Transaction of Azerbaijan National Academy of Sciences, Series of Physical-Technical and Mathematical Sciences: Informatics and Control Problems, Vol. XXXVIII, No.6, 2018
www.icp.az/2018/6-04.pdf

УДК 004.22, 004.773, 004.62

Э.Р. АЛИЕВ, Д.Б. ГАХ

СОЗДАНИЕ ПРОСТОЙ МОДЕЛИ СТРУКТУРЫ ВЫХОДНЫХ ДАННЫХ СЕРВИСА ПОИСКА ОПТИМАЛЬНОГО ПУТИ

Высокопроизводительные сервисы поиска оптимального пути в транспортных сетях требуют оптимизированных структур данных для вывода информации о найденном пути. Подобные структуры должны иметь как алгоритмическую простоту построения и структурного разбора (парсинга), так и малый размер структур запросов/ответов и понимание человеком. Данная статья рассматривает опыт создания и эксплуатации структур выходных данных ответов для высокопроизводительных сервисов поиска пути в сети автомобильных дорог. Рассматриваются вопросы формата данных, передачи данных по сети интернет и особенности их использования настольными и мобильными клиентскими приложениями.

Ключевые слова: поиск пути, веб-служба, GeoJSON, REST API, мобильные приложения

1. Введение. Создание высокопроизводительной веб-службы поиска пути в транспортной сети подразумевает решение вопросов её интеграции с клиентскими приложениями. В связи с тем, что выходные данные о найденном пути могут содержать богатую информацию, они могут иметь сложную структуру и большой размер. Оптимизация структур выходных данных позволяет ускорить их создание на стороне сервера и структурный разбор (парсинг) на стороне клиента. Сокращение размера выходных данных в свою очередь уменьшает сетевой траффик и позволяет использовать более слабые каналы связи.

Одним из движущих факторов исследований информации в среде мобильных коммуникаций стала повышенная доступность объектов с привязкой к геоинформации, включая базы данных точек интереса (POI) и соглашения для добавления географической информации в документы структурированным способом с использованием языков разметки, таких как GML, KML, GeoRSS и GeoJSON [1, с.71]. XML и JSON являются наиболее известными текстовыми форматами данных, тогда как ProtoBuf и Thrift – относительно новые двоичные форматы сериализации данных [2, с.1].

В данной статье рассматривается создание веб службы для сети автомобильных дорог.

2. Постановка задачи. Создание веб-службы поиска оптимального пути имеет целью предотавление широкому спектру потребителей высокоскоростных функций навигации в транспортных сетях и в частности в сетях автомобильных дорог. Указанные условия налагают на формат выходных данных два основных требования – скорость обработки данных и популярность используемого формата серилизации данных. Формат серилизации данных должен обеспечивать работу клиентсвих приложений на базе веб-браузера, нативных мобильных приложений и прочих приложений, использующих популярные компоненты для обработки данных.

Модель дорожной сети. Модель дорожной сети представлена как двунаправленный конечный граф G (N, E) в котором дороги представлены рёбрами E, а пересечения и соединения дорог узлами N. Элементы графа включают также аттрибутивную информацию, требуемую для алгоритма поиска пути – длины, скорости, направленность движения, и т.п.

Элементы выходной информации о пути. Найденный путь является под-графом графа G и представляет собой ломанную линию, состоящую из прямых отрезков, а также вершин и узлов между ними.

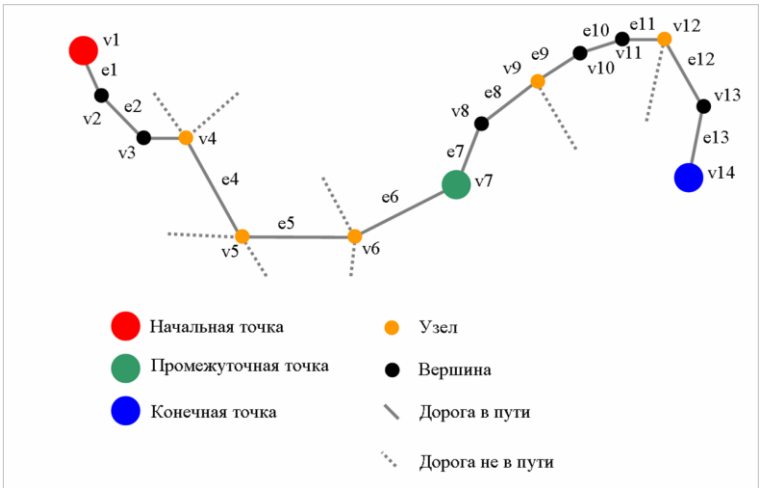


Рис.1. Схема участка найденного пути

3. Формат выходных данных. Два самых распространенных современных формата сериализации данных – это расширяемый язык разметки (Xtensible Markup Language - XML) и нотация объектов JavaScript (JavaScript Object Notation - JSON) [2, с.2]. Хотя формат XML в сравнении с форматом JSON является зрелым и поддерживается множеством языков программирования, он обладает рядом недостатков. Так формат XML существенно уступает формате JSON по времени кодирования/декодирования, а также по размеру файла [2, с.4-5]:

	XML	JSON
Книга	22.842	4.177
Видео	17.884	4.097

Среднее время кодирования (серилизации), мс.

	XML	JSON
Книга	7.908	1.199
Видео	6.742	0.755

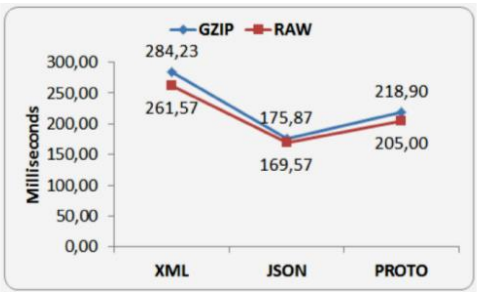
Среднее время декодирования (десерилизации), мс.

	XML	JSON
Книга	873	781
Видео	231	139

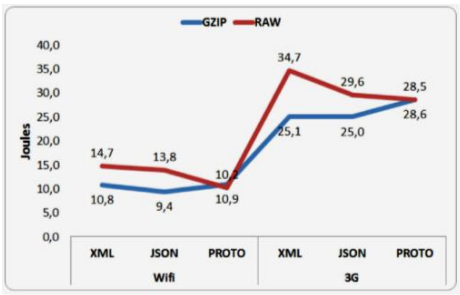
Размер, байт

Для сравнения размеров отчётов службы поиска пути в форматах GeoJSON и XML были сгенерированы несколько файлов, содержащих одинаковые данные. Для чистоты эксперимента из файлов были удалены непечатаемые символы, такие как пробелы, символы табуляции и перевода каретки. В общей сложности средний размер файла XML оказался в процентном соотношении к среднему размеру файла GeoJSON равен 201%, что говорит о более чем 2х кратном увеличении размера при использовании формата XML вместо GeoJSON. Сжатие ZIP позволило сократить разницу в размерах файлов до 115%, что свидетельствует о том, что сжатые GeoJSON данные так же имеют преимущества в размере по сравнению с XML.

4. Мобильные приложения. Формат выходной информации службы поиска пути существенно влияет на работу клиентских мобильных приложений. При разработке мобильных приложений управление потреблением батареи требует особого внимания [3, с.1].



Время декодирования данных [3, с.3]



Потребление энергии на декодирования данных [3, с.5]

Как видно из диаграмм формат JSON на мобильных платформах имеет преимущества в сравнении с форматом XML как по производительности, так и по уровню энергопотребления.

5. REST API. В настоящее время REST (Representational State Transfer / репрезентативный перенос состояний) стал наиболее часто используемым способом создания, публикации и использования веб-служб, используя в качестве формата обмена данными JSON [4, с.102]. Однако, до сих пор нет официальных стандартов для REST (таких как например WSDL для веб-служб) и для JSON (таких как например, XML-схема для XML) [4, с.102]. Использование формата JSON и разработка API (Application Programming Interface - интерфейса прикладного программирования) на основе REST рассматривается как наилучшее решение для создания высокопроизводительных веб служб.

6. Вывод данных о пути в формате GeoJSON. Как было показано формат JSON имеет преимущества перед форматом XML. Следует так же учесть, что формат JSON произошёл из языка JavaScript и среды веб программирования.

Формат GeoJSON основан на формате JSON и имеет все его преимущества [5]. Так как описанная задача требует вывод географических данных, формат GeoJSON наиболее удобен для использования.

Формат GeoJSON содержит описание географической фигуры и её аттрибутов [6, с.1-28]. В целях обеспечения расширяемости корневой документ GeoJSON должен быть типа "FeatureCollection", то есть набор гео-объектов. Эта коллекция содержит один гео-объект – найденный путь. Однако при необходимости в следующих версиях можно добавить другие гео-объекты сохраняя совместимость с предыдущими версиями.

Корневой документ GeoJSON:

```
{
  "type": "FeatureCollection",
  "features": [ Гео-объект ]
}
```

Гео-объект. Описание гео-объекта в формате GeoJSON включает указание типа документа ("Feature"), описание геометрии и свойств гео-объектов.

Описание геообъекта:

```
{
  "type": "Feature",
  "geometry": { Описание геометрии },
  "properties": { Свойства гео-объекта }
}
```

Описание геометрии. Ввиду того, что найденный путь представляет собой ломанную линию, описание геометрии включает последовательность географических координат концов и вершин линий. Тип геометрии - "LineString".

Описание геометрии:

```
{
    "type": "LineString",
    "coordinates": [
      [ 49.87488, 40.38096 ],
      [ 49.87342, 40.37972 ],
...
      [ 49.79757, 40.39738 ],
```

```
    [ 49.79763, 40.39795 ]
   ]
}
```

Описание свойств. Свойства найденного пути представлены двумя массивами - "points" и "roads". Описание этих двух массивов преставлено ниже. Слеует учесть, что размеры наименований ключей сокращены до минимума. Большинство ключей описания свойств пути представлены одиночными символами. Подобное сокращение в итоге сокращает размер всего выходного файла.

Описание свойств пути:

```
{
      "points": [Описание точек],
      "roads": [Описание дорог]
}
```

Описание точек. Описание точек представляет собой массив атрибутов ключевых точек, то есть точек, представляющих собой подмножество узлов графа G.

Описание точки:

```
{
   "v" : 0,
   "t" : 1,
   "r" : 48808,
   "a" : -12
}
```

В приведённом описании использованы следующие ключи:

- "v" – ссылка на координаты угла, приведённые в секции "coordinates";
- "t" – тип точки (узла):
  - 1 – начальная точка;
  - 2 – промежуточная точка (точка, в которой завершается один сегмент пути и начинается другой);
  - 9 – завершающая точка;
- "r" – идентификатор дороги, которая начинается на данной точке;
- "a" – угол поворота с предыдущей дороги на ту, которая начинается на данной точке.

В целях уменьшения размера файла тип точки кодируется целыми числами, что исключает необходимость наличия кавычек. Для внутренних частей сегментов дорог тип точки принимается равным 2 и не указывается. Так как подобных точек большинство, подобное исключение существенно уменьшает размер выходного файла. Параметр угла поворота для значений угла менее 10 градусов так же не приводится и путь на таких участках считается прямым. Отрицательное значение указывает на поворот налево, положительное – поворот направо.

Описание дорог. Описание дорог содержит различную информацию об участке дороги. Минимальная информация содержит идентификатор участка дороги "id" и наименование "n". При этом наименование может содержать строки текста на различных языках. Данная особенность указания аттрибутов дороги обладает большой расширяемостью.

Описание дороги:


```
{
   "id": 117463,
   "n": [
   {
      "l": "en",
      "n": "Mammadsharif Hamidov"
    }
    ]
}
```


Полный выходной файл GeoJSON. Суммируя описание формата ниже приведён пример полного выходного файла. Так как основная задача приведения примера – показать структуру выхоных данных, часть данных с одинаковой структурой в массивах заменена троеточием.

Полный файл выходной информации GeoJSON:


```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "geometry": {
        "type": "LineString",
        "coordinates": [
          [ 49.87488, 40.38096 ],  [ 49.87342, 40.37972 ], ..., [ 49.79757, 40.39738 ],
          [ 49.79763, 40.39795 ]
        ]
      },
      "properties": {
        "points": [
          {
            "v": 0,
            "t": 1,
            "r": 109971
          },
          {
            "v": 2,
            "r": 108853,
            "a": 129
          },
...
          {
            "v": 166,
            "r": 117463,
            "a": 158
          },
          {
            "v": 169,
```

```
          "t": 9,
          "r": 118377
        }
      ],
      "roads": [
        {
          "id": 109971,
          "n": [
            {
              "l": "en",
              "n": "Sabit Orujov"
            }
          ]
        },
        {
          "id": 108853,
          "n": [
            {
              "l": "en",
              "n": "Afiyaddin Jalilov"
            }
          ]
        },
...
        {
          "id": 117463,
          "n": [
            {
              "l": "en",
              "n": "Mammadsharif Hamidov"
            }
          ]
        },
        {
          "id": 118377,
          "n": [
            {
              "l": "en",
              "n": "Zaur Karimov"
            }
          ]
        }
      ]
    }
  ]
}
```

7. Практическая реализация. Описанная структура выходных данных была реализована в веб сервисе поиска пути PFSRV, разработанного в рамках проекта www.GoMap.Az [7]. Веб сервис разработан на языке C++ и реализован как служба MS Windows. Для разработки использовалась среда MS Visual Studio. Основные коды написаны с учётом много-платформенной совместимости и возможности запуска на других операционных системах, таких как Linux и Unix. Для разработки использовались многоплатформенные библиотеки C++ Boost [8]. Были разработаны и тестированы два клиентских приложения, потребляющих службы PFSRV.

8. Выводы. Полученные результаты могут быть охарактеризованы следующим образом:

- Разработанные структуры и формат данных показали свою эффективность с точки зрения производительности и межплатформенной совместимости;
- Реальный опыт позволил разработать и проверить на практике методику создания и использования описываемых структур данных;
- Описываемую методику создания и оценки можно применить на практике так же для веб службы реестра городских адресов;
- Для экономии сетевого трафика возможно сжатие при помощи стандарта GZIP;
- Следует рассмотреть реализацию веб служб в соответствии с технологией REST.

Практическая ценность работы заключается в том, что был достигнут высокий уровень производительности, межплатформенного взаимодействия, масштабируемости, расширяемости, а также лучшая энерго-эффективность процедур, отвечающих за трансформацию и передачу данных. Описанные в данной статье методики могут быть применены и при проектировании других веб-служб, реализующих функционал электронных онлайн карт.

Литература

1. D. Mountain, The dimensions of context and its role in mobile information retrieval, Department of Information Science, City University London, UK, SIGSPATIAL Special: Volume 3 Issue 2, July 2011, c.71-77.
2. A. Sumaray, S. Kami Makki, A Comparison of Data Serialization Formats For Optimal Efficiency on a Mobile Platform, ICUIMC '12 Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication, Article No. 48, 2012, c.1-6.
3. B. Gil, P. Trezentos, Impacts of data interchange formats on energy consumption and performance in Smartphones., OSDOC '11 Proceedings of the 2011 Workshop on Open Source and Design of Communication, 2011, c.1-6.
4. M. Polák, I. Holubová, REST API Management and Evolution Using MDA., C3S2E '15 Proceedings of the Eighth International C* Conference on Computer Science & Software Engineering, 2015, c.1-8.
5. http://geojson.org, доступ 5 октября, 2017.
6. H. Butler, M. Daly, A. Doyle, S. Gillies, S. Hagen, T. Schaub, The GeoJSON Format, Request for Comments, Internet Engineering Task Force (IETF), ISSN: 2070-1721, https://tools.ietf.org/html/rfc7946, доступ 5 октября, 2017
7. http://www.gomap.az Project, доступ 22 января, 2018.
8. Boost C++ Libraries, http://www.boost.org, доступ 5 октября, 2017.

UOT 004.22, 004.773, 004.62

E.R. Əliyev, D.B. Qax

Optimal yolun axtarılması xidmətinin çıxış məlumatlarının sadə struktur modelinin yaradılması

Nəqliyyat şəbəkələrində optimal yolun effektiv axtarış xidmətləri tapılan yol haqqında məlumatların əks olunması üçün optimallaşdırılmış məlumat strukturlarının yaradılması tələb olunur. Belə strukturlar həm sadə alqoritmik quruluşa və struktur sintaksis analizə (parsing), həm də insanın qavraması üçün kiçik ölçülü sorğular / cavablar strukturlarına malik olmalıdır. Təqdim olunan məqalədə avtomobil yolları şəbəkəsində effektiv yol axtarış xidmətləri üçün cavabların əks olunması strukturlarının yaradılması və istismarı təcrübəsi araşdırılır. Bundan başqa,

məqalədə məlumatların formatlaşdırılması məsələləri, onların İnternet şəbəkəsi üzərindən ötürülməsi, stolüstü və mobil müştəri proqram təminatları tərəfindən bu məlumatlardan istifadə olunması xüsusiyyətləri nəzərdən keçirilir.
Açar sözlər: yolun axtarışı, veb servislər, GeoJSON, REST API, mobil proqramlar

E.R. Aliyev, D.B. Gakh

Creating a simple model for output data structure for optimal path finder service

High-performance web services providing functionality of optimal path finding in transportation networks require optimized output data structures to represent information about the found path. Such structures should have algorithmic simplicity of construction and parsing, small size of request/response data, and readability by human. This article describes the experience of creating and operating of output structures for high-performance path finding web services in the road network. Issues relating to the data format, data transmission by the Internet channels, and the specifics of their processing by desktop and mobile client applications are considered.
Keywords Path find, web services, GeoJSON, REST API, mobile applications

Институт Систем Управления НАН Азербайджана Представлено 18.04.2018