МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
"Национальный исследовательский ядерный университет "МИФИ"

ФАКУЛЬТЕТ КИБЕРНЕТИКИ

КАФЕДРА ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему:

Разработка модуля преобразования геоданных из формата OpenStreetMap в формат OpenGIS

Группа К7-361

Студент _____ (Лаврентьева М.О.)

Руководитель _____ (Муравьёв С.К.)

Оценка _____

Члены комиссии _____

_____

_____

_____

Москва 2011

Литература

Содержание

1 Введение

В настоящее время широкое распространение в информационных технологиях получили геоинформационные системы (ГИС). ГИС представляет собой аппаратно-программный человеко-машинный комплекс, обеспечивающий сбор, обработку, отображение и распространение геоданных.

Понятие геоданные (пространственные данные, географические данные) включает в себя цифровые данные о пространственных объектах, сведения об их местоположении и свойствах, пространственных и непространственных атрибутах.

ГИС позволяет наиболее эффективно и комплексно использовать совершенно различные геоданные при решении научных и прикладных географических задач, связанных с инвентаризацией, анализом, моделированием, прогнозированием и управлением окружающей средой и территориальной организацией общества. Термин ГИС используется в более узком смысле — как инструмент (программный продукт), позволяющий пользователям искать, анализировать и редактировать цифровые карты, а также получать различную дополнительную информацию. Наиболее известные и часто используемые современные ГИС - это ArcGIS, MapInfo, а также Google Maps, Yandex-карты.

Данная учебно-исследовательская работа посвящена рассмотрению задачи сбора, обмена и хранения геоданных.

Картографические данные будут получены из открытого источника в интернете OpenStreetMap.org в формате xml.

Однако обмен и хранение пространственных данных требует развитой и всеобъемлющей системы стандартов представления геоданных. Такой системой стандартов для многих приложений на данный момент является формат Open Geospatial Consortium (OGC) (или сокращенно OpenGIS). Таким образом, для дальнейшего анализа и представления полученных геоданных их необходимо представить в данном формате (в формате OpenGIS).

Итак, цель работы: описание алгоритма работы и реализация функционала модуля преобразования геоданных из формата OpenStreetMap в формат OpenGIS.

2 Описание алгоритма работы модуля

2.1 Описание логики алгоритма

Алгоритм состоит из следующих этапов:

1. Ввод имени XML-файла для извлечения необходимой информации.
2. Формирование XPath-запроса для извлечения геоданных об объектах необходимого типа.
3. Формирование результата XPath-запроса, вывод извлеченных геоданных.
4. Формирование SQL-запроса к базе данных для сохранения извлеченных геоданных в формате OpenGIS.
5. Формирование отчёта о занесении геоданных в базу данных.

Ввод имени XML-файла для извлечения необходимой информации.

Выбор XML-файла осуществляется пользователем из заранее заданного набора файлов.

Формирование XPath-запроса для извлечения геоданных об объектах необходимого типа.

Формирование Xpath-запроса происходит непосредственно пользователем в соответствующем окне программы.

Формирование результата XPath-запроса, вывод извлеченных геоданных.

Результаты XPath-запроса выводятся на экран в соответствующем окне программы.

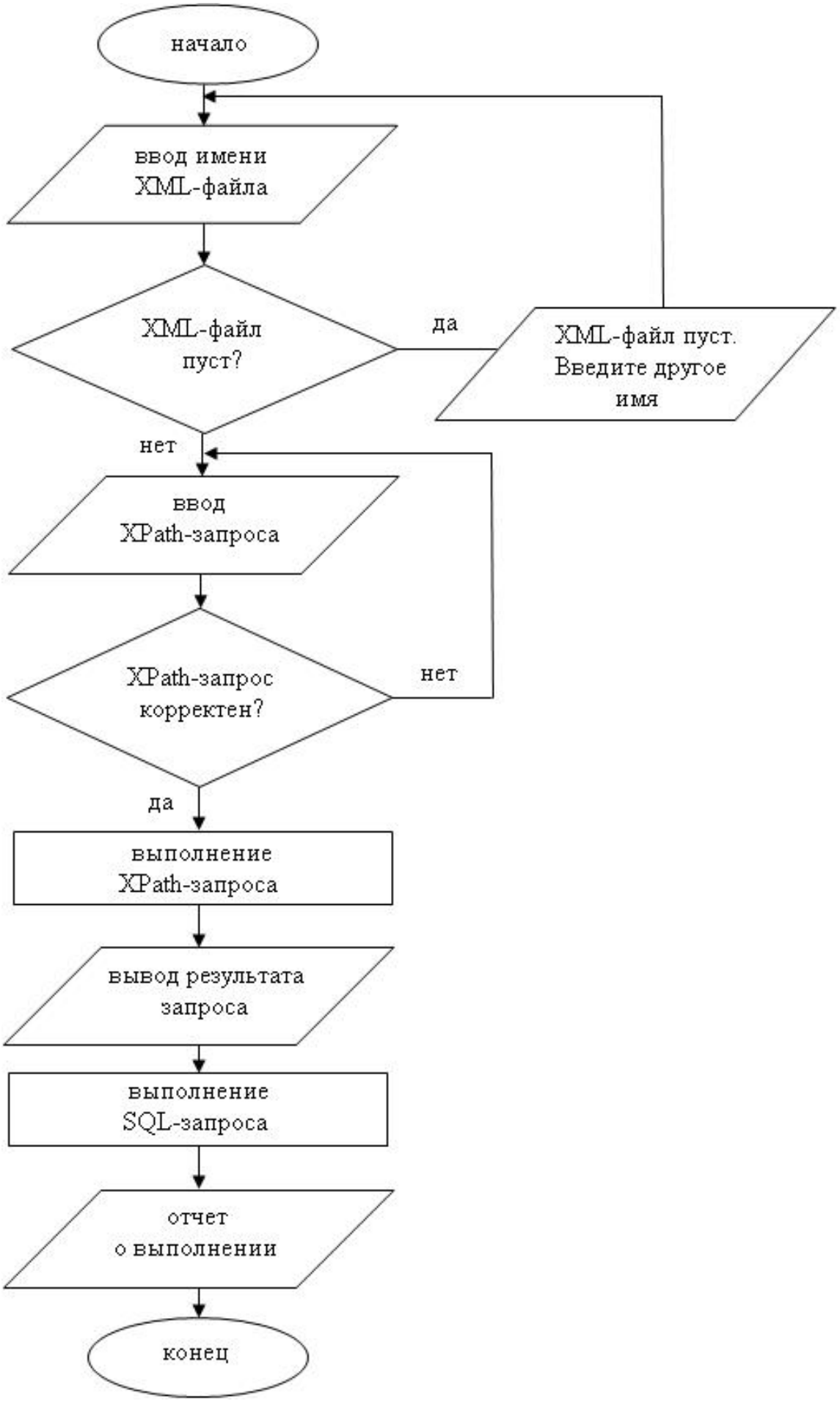
Формирование SQL-запроса к базе данных для сохранения извлеченных геоданных.

SQL-запрос формируется программистом до запуска модуля. Формируется без участия пользователя.

Формирование отчёта о занесении геоданных в базу данных.

Отчёт о занесении геоданных в БД выводится на экран в соответствующем окне программы.

2.2 Блок-схема алгоритма



3 Теоретическая часть

3.1 Язык XML

3.1.1 Общие сведения

Язык разметки (в компьютерной терминологии) - набор символов или последовательностей, вставляемых в текст для передачи информации о его выводе или строении. Принадлежит классу компьютерных языков, но не является языком программирования. Текстовый документ, написанный с использованием языка разметки, содержит не только сам текст (как последовательность слов и знаков препинания), но и дополнительную информацию о различных его участках (например, указание на заголовки, выделения, списки и т.д.). Иногда язык разметки позволяет вставлять в документ интерактивные элементы и содержание других документов. Языки разметки используются везде, где требуется вывод форматированного текста - в типографии (SGML, TeX, PostScript, PDF), пользовательских интерфейсах компьютеров (MicrosoftWord, OpenOffice), Всемирной Сети (HTML, XHTML, XML, WML, VML, PGML, SVG, XBRL).

Общая особенность всех языков разметки состоит в том, что они перемешивают текст документа с инструкциями разметки в потоке данных или файле. Это не необходимость, разметку можно изолировать от текста, используя указатели, метки, идентификаторы или другие методы координации. Такая "отделенная разметка"характерна для внутреннего представления программ, работающих с размеченными документами.

Родоначальником наиболее известных в наше время языков разметки является обобщенный структурированный язык разметки (Structured Generalized Language) - SGML, определенный в международном стандарте ISO 8879:1986. Языки HTML(HyperText Markup Language - язык разметки гипертекста) и XML (eXtensible Markup Language - расширяемый язык разметки) различными способами образованы из SGML.

SGML определяет базовый синтаксис, но дает возможность создавать собственные элементы (отсюда термин "обобщенный"в названии языка). Чтобы использовать SGML для описания определенного документа, необходимо продумать соответствующий набор элементов и структуру документа.

Основные части документа SGML:

- SGML-декларация. Определяет, какие символы и ограничители могут появляться в приложении;
- Document Type Definition. Определяет синтаксис конструкций разметки. DTD может включать дополнительные определения, такие, как символьные ссылки-мнемоники.
- Спецификация семантики, относится к разметке. Даёт ограничения синтаксиса, которые не могут быть выражены внутри DTD;

- Содержимое SGML. В содержимом SGML-документа по крайней мере должен быть корневой элемент.

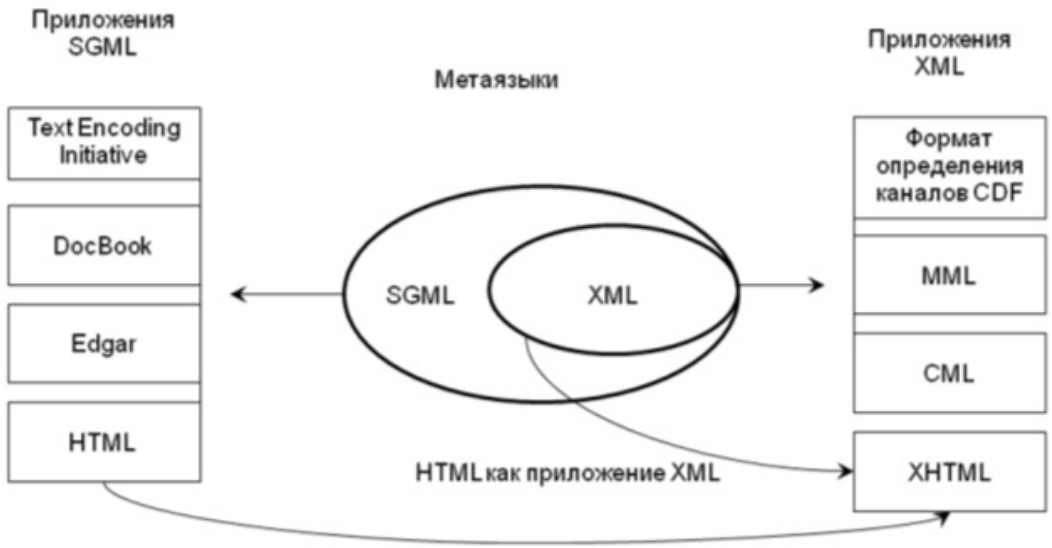
В 1996 г. группа XML Working Group консорциума World Wide Web Consortium (W3C) разработала ветвь языка SGML, специально приспособленную для размещения информации в World Wide Web (аналогично стандартному для Web языку гипертекстовой разметки HTML (Hypertext Markup Language)). "XML предназначен для облегчения использования языка SGML в Web и выполнения задач, которые в настоящее время реализуются с помощью языка HTML. XML разработан с целью усовершенствовать применение и взаимодействие языков SGML и HTML."(спецификация версии 1.0 XML)

Как и SGML, XML дает возможность разрабатывать собственные наборы элементов при описании определенного документа и присваивать им любые имена по выбору (поэтому XML - расширяемый язык). Как и в SGML, в теле программы может быть определено XML-приложение (или словарь), которое содержит набор наиболее употребительных элементов общего назначения и структуру документа, которая может быть использована для описания документа определенного типа.

Таким образом, XML является подмножеством SGML, разработанным для упрощения процесса машинного разбора документа.

Кроме того, XML - это текстовый формат, предназначенный для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки (например, XHTML), иногда называемых словарями, то есть XML можно использовать для описания практически любого документа, от музыкальной партитуры до баз данных.

Взаимосвязь между SGML, XML, HTML и некоторыми другими языками показана на следующей диаграмме:



3.1.2 Структура XML-документа

Основные синтаксические правила построения XML документов:

1. XML документ содержит один и только один корневой элемент, содержащий все остальные элементы.
2. Дочерние элементы, содержащиеся в корневом элементе, должны быть правильно вложены.
3. Имена элементов подчиняются правилам:
   - Имя начинается с буквы, знака подчеркивания или двоеточия.
   - После первого символа в имени могут быть буквы, цифры, знаки переноса, подчеркивания, точка или двоеточие.
   - Имена не могут начинаться с буквосочетания XML.

XML документ имеет следующую структуру:

- Первая строка XML документа называется объявлением XML. Это необязательная строка, указывающая версию стандарта XML (обычно это 1.0). Также здесь может быть указана кодировка символов и внешние зависимости.
- Комментарий может быть размещен в любом месте дерева. XML комментарии размещаются внутри пары тегов <!– и заканчиваются –>. Два знака дефис (–) не могут быть применены ни в какой части внутри комментария.
- Остальная часть этого XML -документа состоит из вложенных элементов, некоторые из которых имеют атрибуты и содержимое.
- Элемент обычно состоит из открывающего и закрывающего тегов, обрамляющих текст и другие элементы.
- Открывающий тег состоит из имени элемента в угловых скобках; закрывающий тег состоит из того же имени в угловых скобках, но перед именем ещё добавляется косая черта.
- Содержимым элемента называется всё, что расположено между открывающим и закрывающим тегами, включая текст и другие (вложенные) элементы.
- Кроме содержания у элемента могут быть атрибуты - пары имя=значение, добавляемые внутрь открывающего тега после названия элемента.
- Значения атрибутов всегда заключаются в кавычки (одинарные или двойные), одно и то же имя атрибута не может встречаться дважды в одном элементе.

- Не рекомендуется использовать разные типы кавычек для значений атрибутов одного тега.

- Для обозначения элемента без содержания, называемого пустым элементом, необходимо применять особую форму записи, состоящую из одного тега, в котором после имени элемента ставится косая черта "/".

Описанные выше правила позволяют контролировать только формальную правильность XML документа, но не содержательную. Для решения второй задачи используются так называемые схемы.

Схема четко определяет имя и структуру корневого элемента, включая спецификацию всех его дочерних элементов. Программист может задать, какие элементы и в каком количестве обязательны, а какие - необязательны. Схема также определяет, какие элементы содержат атрибуты, допустимые значения этих атрибутов, а также значения по умолчанию.

Чаще всего для описания схемы используются следующие спецификации:

- DTD (Document Type Definition) - язык определения типа документов, который первоначально использовался в качестве язык описания структуры SGML -документа.

- XDR (XML Data Reduced) - диалект XML, разработанный Microsoft.

- XSD (язык определения схем XML) - рекомендация консорциумом W3C с 2001 года.

3.2 OpenStreetMap

3.2.1 Описание OpenStreetMap.org

OpenStreetMap ("открытая карта улиц"), сокращённо OSM, - некоммерческий сетевой картографический проект, в котором могут участвовать все желающие пользователи Интернета. Он направлен на создание подробной свободной и бесплатной географической карты всего мира. Все данные доступны для легального копирования, редактирования и коммерческого использования. База данных OSM содержит данные самого разного рода, например, дороги, тропы, здания, магазины, аптеки, памятники, деревья, заборы, детские площадки, точки Wi-Fi, почтовые ящики, телефонные будки, административное деление, адреса, этажность зданий, часы работы, веб-сайты, телефоны и многое другое.

OpenStreetMap остаётся единственным картографическим проектом, который является свободным проектом. Основными конкурентами являются сайты Google Map Maker, Wikimapia, Яндекс.Народная карта, но все эти проекты имеют общий недостаток - они являются "закрытыми"и несвободными.

Данные об основных дорогах обычно получаются из истории "треков". Они записаны GPS-приёмниками или GPS-трекерами. Такие треки создаются добровольцами и выполняются в результате путешествий по исследуемому району пешком, на велосипеде или на автомобиле. Затем треки экспортируются из GPS-устройства в специальную программу.

Спутниковые снимки земной поверхности позволяют рисовать, не имея треков, карты крупных городов (для которых имеются снимки высокого разрешения). В качестве источников используются правительственные сервисы, такие как Landsat, Prototype Global Shorelines (PGS) и TIGER, а также картографические сервисы Yahoo!, Bing Maps и Космоснимки.

3.2.2 Особенности XML-документа OpenStreetMap

OpenStreetMap использует топологическую структуру данных, состоящую из объектов:

- node (точка) - точка с указанными координатами;

- way (линия) - упорядоченный список точек, составляющих линию или полигон (замкнутая линия);

- relation (отношение) - группы точек, линий и других отношений, которым назначаются некоторые свойства;

- tag (тег) - пары "ключ - значение могут назначаться точкам, линиям и отношениям.

Информацию с OSM можно получить и в интересующем нас формате XML. Рассмотрим структуру XML-документа на примере документа OpenSreetMap (OSM).

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="CGImap 0.0.2">
 <bounds minlat="54.0889580" minlon="12.2487570" maxlat="54.0913900" maxlon="12.2524800"/>
 <node id="298884269" lat="54.0901746" lon="12.2482632" user="SvenHRO" uid="46882" visible="true" version="1" changeset="676636" timestamp="2008-09-21T21:37:45Z"/>
 <node id="261728686" lat="54.0906309" lon="12.2441924" user="PikoWinter" uid="36744" visible="true" version="1" changeset="323878" timestamp="2008-05-03T13:39:23Z"/>
 ...
 <node id="298884272" lat="54.0901447" lon="12.2516513" user="SvenHRO" uid="46882" visible="true" version="1" changeset="676636" timestamp="2008-09-21T21:37:45Z"/>
 <way id="26659127" user="Masch" uid="55988" visible="true" version="5" changeset="4142606" timestamp="2010-03-16T11:47:08Z">
  <nd ref="292403538"/>
  <nd ref="298884289"/>
  ...
  <nd ref="261728686"/>
  <tag k="highway" v="unclassified"/>
  <tag k="name" v="Pastower Straße"/>
 </way>
 <relation id="56688" user="kmvar" uid="56190" visible="true" version="28" changeset="6947637" timestamp="2011-01-12T14:23:49Z">
  <member type="node" ref="294942404" role=""/>
  ...
  <member type="node" ref="364933006" role=""/>
  <member type="way" ref="4579143" role=""/>
  ...
  <member type="node" ref="249673494" role=""/>
  <tag k="name" v="Küstenbus Linie 123"/>
  <tag k="network" v="VVW"/>
  <tag k="operator" v="Regionalverkehr Küste"/>
  <tag k="ref" v="123"/>
  <tag k="route" v="bus"/>
  <tag k="type" v="route"/>
 </relation>
 ...
</osm>
```

XML-документ OSM состоит из двух основных частей: пролога и тела документа.

Пролог предваряет любой документ, может содержать идентификационную информацию о XML-документе (спецификация требует ее включения), а может быть абсолютно пустым.

В OSM в пролог входит объявление XML, указывающее на то, что это XML-документ и содержащее номер версии (version) и используемую кодировку (encoding declaration).

<?xml version="1.0"encoding="UTF-8»

Пролог также может содержать комментарии и другие инструкции по обработке, которые сообщают XML-процессору, как обрабатывать конструкции, находящиеся между <? и ?>.

Второй основной частью XML-документа является тело документа (Документ, корневой элемент), который в свою очередь содержит дополнительные элементы. В XML-документе элементы определяют его логическую структуру и несут в себе информацию, содержащуюся в документе (в нашем случае это информация о геообъектах, такая как тип геообъекта, его название, координаты и т.д.). Типовой элемент состоит из начального тега, содержимого элемента и конечного тега. Содержимым элемента могут быть символьные

данные, другие (вложенные) элементы, либо сочетание данных и вложенных элементов.

В нашем случае элемент Документ – osm. Его начальный тег – <osm>, конечный тег – </osm>, а содержимое – вложенные элементы bounds (границы рассматриваемой области с атрибутами minlat, maxlat, minlon, maxlon), node, way, relation. Корневой элемент также имеет атрибуты - versoin (версия Application Programming Interface, для возможности использования во внешних программных продуктах) и generator (программа, с помощью которой был создан этот файл).

В OpenStreetMap нет ограничений на теги, которыми могут быть помечены элементы OSM. Но все же существует рекомендованный набор объектов и соответсвующих им тегов для структурированной работы.

1. Базовые типы элементов: точка (node), линия(way) и полигон(area).

   - точка (node) - базовый элемент в структуре данных OSM. Точка имеет параметры "широта"и "долгота". Точки используются для того, чтобы определить "линию"(см. ниже), однако точка может являться и самостоятельным элементом карты, и использоваться для обозначения отдельного ни с чем не связанного объекта (например, телефонной будки, кафе, для указания координат, к которым привязано название населённого пункта или любого интересного места (POI — point of interest). Отдельные точки (т.е. не входящие в состав линий или областей) всегда должны иметь хотя бы одно свойство, например amenity=parking (объект инфраструктуры - парковка).

     Точки, входящие в состав линии, часто не имеют свойств и нужны только для описания линии; однако это не является незыблемым правилом. Например, точка, входящая в состав линии со свойством railway=rail (т. е. линии, обозначающей полотно железной дороги) и обозначающий остановочный пункт железной дороги имеет свойство railway=station.

   - линия (way). Линия представляет собой ломаную линию, проходящую через точки. Линия состоит, как минимум, из двух точек. Обычно линиями обозначаются улицы, дороги и т.д. Одна точка может принадлежать нескольким линиям одновременно.

     Линия характеризуется свойствами, которые распространяются на неё на всём протяжении. Например, для линии, обозначающей дорогу, такими свойствами могут являться тип и качество покрытия, допустимая скорость движения и т.п. Если при уточнении выясняется, что не все свойства линии сохраняются на всём её протяжении (например, на дороге, которой соответствует линия, имеется участок с другим типом покрытия), то линия может быть разделена на части.

Для того, чтобы считаться корректно определённой, линия должна иметь хотя бы одно свойство.

- полигон (area) или замкнутая линия (closed way) - элемент карты, предназначенный для описания участков поверхности. Полигон формируется замкнутой линией (т.е. первая точка линии совпадает с последней) и является совокупностью этой самой линии и области, находящейся внутри контура этой линии. В этом смысле полигон не является самостоятельным типом элементов, а лишь псевдоэлементом, особой разновидностью линии с соответствующими свойствами.

  Полигоны используются для обозначения участка поверхности, обладающего общими свойствами. Например, полигоны используются для описания водоёмов и лесов.

  Корректно определённый полигон должен иметь хотя бы одно свойство.

  Для описания вырезов, «дыр» в полигонах, например, для описания участка занятого лесом, внутри которого имеется вырубленный участок рисуются мультиполигоны.

2. Тип элементов - отношение (relation). Отношение группирует объекты по определенному признаку и для определенной цели. 'Участниками' (member) отношения могут любые объекты (точки, линии, области) и даже другие отношения. Эти элементы — участники отношения и каждому участнику присваивается 'роль' (role) в отношении. Как и другие типы элементов могут иметь теги. Один объект может входить в несколько отношений.

   Один объект может входить в отношение несколько раз.

   Тег 'тип' (type) устанавливает разновидность отношения. Отношения могут описывать замкнутые дороги, запреты поворотов и т.д.

   Расположение участников в отношении постоянно и определяется очередностью добавления. Повторяющиеся объекты сохраняют их определенный порядок.

3. Теги ('teg'). Тег строго говоря не элемент, а свойство объекта. Каждый тег имеет атрибуты - ключ (k) и значение (v). Ключи и их значения описывают объект и наделяют его свойствами.

Для всех элементов существуют общие атрибуты:

- user - имя пользователя, который совершал последнее изменение объекта;
- uid - числовой идентификатор пользователя, который совершил последнее изменение объекта;

- timestap - время последнего изменения;
- visible - является ли объект удаленным в базе данных. Если visible="false то объект должен быть возвращен вызовом истории изменений;
- version - версия редакции объекта. Версия вновь созданных объектов равна 1, это значение увеличивается на сервере, когда клиент добавляет новую версию объекта. Сервер будет отклонять новую версию объекта, если версия присланная клиентом не соответствует текущей версии объекта в базе данных;
- chengeset - пакет правок (история) в котором указаны создание и изменения объектов.

Особенности элемента 'точка'.

- атрибут id - числовой идентификационный номер, который уникален только среди точек. (Линия может иметь такое же id как и точка.);
- атрибут lat - координаты широты;
- атрибут lon - координаты долготы;
- теги - множество пар тегов (ключ - значение) с уникальным ключом;

Пример.

```
<node id="254" lat="51.5173639" lon="-0.140043" version="1" changeset="2" user="n" uid="1238" visible="true">
    <tag k="created_by" v="JOSM"/>
</node>
```

Особенности элемента 'линия'.

- атрибут id - числовой идентификационный номер, не являются уникальными, линия может иметь такой же идентификатор, как точка;
- nodes - теги (nd) с атибутом ref (идентификатор точки), являющиеся списком всех точек идентификаторы которых составляют линию;
- теги - множество пар тегов (ключ - значение) с уникальным ключом;

Пример.

```
<way id="5090250" visible="true" timestamp="2009-01-19T19:07:25Z" version="8" changeset="816806" user="Blumpsy" uid="64226">
  <nd ref="822403"/>
  <nd ref="21533912"/>
  <nd ref="821601"/>
  <nd ref="21533910"/>
  <nd ref="135791608"/>
  <nd ref="333725784"/>
  <nd ref="333725781"/>
  <nd ref="333725774"/>
  <nd ref="333725776"/>
  <nd ref="823771"/>
  <tag k="highway" v="unclassified"/>
  <tag k="name" v="Улица Ленина"/>
  <tag k="oneway" v="yes"/>
</way>
```

Особенности элемента 'отношение'.

- атрибут id - числовой идентификационный номер, не являются уникальными, линия может иметь такой же идентификатор, как точка;
- теги - множество пар тегов (ключ - значение) с уникальным ключом;

- members - упорядоченный список примитивов с атрибутами роль (где ролью может быть любой текст), тип (тип примитива), ref (идентификатор примитива).

Пример.

```
<relation id="12" timestamp="2008-12-21T19:31:43Z" user="kevjs1982" uid="84075">
  <member type="way" ref="2878061" role="outer"/>
  <member type="way" ref="8125153" role="inner"/>
  <member type="way" ref="8125154" role="inner"/>
  <member type="way" ref="3811966" role=""/>
  <tag k="created_by" v="Potlatch 0.10f"/>
  <tag k="type" v="multipolygon"/>
</relation>
```

3.3 XPath

3.3.1 Общие сведения

Язык XML Path (XPath) является набором синтаксических и семантических правил для ссылок на части XML-документов. XPath предназначен для использования другими спецификациями, например, такими как XSL Transformations (XSLT) и XML Pointer Language (XPointer).

Главная задача языка XPath - адресация частей в XML документе. Для достижения этой цели язык дополнительно наделен основными функциями для манипулирования строками, числами и булевыми значениями. В XPath используется компактный синтаксис, отличающийся от принятого в XML и облегчающий использование языка XPath. XPath работает не с внешним синтаксисом XML документа, а с его абстрактной логической структурой. XPath получил такое название потому, что использовался в URL для записи путей, обеспечивающих навигацию по иерархической структуре XML документа. Язык XPath спроектирован так, что помимо поддержки адресации он обладает естественным набором элементов, которые могут использоваться для сравнения (проверки, соответствует ли узел некому шаблону).

XPath представляет XML-документ как дерево узлов. Узлы могут быть разных типов, таких как узлы элементов, узлы атрибутов или узлы текста. Для каждого типа узлов в XPath определяется способ вычисления строкового значения. Некоторые типы узлов имеют также имя. XPath полностью поддерживает пространства имен XML. В результате, имя любого узла в этом языке образуется из двух частей: локальной части и URI (унифицированный (единообразный) идентификатор ресурса - это последовательность символов, идентифицирующая абстрактный или физический ресурс) некого пространства имен (возможно, нулевого). Такая комбинация называется расширенным именем.

Специальным типом узла является корневой узел. XML-документ содержит только один такой узел и он является корнем дерева, содержащего весь XML-документ. Корневой узел содержит корневой элемент, а также любые узлы инструкций обработки, объявлений или комментариев, которые появляются до или после корневого элемента.

Не существует типа узла для XML-декларации (такой, как <?xml version="1.0"encoding="UTF-8"?>). Следовательно, на такие сущности нельзя ссылаться в XPath.

Элементные узлы представляют каждый элемент в XML-документе. Узлы атрибутов принадлежат элементным узлам и представляют атрибуты в XML-документе. Однако, атрибуты, которые начинаются с xmlns: представляются в XPath с узлами пространств имен. Другие типы узлов включают в себя текстовые узлы, узлы инструкций обработки и узлы комментариев.

Главной синтаксической конструкцией языка XPath является выражение.

В результате обработки выражения получается объект, относящийся к одному из четырех основных типов:

- набор узлов (node-set) - неупорядоченный набор узлов без дубликатов;
- булево значение (boolean) - true или false;
- число (number) - число с плавающей точкой;
- строка (string) - последовательность UCS символов.

3.3.2 Типовые XPath-запросы

Рассмотрим типовые XPath-запросы на следующем примере:

```
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="CGImap 0.0.2">
 <bounds minlat="55.1196000" minlon="37.3708000" maxlat="55.1814000" maxlon="37.5074000"/>
 <node id="76475011" lat="55.0269826" lon="37.4271725" user="kolen" uid="73184" visible="true" version="13" changeset="813236" timestamp="2009-01-19T09:05:54Z"/>
 <node id="139744952" lat="55.0294985" lon="37.4277889" user="unwrecker" uid="69700" visible="true" version="7" changeset="3437942" timestamp="2009-12-23T21:29:34Z
   <tag k="railway" v="level_crossing"/>
 </node>
 <way id="116856438" user="Dimon62" uid="457129" visible="true" version="2" changeset="8409292" timestamp="2011-06-11T19:46:13Z">
   <nd ref="76475011"/>
   <nd ref="139744952"/>
   <tag k="addr:housenumber" v="96"/>
   <tag k="addr:street" v="Московская улица"/>
   <tag k="building" v="yes"/>
   <tag k="name" v="ТРЦ КАРНАВАЛ"/>
   <tag k="shop" v="mall"/>
 </way>
 <relation id="421030" user="Hind" uid="79836" visible="true" version="1" changeset="4008870" timestamp="2010-03-01T14:05:55Z">
   <member type="node" ref="76475011" role="a3"/>
   <member type="node" ref="139744952" role="label"/>
   <tag k="address:a2" v="Чеховский район"/>
   <tag k="address:type" v="a2"/>
   <tag k="name" v="Чеховский район"/>
   <tag k="type" v="address"/>
 </relation>
</osm>
```

Выражения для путей адресации:

/ - дочерний элемент узла. Результат: '/osm' - элемент osm;

// - элементы документа, которые соответствуют указанному шаблону. Результат: '//osm' - элемент osm и его подэлементы;

| - список дочерних элементов. Результат: 'osm|node' - список дочерних элементов.

Оси - это база языка XPath.

- ancestor:: — Возвращает множество предков.
- ancestor-or-self:: — Возвращает множество предков и текущий элемент.
- attribute:: — Возвращает множество атрибутов текущего элемента.
- child:: — Возвращает множество потомков на один уровень ниже.
- descendant:: — Возвращает полное множество потомков.
- descendant-or-self:: — Возвращает полное множество потомков и текущий элемент.
- following:: — Возвращает необработанное множество, ниже текущего элемента.
- following-sibling:: — Возвращает множество элементов на том же уровне, следующих за текущим.
- namespace:: — Возвращает множество имеющее пространство имён (то есть присутствует атрибут xmlns).
- parent:: — Возвращает предка на один уровень назад.
- preceding:: — Возвращает множество обработанных элементов исключая множество предков.

- preceding-sibling:: — Возвращает множество элементов на том же уровне, предшествующих текущему.
- self:: — Возвращает текущий элемент.

Существуют сокращения для некоторых осей, например:

- attribute:: — можно заменить на «@»;
- child:: — часто просто опускают;
- descendant:: — можно заменить на ’//’;
- parent:: — можно заменить на ’..’;
- self:: — можно заменить на ’.’;

Подробное описание функций см. в Приложении А.
(сюда по ходу работы занесу еще информацию)

3.4 PostgreSQL и OpenGIS

3.5 SQL

4 Практическая часть

4.1 Фрагменты кода

5 Приложение

5.1 Приложение А. Список функций XPath

Системные функции.

- node-set document(object, node-set?) - возвращает документ, указанный в параметре object.
- string format-number(number, string, string?) - форматирует число согласно образцу, указанному во втором параметре, третий параметр указывает именованный формат числа, который должен быть учтён.
- string generate-id(node-set?) - возвращает строку, являющуюся уникальным идентификатором.
- node-set key(string, object) - возвращает множество с указанным ключом (аналогично функции id для идентификаторов).
- string unparsed-entity-uri(string) - возвращает непроанализированный URI, если такового нет, возвращает пустую строку.
- boolean element-available(string) - проверяет, доступен ли элемент или множество, указанное в параметре. Параметр рассматривается как XPath.
- boolean function-available(string) - проверяет, доступна ли функция, указанная в параметре. Параметр рассматривается как XPath.
- object system-property(string) - параметры, возвращающие системные переменные, могут быть: - xsl:version — возвращает версию XSLT процессора.

  - xsl:vendor — возвращает производителя XSLT процессора.

  - xsl:vendor-url — возвращает URL, идентифицирующий производителя.

  Если используется неизвестный параметр, функция возвращает пустую строку.
- boolean lang(string) - возвращает истину, если у текущего тега имеется атрибут xml:lang, либо родитель тега имеет атрибут xml:lang и в нем указан совпадающий строке символ.

Функции с множествами.

* — обозначает любое имя или набор символов, @* — любой атрибут.

$name - обращение к переменной, где name — имя переменной или параметра.

[] — дополнительные условия выборки.

{} — если применяется внутри тега другого языка (например HTML), то XSLT процессор рассматривает содержимое фигурных скобок как XPath.

/ — определяет уровень дерева.

- node-set node() - возвращает все узлы. Для этой функции часто используют заменитель '*', но в отличие от звездочки — node() возвращает и текстовые узлы.
- string text() - возвращает набор текстовых узлов
- node-set current() - возвращает множество из одного элемента, который является текущим. Если мы делаем обработку множества с условиями, то единственным способом дотянуться из этого условия до текущего элемента будет данная функция.
- number position() - возвращает позицию элемента в множестве. Корректно работает только в цикле <xsl:for-each/>
- number last() - возвращает номер последнего элемента в множестве. Корректно работает только в цикле <xsl:for-each/>
- number count(node-set) - возвращает количество элементов в node-set.
- string name(node-set?) - возвращает полное имя первого тега в множестве.
- string namespace-uri(node-set?) - возвращает ссылку на url определяющий пространство имён.
- string local-name(node-set?) - возвращает имя первого тега в множестве, без пространства имён.
- node-set id(object) - находит элемент с уникальным идентификатором

Строковые функции.

- string string(object?) - возвращает текстовое содержимое элемента. По сути возвращает объединенное множество текстовых элементов на один уровень ниже.
- string concat(string, string, string*) - объединяет две или более строк
- number string-length(string?) - возвращает длину строки.
- boolean contains(string, string) - возвращает истину, если первая строка содержит вторую, иначе возвращает ложь.
- string substring(string, number, number?) - возвращает строку вырезанную из строки начиная с указанного номера, и если указан второй номер — количество символов.
- string substring-before(string, string) - если найдена вторая строка в первой, возвращает строку до первого вхождения второй строки.
- string substring-after(string, string) - если найдена вторая строка в первой, возвращает строку после первого вхождения второй строки.

- boolean starts-with(string, string) - возвращает истину если вторая строка входит в начало первой, иначе возвращает ложь.
- boolean ends-with(string, string) - возвращает истину если вторая строка входит в конец первой, иначе возвращает ложь.
- string normalize-space(string?) - убирает лишние и повторные пробелы, а также управляющие символы, заменяя их пробелами.
- string translate(string, string, string) - заменяет символы первой строки, которые встречаются во второй строке, на соответствующие позиции символам из второй строки символы из третьей строки. translate(«bar», «abc», «ABC») вернет BAr.

Логические функции

- or — логическое «или»
- and — логическое «и»
- = — логическое «равно»
- < (<) — логическое "меньше"
- > (>) — логическое "больше"
- <= (<=) — логическое "меньше либо равно"
- >= (>=) — логическое "больше либо равно"

Числовые функции

- + — сложение
- - — вычитание
- * — умножение
- div — обычное деление (не деление нацело!)
- mod — остаток от деления