Ключевые слова: математическая модель, неопределенность, оптимизация, многономенклатурный запас.

Mathematical model of optimization cost and terms of multi-item inventory management/ D.O. Markozov //Bulletin of NTU "KhPI". Series: New desicions of modern technologies. – Kharkov: NTU "KhPI", 2014.-№ 17 (1060).- P.56-62. Bibliogr.:7 . ISSN 2079-5459

The mathematical model to optimize the management multi-item stocks under uncertainty. Demonstrated that the use of this model can simultaneously reduce the duration of the work and minimize the total cost of the project, and thus improve the efficiency of the company.

Keywords: mathematical model, uncertainty, optimization, multi-item stock.

УДК 665.9

***Т. Б. ШАТОВСКАЯ*,** канд. техн. наук, доц., ХНУРЭ, Харьков;
В. А. МАРИН, студент, ХНУРЭ, Харьков

РАЗРАБОТКА ВЕБ-СЕРВИСА ДЛЯ ПРЕДОСТАВЛЕНИЯ УСЛУГ ХРАНЕНИЯ ДАННЫХ

В настоящее время развит рынок мобильных технологий, и многие разработчики мобильных приложений сталкиваются с проблемой хранения данных из приложений в сети. В этой работе мы представляем структуру системы для хранения множества данных для различных приложений.

Ключевые слова: сервис, данные, мобильные приложения, хранилища данных, наборы данных.

Введение. Разработка мобильных приложений и мобильных сервисов на сегодняшний день одно из наиболее динамично развивающихся направлений в программировании. Множество разработчиков мобильного ПО сталкиваются с проблемой хранения и обмена данными между пользователями приложения [1]. Для этого разработчикам приходиться создавать дополнительные сервисы. Это в свою очередь требует дополнительных знаний и времени [2].

Поскольку реализуемый серверный функционал часто является однотипным, то имеет смысл предоставить пользователю уже готовые функции, которые объединяют наиболее общие задачи хранения информации [3]. В связи с этим сейчас находят распространения BaaS (Backendasa Service).

BaaS также известный как MbaaS (Mobile Backendasa Service) [4] представляет собой модель для предоставления услуг разработчикам мобильных приложений связать свои приложения с облачным хранилищем данных. Эти услуги предоставляются через использование SDK (Software Development Kits)и API (Application Programming Interfaces) [5].

Цель работы. Целью работы является разработка веб-сервиса для хранения пользовательских данных с гибкой архитектурой способной удовлетворять потребности разработчиков мобильных приложений.

Методы реализации. Для реализации серверной части был выбран Node.js. Node.jsэто программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код) превращающая JavaScript из узко специализированного языка в язык общего назначения. Node.js применяется

преимущественно на сервере. Технология Nodebыла задумана как платформа создания приложений, ориентированных на высокую интенсивность ввода/вывода и невысокую интенсивность вычислений.

В качестве базы данных для нашего сервиса была выбрана NoSQLдокументо-ориентированная база данных MongoDB.

Для реализации iOSSDKбыл выбран ObjectiveC, для AndroidJava.

Архитектура. Важнейшей частью является разработка архитектуры системы, для этого нужно поставить несколько вопросов касающихся нашего сервиса

• Требования — определить основные задачи программного продукта;

• Методы разработки проекта — из чего будет состоять проект, зачем нужна каждая из частей и как эти части должны быть устроены.

1. Требования. Продукт должен быть масштабируемым и переносимым. Каждая составляющая часть сервиса(модуль) должна быть независима, изменение одного модуля недолжно сказываться на работоспособности системы. Добавление нового функционала не должно изменять существующий.

Связь между модулями будет обеспечиваться по технологии REST(RepresentationalStateTransfer). Данные будут передаваться в формате JSON(JavaScriptObjectNotation).

2. Методы разработки проекта. Проект будет состоять из нескольких модулей. А именно:

• Модуль данных;

• Модуль доступа к сервису;

• SDK;

• Панель управления.

Модуль данных будет предоставлять APIдля доступа к базе данных. API включает в себя стандартные функции CRUD(Create, Read, Update, Delete), доступ к данным будет предоставляться по протоколу HTTP.

Модуль доступа к сервису предоставляетAPIдля доступа к модулю данных. Задача сервера доступа это получение пользователя по его «Api-key» и проверка запроса, на существование коллекции и на соответствие схеме данных. Если запрос удовлетворяет всем требованиям, то тогда он передаётся на обработку в модуль данных.

Пользовательские данные будут храниться в формате документа. На рис. 1 представлен пример пользовательских данных.

SDKбудет предоставлять классыдля работы с модулями через модуль доступа к сервису.С появлением новых модулей, функционал SDKрасширяется.


```
{
    "created":"2014-03-26T16:14:51.333Z",
    "dataServer":"http://data1.myBaas.home",
    "name":"2D Race",
    "user":"532f281f045c9749dd712bf0",
    "schemas":[
        {
            "_id":"53330040b75e49501a833fcd",
            "name":"Scores",
            "fields":[
                        {
                        "fieldName":"Score",
                        "fieldType":"Integer"
                        },
                        {
                        "fieldName":"userName",
                        "fieldType":"String"
                        },
                        {
                        "fieldName":"dateCreate",
                        "fieldType":"Date"
                        },
                        {
                        "fieldName":"dateUpdate",
                        "fieldType":"Date"
                        },
                        {
                        "fieldName":"_id",
                        "fieldType":"ObjectId"
                        }
                    ]
        }
    ]
}
```


Рис. 1–Пример пользовательских данных

3. Итоги. Будет использоваться Сервис-ориентированная архитектура(SOA), модули будут общаться между собой по технологии REST, формат ответа JSON.

Для уменьшения нагрузки на модуль данных можно делить пользователей на группы и использовать 2 модуля данных. На рис. 2 представлена архитектура системы.

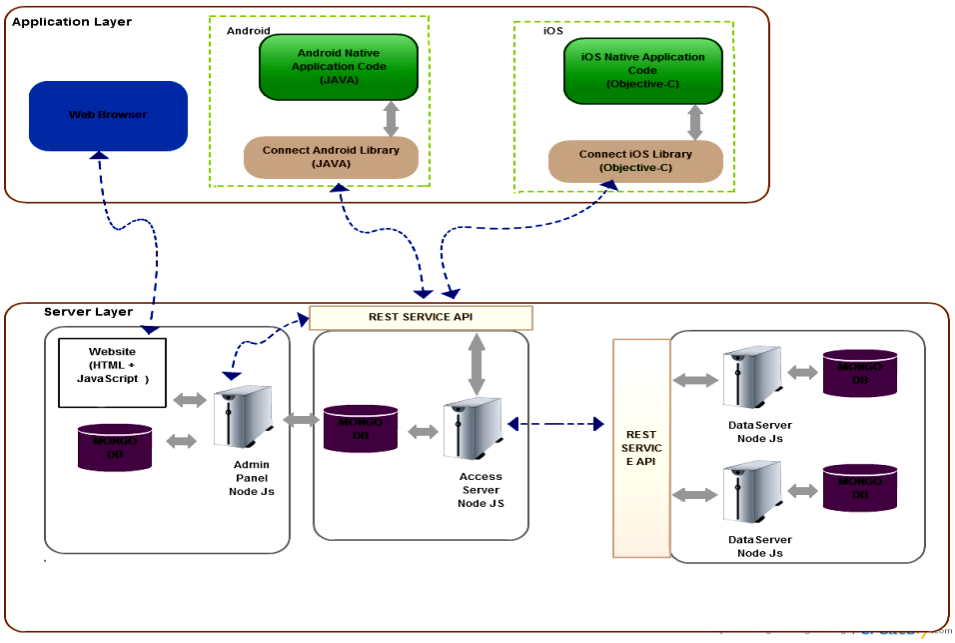

Рис. 2–Архитектура веб-сервиса

Для расширения функционала можно добавить новый модуль. Добавим к нашей архитектуре модуль хранения геоданных. На рис. 3 представлена расширенная архитектура системы.

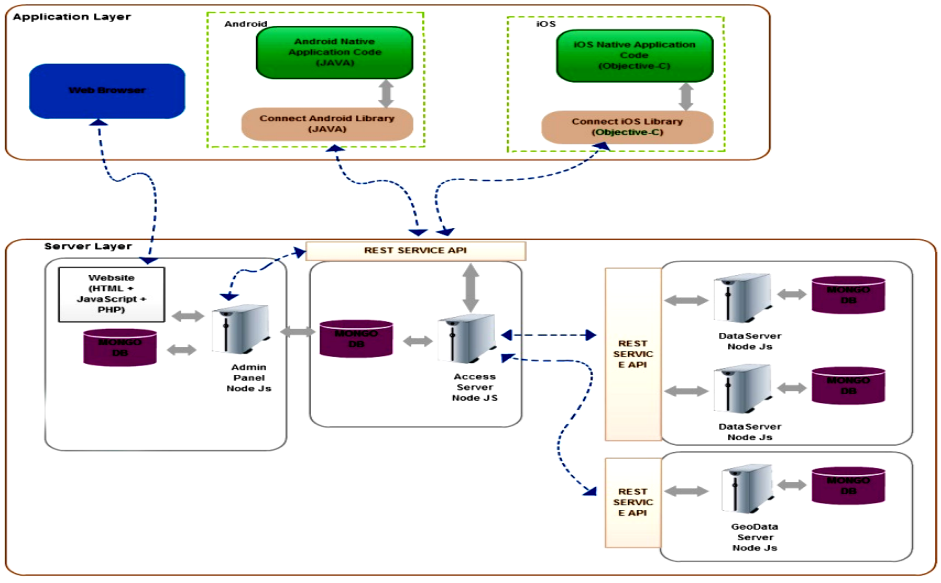

Рис. 3 – Расширенная Архитектура веб-сервиса

Добавление нового функционала не меняет предыдущий, а только расширяет его.Так можно постепенно модернизировать систему, добавляя новые модули, без изменения способа доступа к уже существующему функционалу.

Также интересным решением является добавление в систему Пакетных заданий (Batchoperations). Это уменьшает количество операций на мобильном

клиенте, повышая производительность приложения, передавая список задач на модуль пакетных заданий. На рис. 4 представлен Batch запрос к серверу через curl.

В ответ придёт JSONфайл в котором будут вписаны статусы операции и стандартный ответ как от обычной операции в порядке отправки.

Для добавления обработки Batchопераций нужно добавить модуль обработки пакетных операций. На рис. 5 представлена обновлённая архитектура с Batchмодулем.

Также можно разделить серверы доступа к приложению, если будет сильная нагрузка.

```
curl -X POST \
  -H "API-Key: ${REST_API_KEY}" \
  -H "Content-Type: application/json" \
  -d '{
        "requests": [
          {
            "method": "POST",
            "path": "/data/Score",
            "body": {
              "score": 9000,
              "UserName": "Badanchiiik"
            }
          },
          {
            "method": "PUT",
            "path": "/data/Score/${_id}",
            "body": {
              "score": 9001
            }
          },
          {
            "method": "DELETE",
            "path": "/data/Score/${_id}"
          },
          {
            "method": "GET",
            "path": "/data/Score/"
          }
        ]
      }' \
  https://example.myBaas.home/batch
```

Рис. 4 – образец Batch операции

На каждую отдельную группу выделить отдельный модуль доступа. Например, отдельно на группу модулей данных и отдельно на модуль геоданных.

На случай отсутствия доступа в сеть, можно добавить локальную базу в SDKи сохранять данные для передачи на сервер в неё. Это обеспечит нормальную работоспособность приложения. После получения доступа к серверу, синхронизировать данные. На рис. 6 представлено обновлённое приложение с локальной БД и разделёнными модулями доступа.

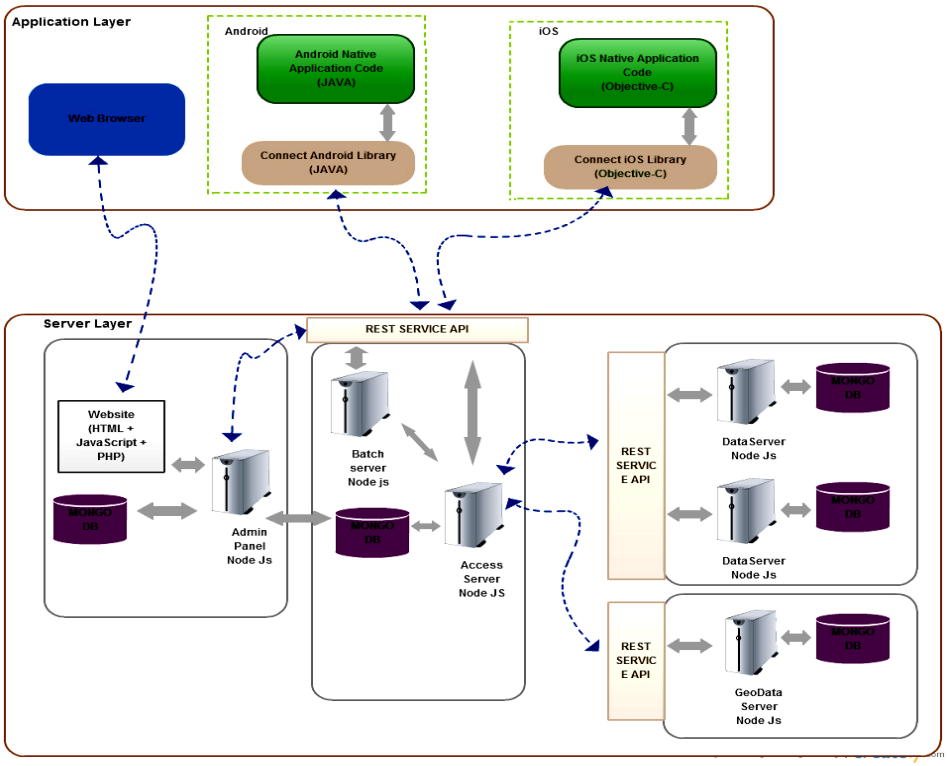

Рис. 5 – Архитектура с Batchмодулем

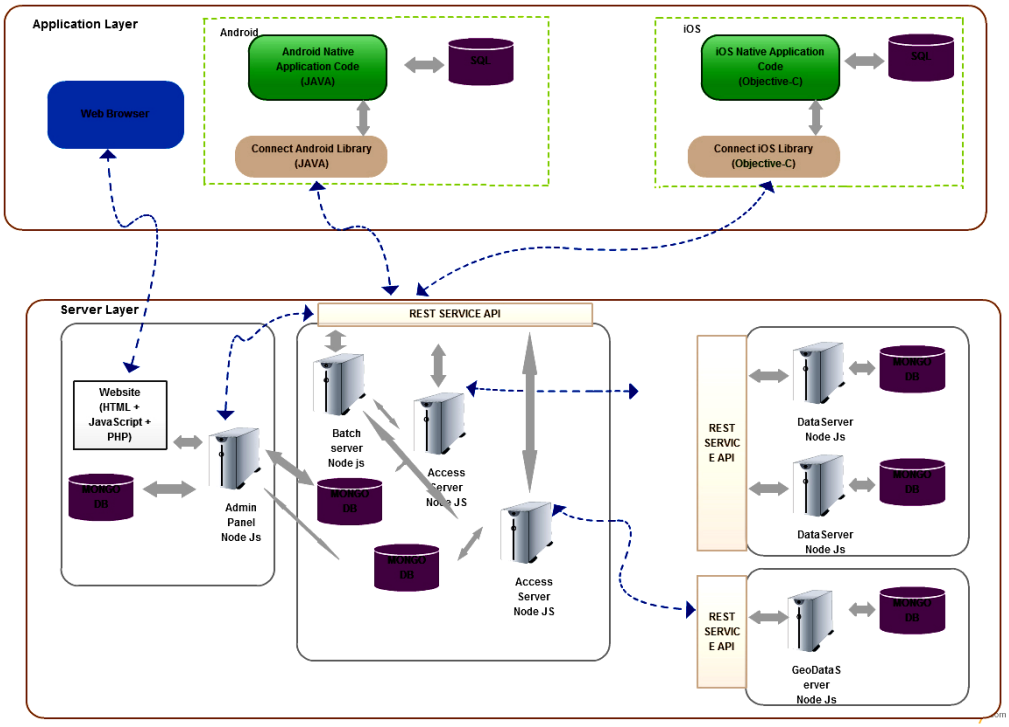

Рис. 6–Архитектура веб-сервиса с локальной БД для SDK

Использование. Для демонстрации возможностей программного продукта рассмотрим пример приложения-органайзера, разработанного на Android

Создадим приложение в системе, назовём его «ExampleTodoApp», и 2 коллекции «User» и «Todo».

Для коллекции пользователей добавим поля логин и пароль, а для коллекции задач нужно добавить поля заголовок, описание, дата окончания и idпользователя.

Приложение будет представлять собой простой интерфейс для добавления задач. Для пользования приложением пользователь должен зарегистрироваться. После прохождения Авторизации пользователь может просматривать свои задачи, добавлять новые, редактировать, удалять.

Для началарассмотрим механизм работы с сервисом на примере нашего приложения. Для работы с сервером будет использовано расширение для GoogleChrome – PostmanRESTClient.

Для работы с сервисом нужно обязательно указывать «Api-Key» в заголовке.

Для того чтобы получить пользователя нужно отправить GET запрос на сервер. На рис. 7 представлен запрос наполучение пользователя с логином Sashka

Рис. 7 – Запрос на получение пользователя с логином «Sashka»

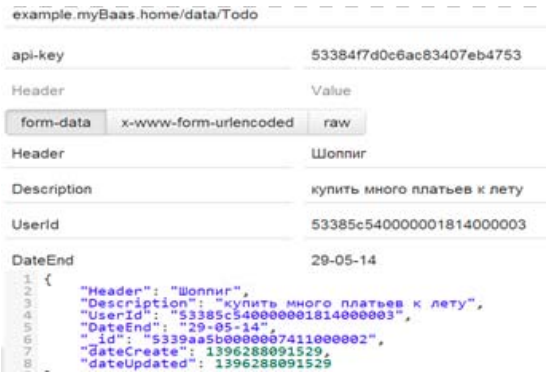

Рис. 8 – Запрос на добавление задачи

Для получения всех задач пользователя «Sashka» нужно отправитьGETзапрос указав userId пользователя.

Для добавления задачи пользователя «Sashka» нужно отправить POST запрос в тело запроса нужно указать значения. На рис. 8 представлен запрос на добавление задачи.

Для редактирования задачи пользователя «Sashka» нужно отправить PUT запросуказав id задачи, а в тело запроса указать поля для изменения. Для удаления задачи пользователя «Sashka» нужно отправить DELETE запрос указав Id заметки.

Используя SDK создали Android приложениедля работы с задачами. На рис. 9 представлен список задач пользователя «Sashka».

Рис. 9 – Список задач пользователя «Sashka»

Выводы. Мобильные BaaSсистемы значительно ускоряют время разработки мобильных приложений за счёт использования готового решения для хранения данных. Данные решения могут быть легко расширены и модифицированы за счёт того что серверная логика инкапсулирована интерфейсом API. На примере стандартного приложения рассмотрен общий алгоритм работы с такими системами.

Список литературы: 1. *Cantelon M.* Node.js in Action [Text] /*Cantelon M. Harter M. Holowaychuk T. J. Rajlich N*// Manning Publications Co. – 2014. – P. 13–40. **2.** *Daniel H.*Node.js for PHP Developers [Text] / *Daniel H.*// O·Reilly Media. –2012. – P. 44–75. **3.** *Taylor R. N.* Software Architecture: Foundations, Theory, and Practice [Text] / *Taylor R. N. Medvidovic N. Dashofy E. M.* // Willey. –2009. – P. 25–75. **4.** http://en.wikipedia.org/wiki/Nodejs - "Node.js". **.**http://en.wikipedia.org/wiki/Service-oriented_architecture – "SOA".

Bibliography (transliterated): 1. *Cantelon M. Harter M. Holowaychuk T.J. Rajlich N* (2014).Node.js in Action.Manning Publications Co, 13-40. **2.** *Daniel H.* (2012). Node.js for PHP Developers.O·Reilly Media, 44-75 . **3.***Taylor R. N. Medvidovic N. Dashofy E. M.* (2009).Software Architecture: Foundations, Theory, and Practice.Willey, 171-183. **4.** http://en.wikipedia.org/wiki/Nodejs - "Node.js". **5.** http://en.wikipedia.org/wiki/Service-oriented_architecture – "SOA".

Поступила (received) 12.03.2014

УДК 665.9

Разработка веб-сервиса для предоставления услуг хранения данных/ Шатовская Т. Б., Марин В. О. // Вісник НТУ «ХПІ». Серія: Нові рішення в сучасних технологіях. – Х: НТУ «ХПІ», – 2014. - № 17 (1060).– С.62-68 . – Бібліогр.: 4 назв. ISSN 2079-5459

В настоящее время развит рынок мобильных технологий, и многие разработчики мобильных приложений сталкиваются с проблемой хранения данных из приложений в сети. В этой работе мы представляем структуру системы для хранения множества данных для различных приложений.

Ключевые слова:сервис, данные, мобильные приложения, хранилища данных, наборы данных.

В даний час розвинутий ринок мобільних технологій, і багато розробників мобільних додатків зустрічаються з проблемою зберігання даних із додатків в мережі. У цій роботі ми представляємо структуру системи для зберігання безлічі даних для різних додатків.

Ключові слова: сервіс, дані, мобільні додатки, сховища даних, набори даних.

Developing web services to provide data storage/ Shatovska T., Marin V. //Bulletin of NTU "KhPI". Series: New desicions of modern technologies. – Kharkov: NTU "KhPI", 2014.-№ 17 (1060).- P.62-68. Bibliogr.:4 . ISSN 2079-5459

Currently developed the market of mobile technology, and many mobile application developers faced with the problem of data storage in the network. In this work we present the structure of a system for storing a plurality of data for various applications.

Keywords: service, data, mobile applications, data warehouses, data sets.

УДК 004.4

К. В. ХАРЧЕНКО, канд. техн. наук, доц., Інститут прикладного системного аналізу, Київ

МЕТОДИ ТА ЗАСОБИ РОЗРОБКИ ПРОГРАМНИХ ДОДАТКІВ ДЛЯ ОПЕРАЦІЙНОЇ СИСТЕМИ АНДРОІД

Пропонується набір сучасних методів та засобів розробки програмних додатків для операційної системи Андроід, які дозволяють ефективно створювати та тестувати програмне забезпечення клієнт-сервер з використанням хмарних систем у якості серверної частини.

Ключові слова:Андроід, TDD, юніт-тести, хмарні системи.

Вступ. Протягом останніх років стався важливий перерозподіл складу апаратних платформ для кінцевих користувачів. Доля ринку мобільних пристроїв у форм-факторі планшетних комп'ютерів уперше перевищила кількість персональних комп'ютерів. Наразі три операційні системи, iOS, Android та Windowsmobile конкурують на ринку. Так, за даними StrategyAnalytics[1] тільки за 3-й квартал 2013 року було активовано 251 млнсмартфонів.

Кількість мобільних додатків у кожній еко-системі для мобільних та планшетних комп'ютерів нелінійно зростає кожного року, перевищив мільйон програм в Андроід [2], 617 тис. програм для iOS, та 156 тис. для WindowsMobile [3].

Особливості розробки програмних додатків для мобільних пристроїв. Процес розробки програмного забезпечення для мобільних пристроїв вимагає підтримувати певні обмеження що до специфіки роботи додатків у мобільних операційних системах. Особливості роботи мобільних додатків стосуються перш за все:

- на витрати живлення акумуляторної батареї мобільного пристрою
- обмеження на кількість даних, що передаються через мережу Інтернет
- зменшену швидкість передачі пакетів в мобільному Інтернет з'єднанні
- можливість втрати пакетів при передачі в мобільному Інтернет з'єднанні
- кількість обмінів даними з зовнішніми пристроями на Bluetooth
- безпеку даних користувача
- значну фрагментацію версій операційних систем та фреймворків
- велику різноманітність розмірів екранів мобільних пристроїв
- обмеження запитів до системи визначення місцезнаходження абонента
- обмеження на розмір файлу додатку