Санкт-Петербургский Государственный Университет

Программная инженерия

Гонта Ксения Алексеевна

Создание расширяемой инфраструктуры групп приложений под платформу iOS

Выпуская квалификационная работа

Научный руководитель:
доцент кафедры системного программирования Брыксин Т. А.

Рецензент:
ведущий разработчик ООО “НМТ — Новые Мобильные Технологии” Невоструев К. Н.

Санкт-Петербург
2017

SAINT PETERSBURG STATE UNIVERSITY

Software engineering

Kseniya Gonta

Development of scalable infrastructure for groups of iOS applications

Graduation Thesis

Scientific supervisor:
Associate Prof. Bryksin T. A.

Reviewer:
Nevostruev K. N.

Saint Petersburg
2017

Оглавление

Введение

В настоящее время компании стремятся максимально упростить взаимодействие между сотрудниками посредством внедрения различных IT-решений, таких как системы электронного документооборота [10], порталы, корпоративная почта, сервисы закупок и так далее.

Подобные сервисы, обычно, по своей сути однотипны, поэтому компании предпочитают не разрабатывать решения самостоятельно, а обращаются к IT-компаниям, предоставляющим услуги по автоматизации бизнес-процессов. Последние разрабатывают решения с нуля или же настраивают имеющиеся у них проекты под нужды заказчика.

Одним из таких решений становятся мобильные приложения, в том числе для платформы iOS [12]. При этом бывают ситуации, когда на iOS-устройстве сотрудника установлено несколько приложений, относящихся к деятельности компании. С ростом числа таких приложений на одном устройстве у пользователей появляется необходимость в наличии единой инфраструктуры для работы с ними, предоставляющей навигацию между приложениями компании, агрегацию данных из них и последующее отображение агрегированных данных.

Отдельного внимания заслуживает вопрос установки на длительный срок новых приложений на различные устройства. Для распространения приложений, разработанных для ограниченного круга пользователей, Apple предоставляет несколько схем: публикация приложения в AppStore с последующей авторизацией, распространение внутри одной компании при помощи корпоративной схемы (минуя AppStore), а также Business-to-Business (B2B) программа, позволяющая публиковать приложения в Business Store, предоставляя доступ к ним ограниченному кругу людей, указанному разработчиком. B2B программа, однако, на данный момент не открыта для использования в России.

Если с установкой приложения из App Store трудностей у пользователей не возникает, то при использовании корпоративной схемы [2], [6] у компании есть установочные файлы приложений, которые необходимо распространить между всеми сотрудниками так, чтобы процесс установки новых приложений на устройства и их обновление был максимально прозрачным.

Постановка задачи

Целью данной работы является разработка единой инфраструктуры групп iOS-приложений от одного разработчика. Для этого необходимо решить следующие задачи:

- Разработать iOS-фреймворк, интегрируемый в приложения из группы, который позволит быстро переходить между приложениями;
- Разработать базовое iOS-приложение, позволяющее навигироваться между приложениями, агрегировать информацию из них, а также отображать неустановленные приложения и давать возможность их установить;
- Разработать iOS-виджет, который будет отображать последние и наиболее часто используемые приложения.

При решении поставленных задач необходимо было учесть, что на устройстве могут со временем появляться новые приложения от этого разработчика, поэтому список доступных приложений должен быть динамически формируемым, а не статическим. Кроме того, у компании Apple достаточно жёсткая политика конфиденциальности, которая не позволяет получать из одного приложения сведения о другом, поэтому каждое приложение из группы должно «добровольно» предоставлять информацию о себе.

1. Обзор используемых технологий

1.1. Группы приложений

Изначально все iOS-приложения абсолютно изолированы друг от друга, они не могут напрямую обмениваться данными, даже если созданы одним разработчиком. Однако, им нередко требуется обмен информаций.

Для решения этой задачи два и более приложений могут быть объединены в одну группу приложений, или App Group. У всех входящих в эту группу есть доступ к общим настройкам, в которые можно записывать информацию и откуда получать её при необходимости.

У групп есть несколько особенностей:

- приложения от различных разработчиков невозможно объединить в одну группу;
- приложение может находиться только в одной группе;
- группа не может существовать без приложений, поэтому она удаляется как только удаляется последнее приложение, входящее в группу.

1.2. Общее хранилище данных

В iOS существует несколько механизмов обмена данными между приложениями. Так, например, любой объект,сериализуемый в NSData, может быть передан через кастомные URL схемы. Механизм AirDrop также позволяет передавать данные из приложения в приложение. Однако, оба эти способа обеспечивают взаимодействие между конкретными двумя приложениями, а не группы приложений, поэтому они не применимы в данной работе.

NSUserDefaults – это класс, предоставляющий программный интерфейс для взаимодействия с системой настроек по умолчанию.

При помощи NSUserDefaults можно сохранять настройки, относящиеся к приложению или пользовательским данным, например, сохранить фотографию профиля, загруженную пользователем, или сохранить цвет фона приложения.

При объединении приложений в группу всем участникам становится доступен один и тот же экземпляр NSUserDefaults. Он инициализируется со своим собственным идентификатором группы, задаваемым строкой, например, "com.example.domain.MyApplication". Экземпляр настроек хранится на устройстве до тех пор, пока установлено хотя бы одно приложение из группы.

Обмен данными между приложениями можно реализовывать через сеть, но этот способ не всегда целесообразен, так как влечёт за собой задержки в передачи информации.

1.3. App Extension + Widget

App extension (расширение приложения) [11] – это метод взаимодействия с приложением без его непосредственного запуска. Расширения не являются отдельными приложениями, они предоставляют функциональность расширяемого приложения и обычно ориентированы на выполнение какой-то одной задачи.

Существует несколько типов расширений, например:

- Today – в центре уведомлений на вкладке Today показывает краткую информацию и позволяет выполнять несложные задачи;
- Share – позволяет приложению делиться контентом с пользователями в социальных сетях и других сервисах;
- Photo Editing – предоставляет возможность редактирования фото и видео без запуска приложения Photos;
- Document Provider – необходим для разрешения другим приложениям доступа к документам текущего приложения.

Widget – это один из типов расширений приложений, отображающийся в центре уведомлений [3]. Хотя расширение распространяется как часть приложения, оно работает в отдельном потоке. Между расширением и приложением (как и между обычными приложениями) также нет возможности непосредственного обмена информацией, поэтому расширения необходимо добавлять в ту же группу, где находится приложение.

1.4. URL Scheme

URL – это тип, который потенциально может содержать местоположение ресурса на удалённом сервере, локальный путь к файлу на диске или даже произвольного фрагмента закодированных данных. URL схема, соответственно, описывает путь к ресурсу [1].

Для открытия одного приложения из другого [9] существует механизм URL схем. Так, например, если на iPhone установлено приложение Instagram, то при введении “instagram://” в строке поиска Safari откроется соответствующее приложение. Для того, чтобы приложение можно было вызвать подобным образом, у него должна быть задана уникальная URL схема. К сожалению, это невозможно сделать из кода и приходится прибегать к ручному заданию схемы через XCode, но для каждого приложения схему необходимо задать всего один раз.

1.5. Framework + Bundle

При разработке библиотеки главная цель – получить её в таком формате, который требует минимум усилий и знаний при её интеграции в другие приложения. В iOS фреймворк [6] – это способ удобного хранения библиотеки. Он представляет собой директорию, которая содержит библиотеку и заголовочные файлы. Для интеграции необходимо перетащить фреймворк к себе в проект, а среда разработки XCode сама выполнит необходимые действия.

Скомпилированную библиотеку необходимо распространять вместе с ресурсами: файлам с описаниями фрагментов пользовательских интерфейсов, заголовочными файлами и т.д. Фреймворк может содержать в себе папку с ресурсами, но Xcode её проигнорирует.

Для хранения ресурсов существует iOS bundle – особый вид папки, содержимое которой следует определённой структуре. Таким образом, ресурсы размещаются в bundle и распространяются вместе с фреймворком.

К тому же есть потребность запускать код библиотеки на всех возможных для iOS архитектурах процессоров. На данный момент актуальны 3 архитектуры для устройств (armv7, armv7s, arm64) [5] и 2 для симуляторов (i386, x86_64) [4]. Для этого необходимо собирать библиотеки для всех конфигураций оборудования, под которым будет работать приложение.

1.6. IPA файл

Файл с расширением .ipa – это архив iOS-приложения [8]. Каждый такой файл состоит из бинарного файла ARM-архитектуры и может быть установлен на iOS-устройства. Файл можно распаковать, если поменять расширение на zip и разархивировать.

1.7. Сервер

При использовании корпоративной программы развёртывания приложения необходимо распространять среди сотрудников, минуя AppStore. Для этого целесообразно создать сервер.

Для разработки сервера был выбран язык Python [7]. Сервер был необходим лишь для тестирования приложения, а Python обладает огромными функциональными возможностями и отлично подходит для прототипирования.

2. Реализация решения

2.1. Архитектура решения

На рисунке 1 представлена архитектура решения для платформы iOS. Основными компонентами инфраструктуры являются iOS-фреймворк, базовое iOS-приложение, виджет для центра уведомлений и веб-сервер.

Приложение условно называется базовым, так как в нём демонстрируется основная функциональность, которую необходимо было реализовать в ходе данной работы, а именно: возможность навигации между приложениями, агрегация данных из приложений одной группы, взаимодействие с веб-сервером и возможность установки приложений на устройство.

Общее хранилище данных – это общие настройки для приложений, входящих в одну группу.

Фреймворк отвечает за обмен информацией между приложениями. Для обеспечения этого он должен быть встроен в каждое приложение группы. В библиотеке существует компонента, загружающая данные, которая обеспечивает наличие актуальной информации о приложениях в общем хранилище данных: при помощи неё приложение сохраняет в хранилище информацию о себе и получает информацию о других.

В базовое приложение тоже встроен фреймворк, то есть приложение обладает всей его функциональность. Кроме того, оно занимается взаимодействием с веб-сервером. На сервере хранятся установочные файлы приложений. Базовое приложение проходит авторизацию и при помощи класса “Загрузчик ipa-файлов” получает ссылки на доступные файлы.

2.1.1. Общие настройки приложений

В центре рисунка 1 обозначено общее хранилище данных, к которому имеют доступ все приложения одной группы. В нём содержатся сведения о всех приложениях текущего разработчика,

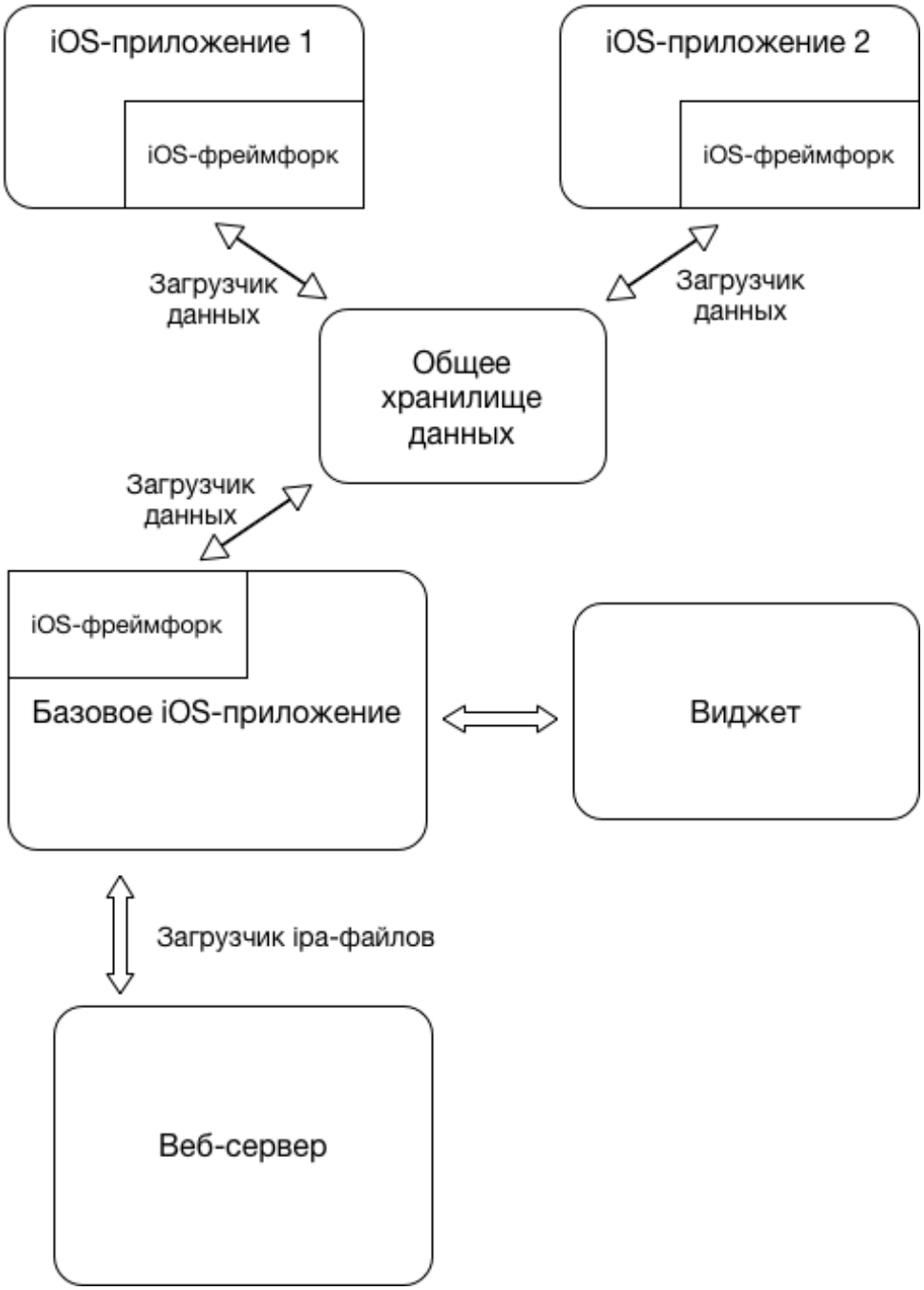

Рис. 1: Общая архитектура предлагаемого решения

установленных на устройстве, его структура представлена на рисунке 2. Для каждого приложения есть поля для записи его названия, иконки и URL схемы, по которой его можно вызывать. При необходимости можно увеличить количество полей информации о каждом приложении.

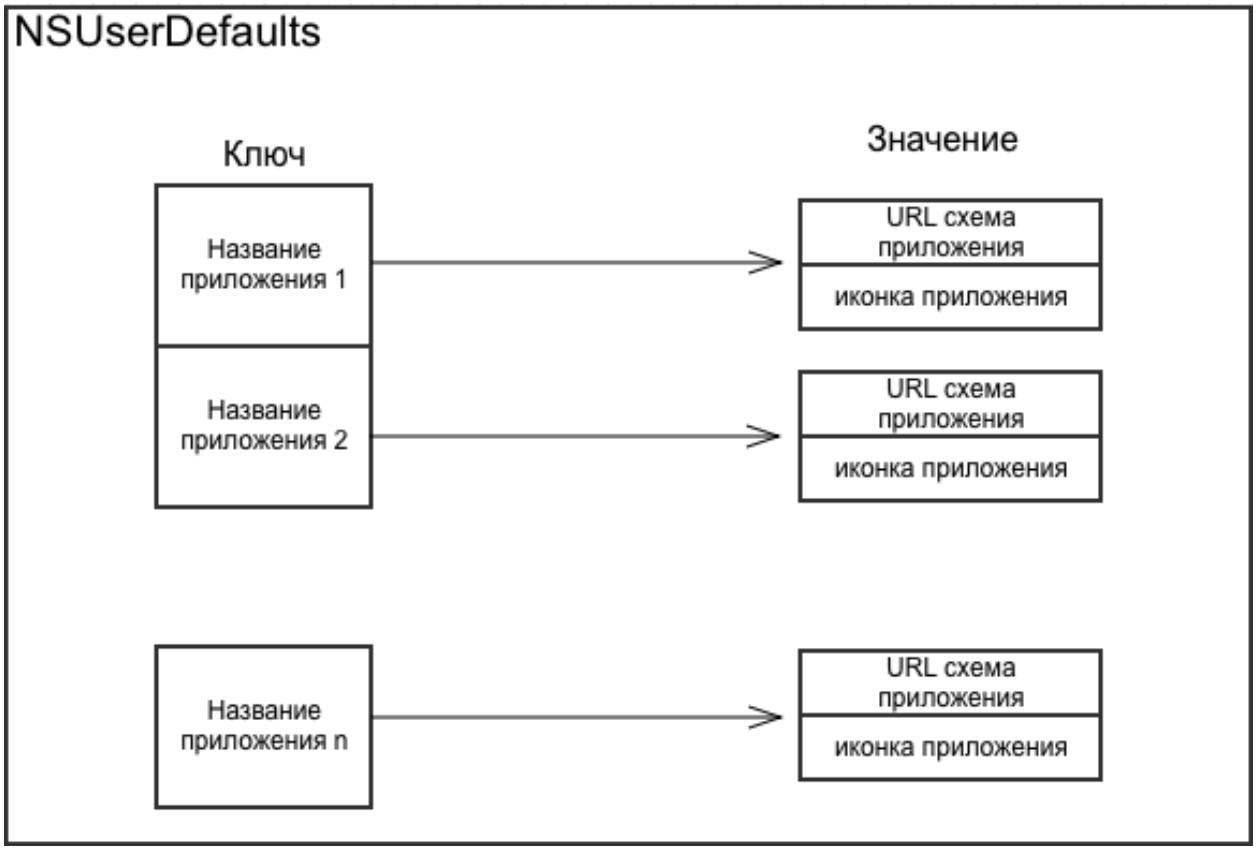

Рис. 2: Устройство общего хранилища данных

2.1.2. iOS-фреймворк

При первом запуске приложения на устройстве класс, загружающий данные, заполняет основную информацию о нём (название, иконка и URL схема) в общих настройках текущей группы приложений.

В библиотеке фреймворка есть класс CollectionViewController, который в виде коллекции ячеек отображает все установленные на устройстве приложения и при нажатии на ячейку перенаправляет пользователя в соответствующее приложение. Для хранения внешнего вида коллекции ячеек используется стандартный для iOS формат xib-файлов (XML Interface Builder). Файлы такого типа упаковываются в bundle и распространяются вместе с фреймворком. Файлы, находящиеся во фреймворке и в bundle могут ссылаться друг на друга.

Для получения информации об установленных приложениях класс, загружающий данные (рисунок 3), обращается к общему хранилищу

данных и запрашивает оттуда всю имеющуюся информацию.

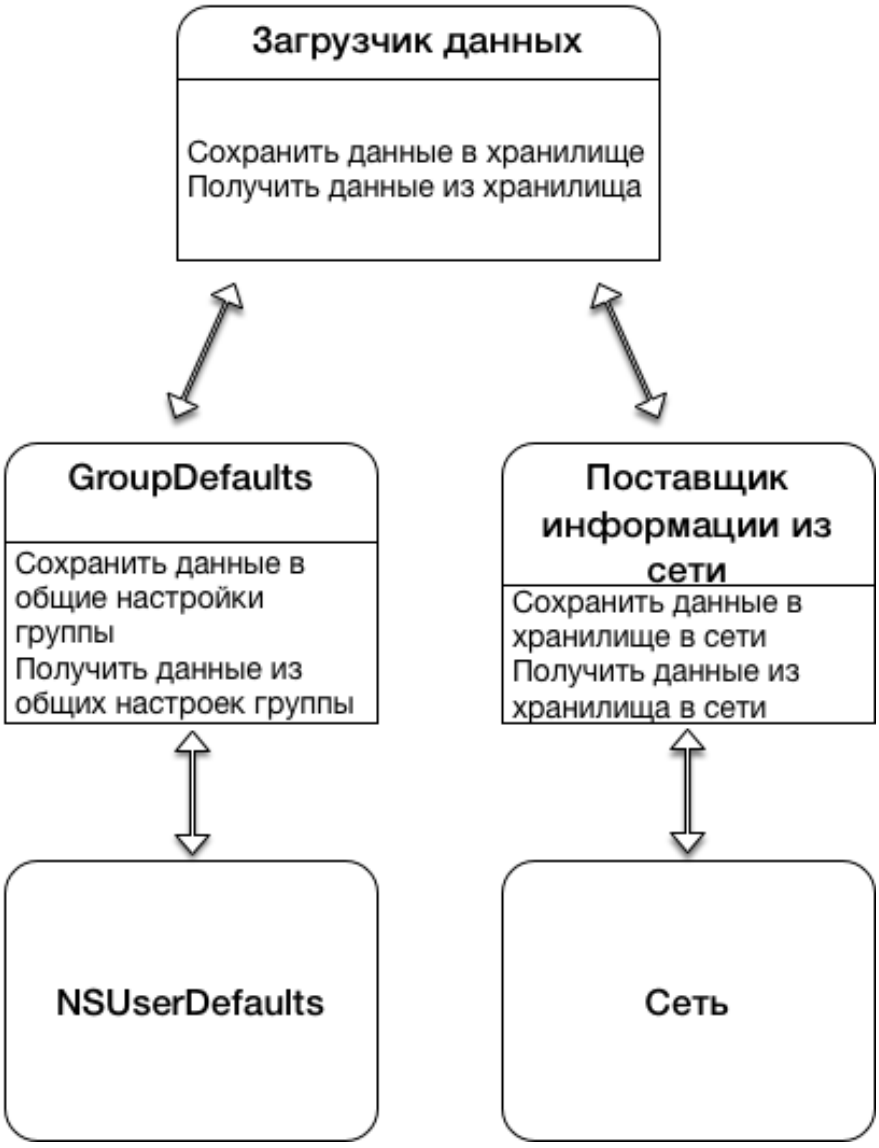

Рис. 3: Схема работы загрузчика данных

Впоследствии способ хранения информации о приложениях может измениться, поэтому класс, загружающий данные, абстрагируется от источника информации. Можно выбирать поставщика данных, в данное случае за это отвечает класс GroupDefaults.

При удалении приложения с устройства запись о нём не стирается из общего хранилища. Перед отображением пользователю приложений необходимо проверять, что ни одно из приложений, перечисленных в хранилище, не было удалено. Для этого существует функция CanOpenUrl, которой можно передать схему приложения, и

она вернёт, откликается ли какое-нибудь из установленных приложений на эту схему. Для каждого приложения создаётся уникальная схема, поэтому коллизий не будет.

2.1.3. Базовое iOS-приложения

В базовое приложение включена библиотека со всей функциональностью, то есть с отображением всех установленных на телефоне приложений и навигации между ними (рисунок 4).

Помимо этого добавлена агрегация однотипных данных, а именно, количество уведомлений. Хранение дополнительной информации достигается при помощи увеличения количества полей информации о каждом приложении в общих настройках группы. Число количества уведомлений отображается в стандартном для iOS виде: в красном кружке в верхнем правом углу иконки. В таком же виде можно хранить любую интересующую информацию.

Кроме того, у базового приложения есть возможность предлагать пользователю устанавливать отсутствующие приложения. В рамках работы был написан протитип веб-сервера, на котором хранится информация о том, какие приложения доступны для скачивания определённому пользователю. На сервер отправляется логин и пароль пользователя, и в ответ приходит JSON файл со ссылками на все приложения, которые доступны пользователю. Если какое-то приложение доступно, но не установлено, то пользователю предлагается его установить. По нажатию пользователь переходит по ссылке, где лежит установочный файл, и дальше iOS сама занимается установкой.

2.1.4. iOS-виджет

Одним из компонентов инфраструктуры является виджет. Виджетами называют расширения приложений, которые отображают информацию в Центре Уведомлений на экране “Сегодня” в iOS и, следовательно, призваны показывать ту информацию, которая важна

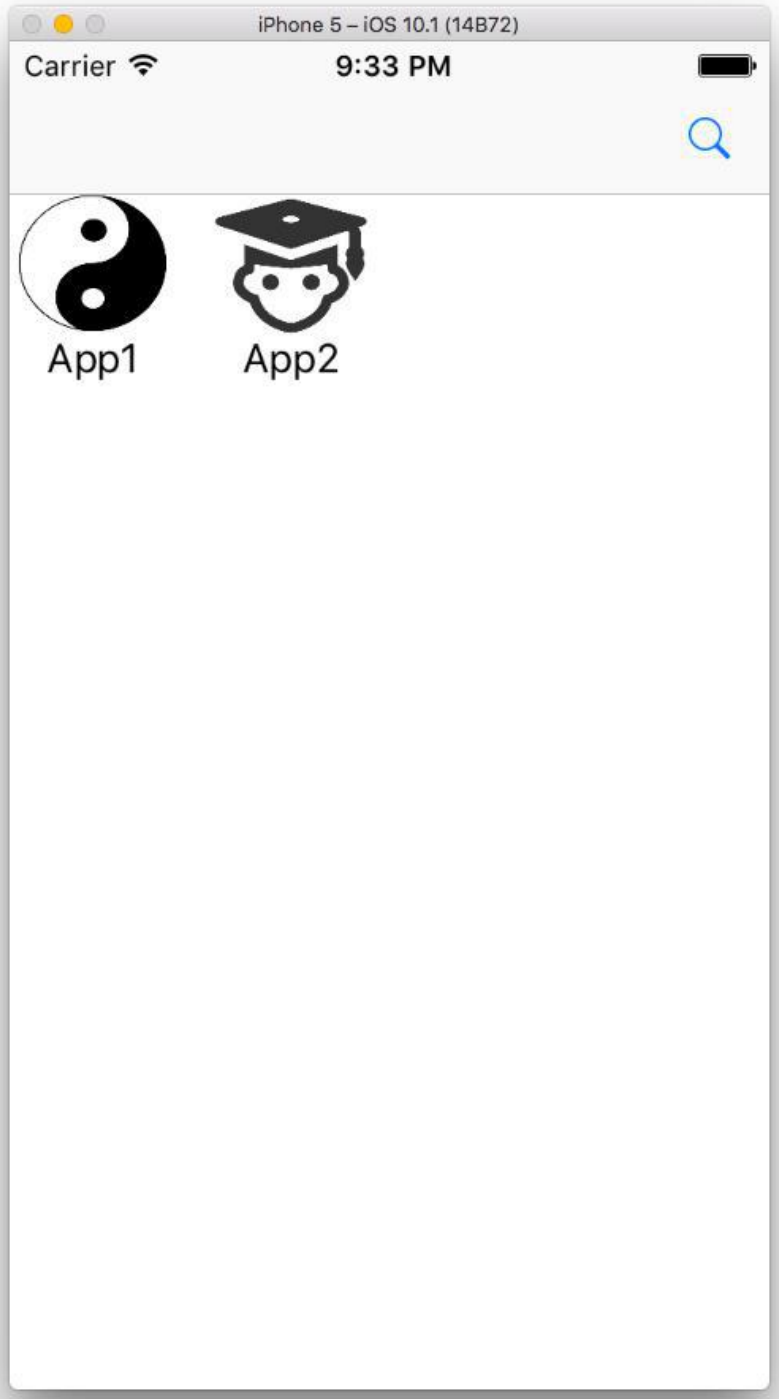

Рис. 4: Пример использования фреймворка

в текущий момент. Когда пользователь открывает экран “Сегодня”, то он ожидает, что интересующая его информация будет мгновенно доступна. Виджет становится доступным после того, как пользователь

установит приложение, содержащее виджет.

Виджет отображает наиболее популярные у пользователя приложения из группы. При нажатие на иконку происходит навигация в соответствующее приложение.

Заключение

В рамках работы была разработана единая инфраструктура групп iOS-приложений, созданных одним разработчиком. Были выполнены следующие задачи:

- Разработан iOS-фреймворк, интегрируемый в приложения из группы, который позволяет быстро переходить между приложениями;
- Разработано базовое iOS-приложение, которое агрегирует информацию из всех других приложений (и позволять переходить в эти приложения);
- Разработать iOS-виджет, который отображает последние и наиболее часто используемые приложения.

Список литературы

[1] Uniform resource locators (URL) : Rep. ; Executor: Tim Berners-Lee, Larry Masinter, Mark McCahill : 1994.

[2] Cox Jack, Jones Nathan, Szumski John. Professional iOS network programming: connecting the enterprise to the iPhone and iPad. — John Wiley & Sons, 2012.

[3] Rogers T Michael. Advanced iOS 8 Features // Swift Recipes. — Springer, 2015. — P. 277–301.

[4] Saini A. An Overview of the Intel Pentium™ Processor; Intel Corp. Santa Clara. — 1993.

[5] Singh Mahendra Pratap, Jain Manoj Kumar. Evolution of processor architecture in mobile phones // International Journal of Computer Applications. — 2014. — Vol. 90, no. 4.

[6] Turner James. Developing Enterprise IOS Applications: IPhone and IPad Apps for Companies and Organizations. — " O'Reilly Media, Inc.", 2011.

[7] Zelle John M. Python programming: an introduction to computer science. — Franklin, Beedle & Associates, Inc., 2004.

[8] ipa расширение файла. — URL: `https://www.reviversoft.com/ru/file-extensions/ipa`.

[9] launchServiceProgrammingGuide. — URL: `https://developer.apple.com/library/content/documentation/Carbon/Conceptual/LaunchServicesConcepts/LSCIntro/LSCIntro.html`.

[10] Круковский МЮ. Критерии эффективности систем электронного документооборота. — 2005. — P. 107–111.

[11] Расширения приложений. — URL: https://developer.apple.com/library/content/documentation/General/Conceptual/ExtensibilityPG/.

[12] Сивченко О, Нахавандипур Вандад. iOS. Разработка приложений iPhone, iPad и iPod. — 2013.