раз дешевле смс)
- В целом, с помощью технологии push-уведомлений можно построить полностью интерактивный диалог с пользователем

Как видно из этих перечней, каждая технология всё же имеет право на жизнь, обладая своими уникальными возможностями. Выбор должен производиться в зависимости от конкретной отрасли, в которой необходимо использовать рассылку уведомлений.

Однако хочется отметить, что используя в качестве основы результаты, полученные в данной работе, можно решить комплекс проблем, что позволит использовать информационные системы на основе push-уведомлений гораздо более широкому кругу пользователей.

5. Описание практической части

Все исходные коды программы выложены в публичном репозитории сервиса GitHub (ссылку для клонирования репозитория см. в [11]).

Также сразу отметим, что в силу ограничений, происходящих от API службы-посредника доставки уведомлений (см. раздел «Модель службы доставки уведомлений») поддерживается версия Android 2.3+.

5.1. Мобильное приложение

Для создания мобильного приложения под операционную систему Android сегодня предлагается довольно широкий спектр инструментов. Вот перечень некоторых из них:

- Android Studio

  Эта среда разработки позиционируется создателями Android как основное средство программирования приложений. Используется язык Java в сочетании с подмножеством языка XML для разметки рабочей области приложения. Свободно распространяется и позволяет создавать приложения без каких-либо ограничений.

- Xamarin Studio

  Среда разработки, позволяющая создавать кроссплатформенные мобильные приложения на языке C# в сочетании с теми же инструментами разметки. В отличие от предыдущего кандидата, для сборки приложений в режиме Release, то есть с возможностью распространять с ключом подписи разработчика, нужно приобрести платный аккаунт. Хотя для студентов есть возможность подключения бесплатного студенческого аккаунта.

- Titanium SDK

  Эта среда разработки позволяет создавать настольные и мобильные приложения подобно веб-страницам, используя средства HTML, CSS и JavaScript. Причём бесплатно.

- Qt (Qt Creator)

  Qt – сборник библиотек для разработки кросс-платформенных приложений с графическим интерфейсом и без него. В том числе имеется поддержка разработки под Android. Изначально подразумевалось программирование на языке c++, однако сегодня существует довольно большое количество поддерживаемых языков, к примеру C# и Python. Qt Creator – среда разработки под Qt.

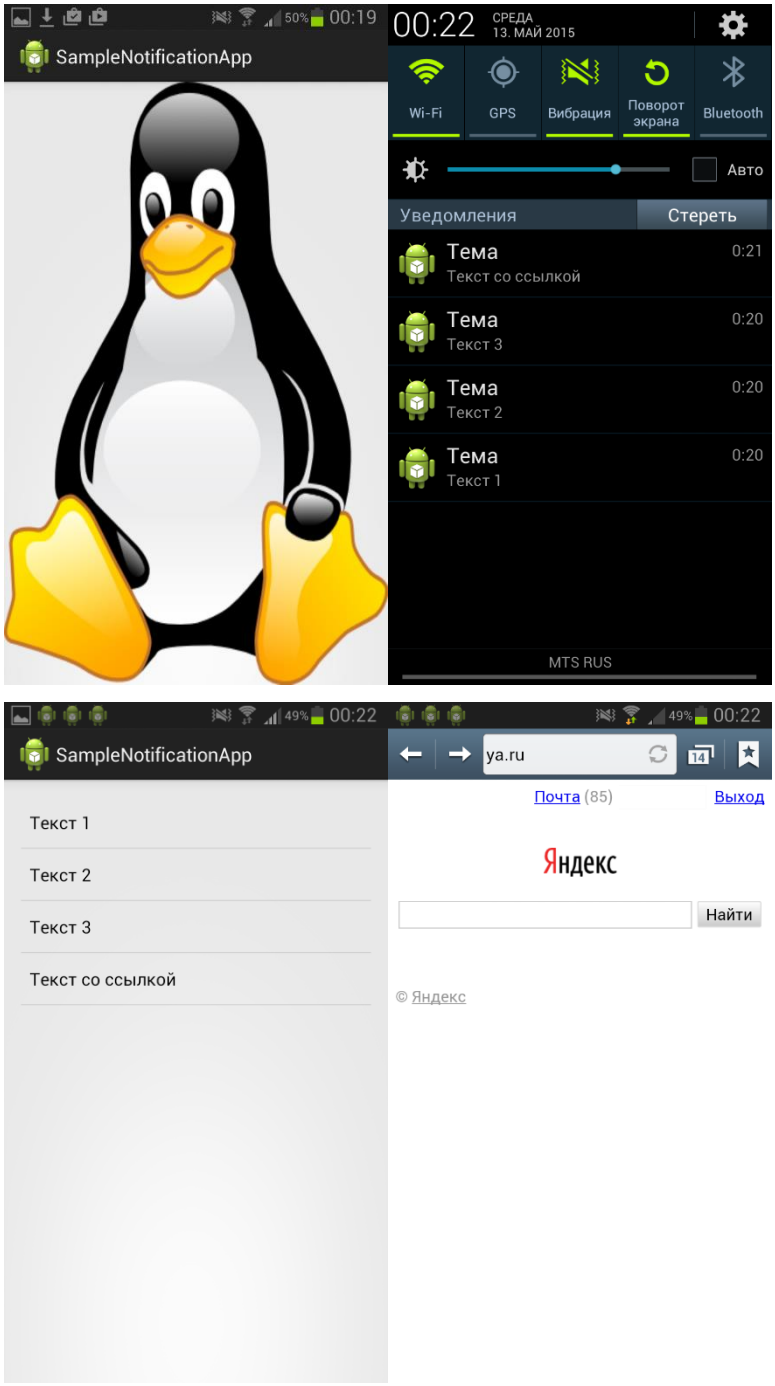
Существуют и другие решения, особенно их стало много в последние год-два в связи с резким ростом популярности мобильных платформ и устройств. Однако перечисленные выше являются одними из самых популярных.

Для разработки проекта была выбрана среда Android Studio, потому что она бесплатная и является решением от самих разработчиков Android, что означает наибольшую поддержку и гибкость разработки. К тому же в состав этой среды разработки входит консольная утилита Gradle, с помощью которой осуществляется сборка приложений. Таким образом, если установить эту утилиту и все необходимые дополнительные пакеты на сервере, то можно собирать приложение из исходного кода с помощью консольных команд. Это, в свою очередь, даёт практически не ограниченную возможность автоматизации построения мобильного приложения.

Как уже пояснялось в разделе описания решения, сервис, созданный в рамках данной работы, предоставляет визуальный конструктор для настройки шаблона приложения. Изменяя код приложения согласно изменениям, вносимым пользователем, мы получаем код уникального приложения, которое может быть скомпилировано упомянутой ранее утилитой Gradle.

Приложение носит демонстрационный характер и может быть впоследствии доработано. На данный момент оно состоит из двух частей: демонстрации логотипа при запуске и перечня полученных ранее уведомлений. Для примера, нами было собрано приложение с логотипом – изображением пингвина, а также отправлено на него 4 уведомления: 3 с обычным текстом, и 4ое с привязанной ссылкой для перехода.

Ниже показаны скриншоты мобильного экрана, демонстрирующие отображение логотипа, перечень полученных уведомлений, историю уведомлений и результат клика по уведомлению с ссылкой (мы указали http://ya.ru). Заметим, что при нажатии на уведомление без привязанной ссылки просто открывается история уведомлений с демонстрацией логотипа приложения:



5.2. Веб-сервис

В первую очередь, необходимо было определиться с используемой серверной операционной системой. Автор предпочёл операционную систему Linux, поскольку она является свободно распространяемой операционной системой, и к тому же прочно зарекомендовала себя как хорошее решение для серверной машины. Для простоты работы, использовалась версия Ubuntu 14.04.

Далее, для написания самой программы веб-сервиса, разумно использовать некоторый framework - программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта[9]. Учитывая то, что веб-сервис может потенциально расширять свой функционал, а также выбор серверной операционной системы, удобно использовать в качестве основного языка разработки язык программирования Python. Обилие библиотек для этого языка оставляет широкие возможности для расширения функционала сервиса без значительных усилий. Для данного языка одним из самых популярных фреймворков является Django Framework – его и было решено использовать. Помимо популярности, он обладает тем преимуществом, что активно поддерживает принцип DRY – don't repeat yourself (в переводе: не надо повторять самостоятельно, то есть нет смысла реализовывать то, что до вас реализовывали другие). На практике это означает широкий выбор плагинов, которые позволят в будущем расширять функциональность сервера без значительных усилий (или по крайней мере усилия будут меньшими, нежели для большинства других фреймворков).

Для хранения данных пользователей используется реляционная база данных под управлением MySQL. В первую очередь, предпочтение было отдано реляционным базам данных в отличие от не реляционных (noSQL) баз данных потому, что noSQL базы данных ещё не имеют такой хорошей поддержки в фреймворках (в том числе Django), а для наших целей различия в хранении данных между реляционными базами и не реляционными не играет большой роли. Также была выбрана именно MySQL, а не другая подобная СУБД, вроде PostgreSQL, поскольку MySQL имеет более удобную, с точки зрения автора, консоль администратора, а в целом для требований, определяемым нашим сервисом, различие между этими СУБД не является значительным.

Таким образом, веб-сервис предоставляет в публичный доступ вебсайт, на котором пользователи-публикаторы создают приложения и отправляют уведомления, а также в целом управляют личным кабинетом. Для управления используются обычные формы пользовательского ввода, передаваемые методом HTTP POST. Для отправки уведомлений, как уже было описано в разделе описания решения, используется API веб-сервиса Parse – в нашем случае это просто отправка POST запроса с соответствующими параметрами:

```
import json, httplib
connection = httplib.HTTPSConnection('api.parse.com', 443)
connection.connect()
request = {
   "where": {
```

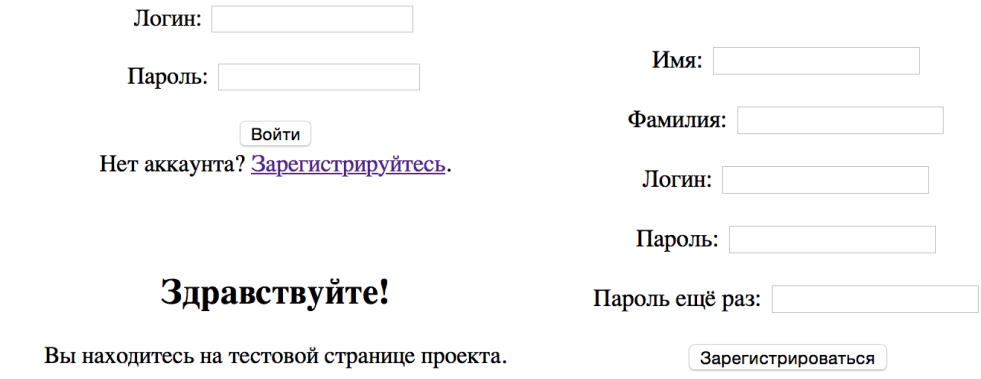
```
        "deviceType": "android",
        "appID": "%i"%message.theme.application.id,
    },
    "data": {
        "alert": message.text,
        "title": message.theme.name,
    }
}
if message.url is not None and len(message.url) > 0:
    request["data"]["uri"] = message.url

connection.request('POST', '/1/push', json.dumps(request), {
       "X-Parse-Application-Id": "SOME_ID", #hidden
       "X-Parse-REST-API-Key": "SOME_KEY", #hidden
       "Content-Type": "application/json"
})
result = json.loads(connection.getresponse().read())
return result
```

Для сборки приложения создаётся копия исходного кода шаблона, модифицируется согласно уникальным настройкам приложения, заданным пользователем в визуальном конструкторе, после чего производится сборка приложения и выдача готового *.APK файла пользователю для загрузки.

Далее показаны скриншоты страниц сервиса с комментариями:

1) Страницы входа и регистрации, а также основная страница для уже вошедшего пользователя



2) Страница с перечнем созданных приложений. Приложение можно скачать, или перейти к связанным с ним рассылкам

Это Ваш личный кабинет. Здесь вы можете создавать мобильные приложения и управлять рассылками.

Название	Описание Приложения	Ключ Подписи	Создано	Actions
app	desc	alias	23:27 2015-05-04	Подробнее/Скачать

1 mobile app

Создать новое приложение

Перейти к управлению ключами

3) Страница рассылок конкретного приложения

Название приложения: app

Отписание приложения: desc

Ключ подписи: alias

Название Рассылки	Создана	Actions
Тема	23:52 2015-05-04	Подробнее/Удалить

1 theme

Создать новую рассылку: Создать рассылку

Выбрать новый ключ подписи: --------- Изменить

4) Страница конкретной рассылки. Можно отправлять уведомления со ссылкой и без неё

Главная » Личный кабинет » Приложение 'app' » Рассылка 'Тема'

Дата Отправки	Текст Сообщения	Ссылка Для Перехода
05/05/2015 2:55 a.m.	Это супер поисковик	http://www.google.ru/
05/13/2015 12:20 a.m.	Текст 1	—
05/13/2015 12:20 a.m.	Текст 2	—
05/13/2015 12:20 a.m.	Текст 3	—
05/13/2015 12:21 a.m.	Текст со ссылкой	http://ya.ru/

5 messages

Текст сообщения:	
Ссылка для перехода (не обязательно):	

Отправить сообщение

5) Страница создания ключей подписи приложения

Имя Ключа (Алиас)	Создан	Actions
alias	23:27 2015-05-04	Удалить

1 app key

Все поля должны заполняться, используя цифры и латинские буквы, подчёркивание и дефис!

Организационный юнит:	
Название организации:	
Город:	
Область/Регион:	
Код страны:	---------
Название ключа (алиас):	
Пароль для хранилища ключа (не менее 6ти символов):	
Повторите пароль алиаса:	
Пароль ключа (не менее 6ти символов):	
Повторите пароль ключа:	

Создать ключ

6. Заключение

В рамках дипломной работы разработана информационная система, позволяющая пользователю, не имеющему навыков программирования, осуществлять рассылку информации на базе push-уведомлений как группам людей, так и отдельным пользователям, желающим быть подписчиками рассылки.

7. Список литературы

1) Solovey A., What is Push-notifications and how to use them // Интернет-маркетинг, теория и практика, 2014 [HTML] (http://marketingbuzz.info/chto-takoe-push-uvedomleniya-i-kak-ix-pravilno-ispolzovat.html)
2) Vorona T. Ukraine startap Jeapie got 50$ of investments from Digital Future [URL] // (http://ain.ua/2014/11/18/550918)
3) New statistics of usage mobile operation systems // Портал 4PDA 02.2014 [URL] (http://4pda.ru/2014/2/13/141037/)
4) Павлов А. Д., Намиот Д. Е. Информационные системы на основе push-уведомлений //International Journal of Open Information Technologies. – 2014. – Т. 2. – №. 8. – С. 11-19.
5) Mathew S., Varia J. Overview of amazon web services //Amazon Whitepapers. – 2013.
6) Webber-Cross G. Learning Windows Azure Mobile Services for Windows 8 and Windows Phone 8. – Packt Publishing, 2014.
7) Jeapie // Push notification service for web and mobile developers [URL] (http://jeapie.com/features)
8) William S. Davis, David C. Yen. // The Information System Consultant's Handbook. Systems Analysis and Design.: CRC Press, 1998
9) Фаронов В. Создание приложений с помощью C# //Руководство программиста. М.: Эксмо. – 2008.
10) Ворошилов В. В. Журналистика: учебник. – Изд-во Михайлова ВА, 2002
11) Федченко И.О. Репозиторий с исходными кодами программ дипломной работы «Информационные системы на базе push-уведомлений» https://github.com/IgorFedchenko/NotificationService.git
12) HOLOVATY A., KAPLAN-MOSS J. The Django Book: Version 2.0 //The Django Book. – 2009. – Т. 16.
13) Harms D. D., McDonald K. The Quick Python Book. – Manning, 2000. – С. 422.
14) Richardson L., Ruby S. RESTful web services. – " O'Reilly Media, Inc.", 2008.
15) Kincaid J. Urban Airship Brings Easy Push Notifications to Android //TechCrunch, Aug. – 2010. – Т. 10.

16) Rogers R. et al. Android application development: Programming with the Google SDK. – O'Reilly Media, Inc., 2009.
17) Burnette E. Hello, Android: introducing Google's mobile development platform. – Pragmatic Bookshelf, 2009.
18) MySQL A. B. MySQL reference manual. – 2001.