Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего профессионального образования

**“Уральский федеральный университет**

**имени первого Президента России Б.Н.Ельцина”**

Математико-механический факультет

Кафедра вычислительной математики

РАЗРАБОТКА СПЕЦИАЛИЗИРОВАННОЙ СИСТЕМЫ ВИЗУАЛИЗАЦИИ И АНАЛИЗА ГОРОДСКОЙ СРЕДЫ

|  |  |  |
| --- | --- | --- |
| "Допущен к защите"  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_2011 г. |  | Дипломная работа  студента 5 курса Арефьева Алексея Андреевича  Научный руководитель  Авербух Владимир Лазаревич доцент КИПУ  заведующий сектором компьютерной  визуализации ИММ УрО РАН  к.т.н. |

Екатеринбург

2011

**РЕФЕРАТ**

Арефьев А.А. РАЗРАБОТКА СПЕЦИАЛИЗИРОВАНННОЙ СИСТЕМЫ ВМЗУАЛИЗАЦИИ И АНАЛИЗА ГОРОДСКОЙ СРЕДЫ, дипломная работа: стр. 111, рис. 32, библ. 16 назв.

Ключевые слова: ИНФОРМАЦИОННАЯ ВИЗУАЛИЗАЦИЯ, ГЕОИНФОРМАЦИОННАЯ СИСТЕМА, ЭЛЕКТРОННАЯ КАРТА, ИНФОРМАЦИОННАЯ ПОДДЕРЖКА ПОЛЬЗОВАТЕЛЯ, ВИЗУАЛЬНЫЙ АНАЛИЗ

Объект исследования – географические информационные системы.

Цель работы – анализ методов информационной визуализации и визуального анализа и применение их для разработки прототипа системы поддержки конечного пользователя.

В процессе работы проводился анализ возможностей географических информационных систем как средств визуализации и анализа данных, исследовались различные методы представления географических и семантических данных.

В результате работы разработан прототип информационной системы городской среды Екатеринбурга.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc295115057)

[1. ГИС КАК СРЕДСТВО ВИЗУАЛИЗАЦИИ И АНАЛИЗА ДАННЫХ РАЗЛИЧНОЙ ПРИРОДЫ 6](#_Toc295115058)

[1.1. Введение в ГИС 6](#_Toc295115059)

[1.2. Составные части ГИС 8](#_Toc295115060)

[1.3. Задачи, которые решает ГИС 10](#_Toc295115061)

[1.4. Основные понятия ГИС 13](#_Toc295115062)

[1.5. Модели ГИС 16](#_Toc295115063)

[1.6. Послойная организация данных 18](#_Toc295115064)

[1.7. Визуальная обработка информации в ГИС 19](#_Toc295115065)

[1.8. Анализ данных в ГИС 22](#_Toc295115066)

[2. ГИС В СОВРЕМЕННОМ ОБЩЕСТВЕ 24](#_Toc295115067)

[2.1. Гис и бизнес 24](#_Toc295115068)

[2.2. Некоторые применения ГИС в бизнесе 28](#_Toc295115069)

[2.3. Гис и транспорт 32](#_Toc295115070)

[3. АНАЛИЗ ФУНКЦИЙ СИСТЕМЫ 34](#_Toc295115071)

[4. ОСНОВНЫЕ ПРОБЛЕМЫ ТРАНСПОРТНОЙ СИСТЕМЫ 38](#_Toc295115072)

[ЕКАТЕРИНБУРГА 38](#_Toc295115073)

[5. ОПИСАНИЕ ПРОЕКТА СИСТЕМЫ 42](#_Toc295115074)

[5.1. Описание электронной карты 42](#_Toc295115075)

[5.3. Информация об объекте 51](#_Toc295115076)

[5.4. Описание средств поиска объектов на электронной карте 54](#_Toc295115077)

[5.5. Инструменты измерения расстояний и площади объектов 57](#_Toc295115078)

[5.6. Инструменты анализа транспортной системы города. «Узкие» места 59](#_Toc295115079)

[на дорогах Екатеринбурга 59](#_Toc295115080)

[5.7.Средства печати 64](#_Toc295115081)

[6. ПРИНЦИПЫ ПОСТРОЕНИЯ ЭФФЕКТИВНЫХ ВИДОВ 67](#_Toc295115082)

[ОТОБРАЖЕНИЯ ИНФОРМАЦИИ 67](#_Toc295115083)

[7. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ 70](#_Toc295115084)

[7.1. Формат карт shape 71](#_Toc295115085)

[7.2. Краткий обзор QGis 74](#_Toc295115086)

[7.3. Краткий обзор С# 76](#_Toc295115087)

[7.4. Обзор библиотеки MapWindow 78](#_Toc295115088)

[7.5. Описание структуры класса Map 79](#_Toc295115089)

[7.6.Описание структуры классов слоев 81](#_Toc295115090)

[8. СТРУКТУРНАЯ ОРГАНИЗАЦИЯ ПРОГРАММЫ 82](#_Toc295115091)

[8.1. Основное окно программы. Форма «MainForm» 82](#_Toc295115092)

[8.2. Окно информации об объекте. Форма «MyFeatureIdentifire» 85](#_Toc295115093)

[8.3. Окно печати. Форма «MyLayoutForm» 86](#_Toc295115094)

[ЛИТЕРАТУРА 88](#_Toc295115095)

[ПРИЛОЖЕНИЕ А 90](#_Toc295115096)

# ВВЕДЕНИЕ

XXI век называют веком информатизации всей сферы жизнедеятельности человека: управления, образования, здравоохранения, сельского хозяйства и многих других сфер. Одним из направлений является использование компьютерных технологий в повседневной деятельности, в частности обеспечение средствами информационной визуализации и визуального анализа представителей муниципальных органов власти, бизнесменов, менеджеров и продавцов, а так же потребителей различных услуг.

Наша задача – исследование возможностей применения методов информационной визуализации, в частности геоинформационной визуализации, в сфере торговли недвижимостью (квартиры, офисы и другие объекты), а так же анализе транспортной сети города. Частично эту задачу выполняют такие системы как 2Gis, «Яндекс.Карты». Однако, они носят общий характер и нуждаются в дополнительных настройках. Их главная задача – предоставление информации, а не поддержка работы конечного пользователя.

В ходе работы мной проанализированы задачи по информационной визуализации купли-продажи недвижимости и анализу транспортной сети города. Разработан прототип географической информационной системы, выполняющий их.

# 1. ГИС КАК СРЕДСТВО ВИЗУАЛИЗАЦИИ И АНАЛИЗА ДАННЫХ РАЗЛИЧНОЙ ПРИРОДЫ

## Введение в ГИС

Географическая информационная система (ГИС, геоинформационная система) - программно-аппаратный комплекс, предназначенный для сбора, управления, анализа и отображения пространственно распределенной информации. ГИС - не только и не столько информационные системы для географии, сколько информационные системы с географически организованной информацией. В простейшем варианте географические информационные системы - сочетание обычных баз данных (атрибутивной информации) с электронными картами, то есть мощными графическими средствами.

Основная идея ГИС - связь данных на карте и в базе данных. ГИС - это и аналитические средства для работы с любой координатно-привязанной информацией. В принципе, ГИС можно рассматривать как некое расширение концепции баз данных. В этом смысле ГИС фактически представляет собой новый уровень и способ интеграции и структурирования информации. ГИС предлагает совершенно новый путь развития картографии. Преодолеваются основные недостатки обычных карт - их статичность и ограниченная емкость как носителя информации. В последние десятилетия бумажные карты из-за перегруженности информацией становятся нечитабельными. ГИС же обеспечивает управление визуализацией информации. Появляется возможность выводить (на экран, на твердую копию) только те объекты или их множества, которые интересуют нас в данный момент. Фактически осуществляется переход от сложных комплексных карт к серии взаимоувязанных частных карт. При этом улучшается структурированность информации, а следовательно, повышается эффективность ее обработки и анализа. В ГИС карта оживает и становится действительно динамическим объектом в смысле:

• изменяемости масштаба;

• преобразования картографических проекций;

• варьирования объектным составом карты;

• возможности опрашивать через карту в режиме реального времени многочисленные базы данных;

• изменения способа отображения объектов (цвет, тип линии и т.п.), в том числе и определения символогии через значения атрибутов, то есть синхронизации визуализации с изменениями в базах данных;

• легкости внесения любых изменений.[1]

 Гис напрямую связан с картографией и другими науками (Рис.1)[2]

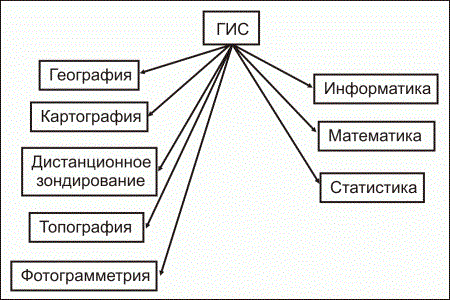


Рис.1 Связь ГИС с науками

## Составные части ГИС

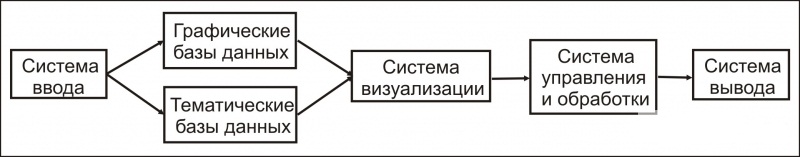
 Базы данных являются обязательными компонентами ГИС, всегда имеющими два их типа – графические и тематические. В графических базах данных хранится то, что принято называть топографической основой, тематические содержат нагрузку карт и дополнительные данные, которые относятся к пространственным, но не могут быть прямо нанесены на карту.   
      Кроме того, любая ГИС имеет систему визуализации данных, выводящую на экран или на плоттер (принтер) имеющуюся информацию в виде карт, таблиц, схем и т.п. и систему управления данными при помощи которой происходит их поиск, сортировка, удаление, добавление, исправление и анализ (Рис.2 и Рис.3)[2]

Рис.2. Составные части ГИС

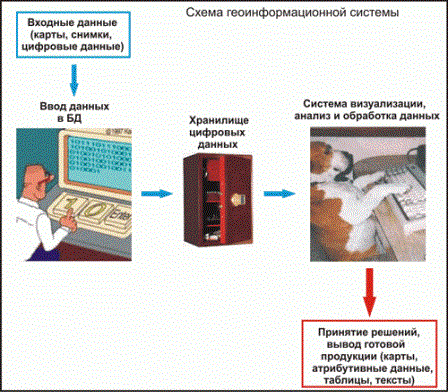
****

Рис.3 Схема ГИС

Система ввода – это программный или аппаратно-программный блок, отвечающий за получения данных. Например, дигитайзеры, на котором осуществляется оцифровка карт, сканер, считывающий изображение в виде растра, электронные геодезические приборы. Информация может быть введена с клавиатуры, получена по сети. Ее источником может быть аэрофото и космические снимки, вводимые и обрабатываемые на специализированных рабочих станциях или персональных станциях приема спутниковых данных.

    Система ввода:

* Клавиатура
* Внешние компьютерные системы (включая Интернет)
* Сканер
* Дигитайзер
* Электронные геодезические приборы
* Космические и аэрофотоснимки

    Система вывода ГИС предназначена для представления результатов информации в удобном для пользователя виде. При помощи плоттера можно получить очень качественные карты. Используются также принтеры. Результаты могут быть представлены на видеофильмах, хранится на диске.[2]

## Задачи, которые решает ГИС

ГИС общего назначения, в числе прочего, обычно выполняет пять процедур (задач) с данными:

1. ввод,
2. манипулирование,
3. управление,
4. запрос и анализ,
5. визуализацию.

**Ввод.**

Для использования в ГИС данные должны быть преобразованы в подходящий цифровой формат. Процесс преобразования данных с бумажных карт в компьютерные файлы называется оцифровкой. В современных ГИС этот процесс может быть автоматизирован с применением сканерной технологии, что особенно важно при выполнении крупных проектов, либо, при небольшом объеме работ, данные можно вводить с помощью дигитайзера. Многие данные уже переведены в форматы, напрямую воспринимаемые ГИС-пакетами.

**Манипулирование.**

Часто для выполнения конкретного проекта имеющиеся данные нужно дополнительно видоизменить в соответствии с требованиями вашей системы. Например, географическая информация может быть в разных масштабах (осевые линии улиц имеются в масштабе 1: 100 000, границы округов переписи населения - в масштабе 1: 50 000, а жилые объекты - в масштабе 1: 10 000). Для совместной обработки и визуализации все данные удобнее представить в едином масштабе. ГИС-технология предоставляет разные способы манипулирования пространственными данными и выделения данных, нужных для конкретной задачи.

**Управление.**

В небольших проектах географическая информация может храниться в виде обычных файлов. Но при увеличении объема информации и росте числа пользователей для хранения, структурирования и управления данными эффективнее применять системы управления базами данных (СУБД), то специальными компьютерными средствами для работы с интегрированными наборами данных (базами данных). В ГИС наиболее удобно использовать реляционную структуру, при которой данные хранятся в табличной форме. При этом для связывания таблиц применяются общие поля. Этот простой подход достаточно гибок и широко используется во многих, как ГИС, так и не ГИС приложениях.

**Запрос и анализ.**

При наличии ГИС и географической информации можно получать ответы на простые вопросы (Кто владелец данного земельного участка? На каком расстоянии друг от друга расположены эти объекты? Где расположена данная промзона?) и более сложные, требующие дополнительного анализа, запросы (Где есть места для строительства нового дома? Каков основный тип почв под еловыми лесами? Как повлияет на движение транспорта строительство новой дороги?). Запросы можно задавать как простым щелчком мышью на определенном объекте, так и с посредством развитых аналитических средств. С помощью ГИС можно выявлять и задавать шаблоны для поиска, проигрывать сценарии по типу “что будет, если…”.

**Визуализация.**

Для многих типов пространственных операций конечным результатом является представление данных в виде карты или графика. Карта - это очень эффективный и информативный способ хранения, представления и передачи географической (имеющей пространственную привязку) информации. Раньше карты создавались на столетия. ГИС предоставляет новые удивительные инструменты, расширяющие и развивающие искусство и научные основы картографии. С ее помощью визуализация самих карт может быть легко дополнена отчетными документами, трехмерными изображениями, графиками и таблицами, фотографиями и другими средствами, например, мультимедийными.[3]

## Основные понятия ГИС

Рассмотрим основные понятия ГИС, в том или ином виде присутствующие во всех современных геоинформационных системах.

**Данные**

В ГИС данные делятся на две категории:

• пространственные (местоположение);

• семантические (атрибуты).

Это вероятно наиболее важный компонент ГИС. Данные о пространственном положении и связанные с ними табличные данные могут собираться и подготавливаться самим пользователем, либо приобретаться у поставщиков на коммерческой или другой основе. В процессе управления пространственными данными ГИС интегрирует пространственные данные с другими типами и источниками данных, а также может использовать СУБД, применяемые многими организациями для упорядочивания и поддержки имеющихся в их распоряжении данных.

**Объекты**

Пространственные данные включают географические объекты, представляемые:

• точками;

• линиями;

• полигонами.

*Дугами* описываются те реальные объекты, которые можно рассматривать как линии. Дуга состоит из отрезков линий и дуг окружностей.

*Полигоны* - замкнутые области, которые представляют однородные по некоторым критериям участки.

*Атрибутивные данные* могут включать идентификатор объекта, любую описательную информацию из баз данных, изображение и многое

другое.

**Слой**

Слои в карте подразделяются на два основных вида - растровые и векторные.

*Векторные слои* - это совокупность простых геометрических объектов (точка, дуга, полигон), которые представляют те или иные объекты на местности. Векторные слои могут также хранить топологию, т.е. информацию о взаимном расположении объектов.

*Растровые слои* представляют из себя сплошные изображения. Они не могут содержать объекты. Однако они могут служить фоном для векторных слоев.

**Объект слоя**

Каждому объекту векторного слоя может соответствовать запись в базе данных, чем обеспечивается привязка информации к местности. Это соответствие может обеспечиваться в частности назначением каждому объекту соответствующего идентификатора.

**Легенда карты**

Легенда карты - свод *условных обозначений*, использованных на карте, с текстовыми пояснениями к ним. Обычно, легенды создаются на основе классификаций изображаемых объектов и явлений, они становятся их графической моделью и часто служат для построения классификаторов.

**Карта**

Представляет собой набор географических слоев, каждый из которых привносит в карту информацию по какой-либо определенной теме. Например, на слой границ некоторой территории может быть нанесен слой рек, затем слой, отображающий количество атмосферных осадков в процентном отношении и т.д.

Электронную карту в ГИС можно рассматривать как

многокомпонентную модель реальности. Основными целями ее создания

являются:

• графическая коммуникация пространственных отношений и распределений;

• улучшение возможности анализа, обработки и отображения

географических информационных данных;

• визуальное отображение цифровых моделей явлений, невидимых для человеческого глаза;

• автоматизация отображения и картографического анализа в системах управления; исследование объектов, явлений и процессов с учетом динамики их развития и возможного использования;

• получение аналитических решений в графическом виде в режимах реального и разделенного времени и т.д.[1]

## Модели ГИС

Основой визуального представления данных при помощи ГИС-технологий служит так называемая графическая среда. Основу графической среды и соответственно визуализации базы данных ГИС составляют векторные и растровые модели. В общем случае модели пространственных (координатных) данных могут иметь векторное или растровое (ячеистое) представление, содержать или не содержать топологические характеристики. Этот подход позволяет классифицировать модели по трем типам:

• растровая модель;

• векторная нетопологическая модель;

• векторная топологическая модель.

Все эти модели взаимно преобразуемы. Тем не менее, при получении каждой из них необходимо учитывать их особенности. В ГИС форме представления координатных данных соответствуют два основных подкласса моделей - векторные и растровые (ячеистые или мозаичные). Возможен класс моделей, которые содержат характеристики как векторов, так и мозаик. Они называются гибридными моделями.

Между векторными и растровыми изображениями имеется различие, характерное именно для ГИС. Растровые изображения отображают поля данных, т.е. носят полевой характер. Векторные изображения в ГИС, как правило, отображают геоинформационные объекты, т.е. носят объектный характер.

Если векторная модель дает информацию о том, где расположен тот или иной объект, то растровая - информацию о том, что расположено в той или иной точке территории. Это определяет основное назначение растровых моделей - непрерывное отображение поверхности.

Данные для анализа могут быть получены из векторных слоев, отражающих поля тематических или/и временных характеристик, растеризацией и записаны в таблицу или напрямую занесены туда из отчетов. Таблица, содержащая атрибуты объектов, называется таблицей атрибутов. В таблице каждому объекту соответствует строка таблицы, каждому тематическому признаку - столбец таблицы. Каждая клетка таблицы отражает значение определенного признака для определенного объекта.

В общем случае ввод информации для задач ГИС осуществляется комплексно: по данным дистанционного зондирования, со снимков спутников, аэроснимков, по материалам дешифрования снимков, полевым измерениям, по информации с карт.[1]

## 1.6. Послойная организация данных

Концепция послойного представления графической информации заимствована из систем CAD, однако в ГИС она получила качественно новое развитие. Технологически организация слоев основана на типизации данных. Множество разнообразных данных имеет различные характеристики и в процессе визуальной обработки это множество может быть информационно перегружено. Для уменьшения информационной нагрузки на оператора графические данные типизируют и объединяют в слои. Таким образом, разбиение на слои упрощает процесс обработки и повышает ее качество. Слои в ГИС могут быть как векторными, так и растровыми, причем векторные слои обязательно должны иметь одну из трех характеристик векторных данных. Т.е. векторный слой должен быть определен как точечный, линейный или полигональный дополнительно к его тематической направленности.

Каждый слой может использоваться, как отдельно, так и в комплексе.(Рис.4) С помощью системы фильтров или заданных параметров объекты, принадлежащие слою, могут быть одновременно масштабированы, перемещены, скопированы, записаны в базу данных. В других случаях (при установке других режимов) можно наложить запрет на редактирование объектов слоя, запретить их просмотр или сделать невидимыми. Введение топологических свойств в графические данные ГИС позволяет решать многие задачи.

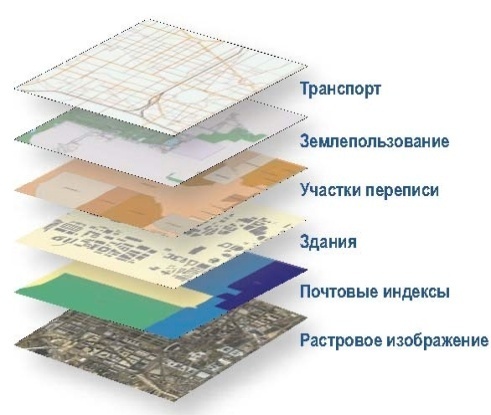


Рис.4 Послойная организация данных

## 1.7. Визуальная обработка информации в ГИС

Одним из противоречий большинства существующих информационных систем является противоречие между высокоскоростной обработкой данных в компьютерной среде и низкой пропускной способностью канала "человек - компьютер", особенно в режимах интерактивной обработки. Для повышения производительности интерактивной обработки предлагают разные подходы. Одним из таких является метод визуальной обработки информации, основанный на выделении и обобщении необходимых данных и представлении их в визуальной форме. Данные, представление в графическом виде позволяют выявлять отдельные явления на порядки быстрее, чем анализ табличной или текстовой информации. Эффективность визуальной обработки информации выражается в том, что она позволяют подключить к активной работе по принятию решения резервы образного, ассоциативного мышления. Представление ситуации в виде образов обобщает информацию и позволяет принимать решение специалисту в данной предметной области. Визуальная обработка информации представляет собой комплекс технологий, основанных на: группировании и обобщении исходных данных и сопоставление характеристикам данных графических образов; применении методов компьютерной графики для обобщения, анализа и представления информации; применении объектно-ориентированного подхода для построения моделей графических и неграфических объектов; применении современных интеллектуальных или полуинтеллектуальных графических интерфейсов. Можно сказать, что визуальная обработка основана на использовании развитых систем компьютерной графики, включающих в свой состав базы данных моделей (шаблонов - объектов) и базы данных процедур (методов обработки). Примером такого подхода может служить визуальное программирование, которое допускает написание программ традиционным способом и дополнительно к этому позволяет оперативно создавать программные проекты без использования традиционного написания программ с созданием необходимого графического интерфейса. Визуальная обработка на уровне пользователя упрощает процесс обработки данных и снижает уровень, требуемых при обычных методах обработки, специальных знаний в области программирования и предметной области. Визуальная обработка информации основана на использовании дополнительных графических интерфейсов, позволяющих обобщать данные в удобном для пользователя виде и избегать по мере возможности вопросов, требующих специальной подготовки в данной предметной области. Эффект от технологии визуальной обработки информации во многом зависит от развитости используемых методов сбора информации, структурирования данных, построения сценариев и применяемых технологий. Большой объем достоверной информации о различных аспектах явления - признак устойчивости его динамики, залог эффективности принимаемых корпоративных решений. На ней можно построить надежную модель развития явления. При малом количестве достоверной информации особого внимания заслуживают некоторые подобласти методов искусственного интеллекта. ГИС как интегрированная информационная система включает в свой состав систему компьютерной графики и базы данных. По этой причине визуальная обработка информации в ГИС - естественное расширение возможности ее технологий. Именно это свойство делает привлекательными и доступными ГИС для широкого круга пользователей, от которых не требуется знаний ни в геодезии, ни в картографии. Для массового пользователя ГИС появились именно как системы поддержки принятия решений, использующие визуальные методы деловой и компьютерной графики. С этих позиций ГИС можно рассматривать как особый вид систем компьютерной графики. Принципиальным является то, что ГИС позволяет визуально представлять разные объекты и явления в единой системе поверхности Земли. Основой визуальной обработки информации служат графические модели, хранимые в ГИС. Как уже отмечалось это цифровые карты и цифровые модели.

Визуальная обработка информации позволяет пользователю, не рассматривая атрибутивные табличные данные, работать с графическими данными, что существенно повышает скорость обработки и анализа данных. Например, классическая организация запросов в БД требует использования специальных языков SQL или QBE. ГИС обладает этими возможностями как любая информационная система. Однако дополнительно к этому она позволяет организовывать запросы только на основе манипуляций с графическими данными и графическим интерфейсом. При этом результат запроса может быть представлен как в табличном, так и в графическом виде. Например, поиск объекта в заданной зоне. В обычной базе данных результатом поиска может быть таблица, совокупность таблиц, справка. В ГИС результатом поиска является фрагмент территории с искомым объектом. Для данного объекта по мере необходимости могут быть выведены все атрибуты, хранимые в базе данных, а также выполнены расчеты, показывающие, например, его удаление от другого объекта. Фрагмент территории может быть детализован для выявления подробностей или наоборот переведен в более мелкий масштаб и генерализован. При этом может быть получена визуальная совокупность детализованных фрагментов с общей генерализованной картиной в мелком масштабе. Следует помнить, что визуальная обработка использует концепцию объектно-ориентированного подхода. Последний требует более глубокой проработки создания информационной системы, включая ГИС. Применение визуальной обработки информации является эффективным методом использования интеллектуального потенциала, информационных потоков, телекоммуникаций, средств мультимедиа и геоинформационных технологий при управлении и поддержке принятия решений. Специалисты различных областей уже начинают осознавать важность данного компонента в системе управления и в следствии этого все большее внимание уделяют геоинформационым системам как системам управления и поддержки принятия решений.[4]

## 1.8. Анализ данных в ГИС

Далеко не все ГИС снабжены возможностями специализированного анализа. Связано это с тем, что четкой схемы проведения таких работ не существует и организации, занимающиеся ими, предпочитают производить анализ по собственным методикам и правилам. Кроме того, взгляды на приемы проведения анализа в ГИС могут меняться с течением времени. Сложность состоит в том, что для каждой специализированной области возникает необходимость создавать отдельное приложение и часто даже свою методику обработки. Это не всегда возможно и часто дорого.

Географические комплексы плохо поддаются формализации. Существующий математический аппарат недостаточно приспособлен для решения географических задач. Формулировки географических задач, описания явлений допускают некоторый произвол или двоякое толкование. Строгие алгоритмы могут не соответствовать уровню строгости и точности самих задач. Это иногда приводит к результатам , не отвечающим существу и содержательному смыслу. В связи с этим особое внимание стоит уделить именно визуальному анализу данных.[1]

Информация в ГИС может содержать в себе скрытые знания, закономерности и потому, при соответствующем анализе, способна оказать влияние при принятии решений в различных областях человеческой деятельности.

Существует множество способов поиска скрытых закономерностей в данных алгоритмами, но также не стоит упускать из вида возможности человека. Полезно сочетать огромные вычислительные ресурсы современных компьютеров с творческим и гибким человеческим мышлением.  
Визуальный анализ данных призван вовлечь человека в процесс поиска знаний в данных. Основная идея заключается в том, чтобы представить большие объёмы географических и атрибутивных данных в такой форме, где человек мог бы увидеть то, что трудно выделить алгоритмически, смог погрузиться в эти данные, работать с их визуальным представлением, понять их суть и сделать выводы.

С помощью ГИС пользователь способен оценивать: большие объекты и маленькие, далеко они находятся или близко. Пользователь в реальном времени может перемещаться по карте и исследовать ее.   
Визуальный анализ данных особенно полезен, когда о самих данных мало что известно и цели исследования до конца не понятны. За счёт того, что пользователь напрямую работает с данными, представленными визуально, он может получить дополнительную информацию, которая поможет ему более чётко сформулировать цели исследования.

ГИС является хорошей средой для внедрения методов визуального анализа информации. Это вызвано тем, что большинство проблем и задач в ГИС слабо структурировано и слабо формализовано.

Визуальный анализ данных в ГИС можно выполнять в три этапа:

* беглый анализ - позволяет идентифицировать интересные объекты на карте и сфокусироваться на одном или нескольких из них;
* увеличение и фильтрация - идентифицированные на предыдущем этапе объекты отфильтровываются и рассматриваются в большем масштабе;
* детализация по необходимости - если пользователю нужно получить дополнительную информацию, он может визуализировать более детальные данные.

# ГИС В СОВРЕМЕННОМ ОБЩЕСТВЕ

## 2.1. Гис и бизнес

ГИС может помочь делать успешный (доходный) бизнес всем предпринимателям. ГИС – это инструментальное средство для управления бизнес информацией любого типа с точки зрения ее пространственного местоположения. Приложения этой технологии в сфере бизнеса разнообразны. Основные решаемые с ее помощью задачи можно сгруппировать по ответам на базовые вопросы: "Где?", "Кто (или Что)?" и "Как?". Вы сможете: проследить, где проживают ваши клиенты, кто они такие, каковы их потребности и финансовые возможности; определить расположение магазинов – ваших и ваших конкурентов; узнать, как точнее направить маркетинговую активность и как получить от нее наибольшую отдачу, как оптимизировать области продаж и смоделировать последствия принимаемых решений; подобрать дом для покупки и определить кратчайший маршрут проезда к нужному месту. Впрочем, эти вопросы общие, они присутствуют и в других областях применения ГИС, а не только в сфере бизнеса, поскольку большая часть информации, с которой мы имеем дело, в явном или в неявном виде привязана к определенному месту или конкретной территории.

На подобные вопросы ГИС отвечает с большей эффективностью и определенностью, чем любые другие информационные технологии, интегрируя широкий набор данных, хранящихся в электронных таблицах и других видах документов, в одном удобном и легком для понимания формате – карте. С помощью ГИС Вы можете получить наиболее наглядное представление об этих данных. Сможете выделить наиболее интересные для Вас данные, меняя значки соответствующих символов, их цвет и их значения в таблицах баз данных. Сможете создать и поместить на карту или рядом с ней поясняющие диаграммы, графики, таблицы, чертежи и снимки. Сможете совместно отобразить разные типы данных в одном географическом пространстве, либо выделить из базы данных и отобразить на карте данные, связанные с конкретной тематической задачей. И, наконец, отобразив нужные данные на карте или нескольких картах, провести их полноценный анализ. ГИС позволяет создавать и изменять карты "на лету", моментально переходить от объекта или слоя карты к соответствующей строке или таблице базы данных и из записи в базе данных к связанному с ней объекту на карте.

Эффективность решения стоящих перед бизнесменами задач с помощью ГИС значительно повышается. Этому способствует все большая доступность и достоверность исходных данных, а также появление все более мощных и, одновременно, достаточно дешевых персональных компьютеров и простых в использовании современных программных продуктов. Для поддержки принятия решений в сфере бизнеса могут успешно применяться ГИС продукты общего назначения, имеющие средства настройки под конкретные задачи и возможности взаимодействия с другими применяемыми в этой области информационными технологиями: технологией управления активами предприятия, например SAP R/3; технологией генерирования отчетной документации, например Seagate Crystal Reports; системами управления базами данных (СУБД), технологиями инженерного проектирования (САПР); технологией компрессии данных, например MrSID, и др. Имеются и готовые специализированные ГИС пакеты, обеспечивающие решение типовых бизнес задач.

С ГИС Вы можете достичь значительно большего, чем просто отобразить ваши данные на карте. ГИС объединяет средства обычных пакетов картографического отображения, функции тематического представления информации на основе привязки табличных данных к адресам и улицам, возможности анализа географических местоположений с учетом дополнительной информации по находящимся в этих местах объектам. Эта технология связывает воедино инструменты графического отображения, работу с электронными таблицами, базами и хранилищами данных. Функции пространственного анализа позволяют, например, с помощью ГИС решить, где следует открыть новый магазин, аптеку или отделение банка, основываясь на новых демографических данных и планах развития города. Вы можете сразу получить нужную информацию об объекте, щелкнув на нем на электронной карте, либо создать и отобразить карту на основе информации, выбранной в базе данных. Причем связь карты с данными динамическая. Созданные вами карты не привязаны к отдельному моменту времени. В любой момент Вы можете обновить информацию, привязанную к карте, и внесенные изменения автоматически отразятся на карте. И для этого не нужно специальной подготовки.

Теперь ГИС, больше чем когда-либо, означает реальный бизнес. Ее внедрение приносит доход и, порой, немалый. Многонациональных корпорации и малые предприятия, магазины и больницы, риэлторские фирмы и транспортные предприятия, страховые обществ и предприятия энергетического комплекса, телефонные и телекоммуникационные фирмы - самые разные компании все чаще используют возможности географического анализа для решения свои деловых задач. За счет этого они получают преимущество в конкурентной борьбе, так как быстрее находят оптимальные решения, выявляют новые рынки и новые перспективные области сбыта своих товаров и услуг, лучше обслуживают заказчиков, точнее направляют рекламные компании, лучше контролируют и оптимально перераспределяют материальные и финансовые ресурсы.

ГИС позволяет создать новые данные, легко обратиться к уже существующим бизнес данным и связать их с пространственной информацией, чтобы выявить те особенности и взаимосвязи, которые не видны из таблиц, диаграмм и графиков. Недостающие для детального бизнес анализа данные, например, о населении, расположении предприятий, банков, объектов недвижимости, можно быстро добавить в ГИС, получив их от партнеров или из коммерческих источников.

ГИС позволяет Вам создавать картографические отображения и карты для презентаций, просто указав на них и сделав щелчок кнопкой мыши. ГИС позволяет отображать и анализировать бизнес информацию новыми методами, выявлять скрытые ранее взаимосвязи, примеры и тренды. ГИС может изменить стиль вашей работы.

Современным деловым людям приходится иметь дело с огромными объемами информации о продажах, клиентах, партнерах и конкурентах, демографии жителей, списками рассылки и многим другим. В основе этой информации лежит географическое расположение: адрес, почтовый индекс, граница зоны обслуживания, область сбыта продукции, маршрут доставки. Вся эта информация может быть отображена на карте, и ею можно управлять в интерактивном режиме.[5]

## 2.2. Некоторые применения ГИС в бизнесе

**ГИС для демографического анализа.**

Демографический анализ является основой для принятия решений во многих бизнес задачах: предоставление услуг клиентам, подбор мест для строительства, следование местных постановлениям, маркетинговые исследования и рекламные компании. Знание ваших клиентов, их нужд и возможностей критически важно для успеха в бизнесе. Понимание пространственной демографии населения важно также для составления списков прямой рассылки рекламы по почте, составления подходящих рекламных брошюр, проведения рекламных компаний в средствах массовой информации. За основу здесь берутся привязанные к карте данные об образе жизни, предпочтениях при покупках, душевых доходах и других важных критериях. ГИС позволяет создать демографические шаблоны и соответствующие карты на основе информации, получаемой при опросах и анкетировании покупателей в магазинах, по телефону и т.д., и последующей привязки собранных данных к адресам. При этом можно выявить примеры и тенденции, не очевидные при простом просмотре электронных таблиц, содержащих эти данных. Эти данные, или их часть, как правило, уже хранятся в корпоративной базе данных. Помощь в проведении демографического анализа могут также оказать специализирующиеся на подобных исследованиях компании.

**ГИС для связи с клиентами и партнерами.**

В среднем затраты на привлечение нового клиента в пять раз превышают затраты на сохранение существующих клиентов. Отделы по работе с клиентами рассматривают все аспекты бизнеса, от определения наилучшего продукта для конкретного клиента до рассылки товаров и предоставления дополнительных услуг клиентам в их доме или офисе. Одной из обычных задач службы по работе с клиентами является их перенаправление к ближайшему дилеру или сервисному центру, имеющим необходимый товар или предоставляющим нужную услугу. ГИС использует информацию об адресе клиента и данные из корпоративной базы данных для того, чтобы определить, где находится клиент, и выдает карту расположения ближайших дилеров и сервисных центров компании. Возможность интеграции карт и табличных данных делает ГИС технологию идеальным средством повышения качества обслуживания клиентов.

**ГИС для доставки товаров и маршрутизации**.

С помощью ГИС можно внедрить функции географического анализа в процесс обслуживания клиентов: от расчета времени и кратчайшего маршрута проезда к клиенту до составления маршрутного листа и расписания движения при обслуживании нескольких клиентов. Точное планирование доставки ведет к значительной экономии средств (например, компания Sears, ведущий поставщик розничных товаров, экономит на этом десятки миллионов долларов в год). Усовершенствованные функции сетевого анализа позволяют отследить движение машин по маршрутам, выделить варианты доставки с учетом времени суток, транспортных нагрузок, наличного числа автомашин и т.д.

**ГИС для выбора и анализа местоположений**.

ГИС предоставляет удобные средства поиска подходящего места для нового магазина, склада или сервисного центра. Вы сможете сопоставить информации по клиентам с данными переписи населения, чтобы выяснить степень проникновения товаров данного вида на рынок, свою долю на этом рынке и в данном регионе. При изменении коньюктуры ГИС поможет составить планы безопасного ухода с данного рынка с учетом распродажи принадлежащей вам собственности. Примером может служить опыт компании Texaco/Star. По данным о том, в какое время и в каких пунктах ее клиенты останавливаются на заправку горючим, куда они направляются, какие сопутствующие товары и дополнительные услуги они обычно приобретают, компания составила представление о типах своих клиентов и использовала полученные шаблоны поведения для выявления возможностей повышения доходов на имеющихся заправочных станциях и выбора мест для открытия новых заправок.[5]

Рассмотрим пример применения ГИС в бизнесе.

Издательский дом "Коммерсантъ" решил проанализировать распределение подписчиков выпускаемых им изданий по территории Москвы. Это позволило бы рекламодателям и издателям скорректировать содержание каждого из изданий; точнее определить объемы заказов изданий по районам с разным спросом; оптимально разместить пункты реализации; выявить районы потенциального спроса и целенаправленно проводить рекламную компанию.

Результаты исследований приведены ниже.

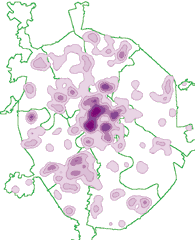


Рис.5. Подписчики журнала "Автопилот"

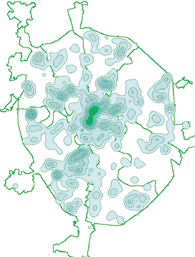


Рис.6. Подписчики журнала "Деньги".

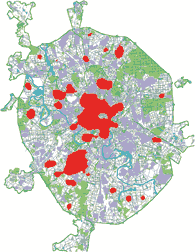
****

Рис.7. Области с наивысшим суммарным показателем реализации всех изданий.

## 2.3. Гис и транспорт

Территориальная распределенность транспортных систем делает их идеальным объектом автоматизации посредством геоинформационных систем. Вообще говоря, ГИС являются оптимальной платформой для *комплексных* решений в сфере транспорта. Ведь пространственная составляющая является естественной основой интеграции задач управления транспортной инфраструктурой, расчетных задач, задач оперативного управления, навигации и т.д. Внедрение ГИС-технологии у нас происходит по отдельным целевым направлениям, а не по всему «фронту» транспортных и смежных с ними задач. Рассмотрим некоторые из этих направлений.

**Управление парком транспортных средств (ТС)**. Эта задача стоит перед коммерческими перевозчиками, осуществляющими заказную транспортировку грузов и пассажиров (такси), перед сетевыми торговыми компаниями, сбытовыми подразделениями нефтяных компаний, а также компаниями, торгующими по каталогам и через интернет-магазины. Цель – снизить общие расходы на транспортировку и ускорить выполнение заказов. Это классическая задача транспортной логистики.

**Построение и оптимизация маршрутов на существующей дорожной сети**. В больших городах это насущная задача. В Москве, например, больше тысячи маршрутов общественного транспорта (не считая "маршруток"). Удержать их в одной голове и тем более проанализировать просто невозможно. К тому же оптимизировать нужно не один вид транспорта, а всех их в комплексе – метро, автобусы, трамваи, троллейбусы, электрички. Эта задача – наиболее сложная организационно, так как требует координации большого количества управляющих организаций. Она сложна также и технически, так как требует сбора, систематизации и анализа большого объема исходных данных.

**Мониторинг состояния дорожного полотна и планирование ремонтов**. Это одно из наиболее популярных направлений применения ГИС в дорожных администрациях. Часто одного лишь цветового кодирования участков дорог по срокам ремонта бывает достаточно, чтобы существенно оптимизировать процесс и повысить качество дорожного покрытия в целом. Если же использовать ГИС для интеграции разносторонней информации по дорожной сети (виды/качество покрытия, транспортная нагрузка, даты ремонтов), на ее основе можно построить динамическую модель износа и автоматизировать планирование ремонтов (на Западе уже давно так делают). В базе геоданных удобно хранить и сведения о дорожных знаках, и другую «придорожную» информацию, привязанную к географическим или линейным координатам.

**Информационные услуги населению**. Информация о дорогах, маршрутах, расписаниях нужна нам всем. Средства для ее картографического представления в Интернете существуют уже 10 лет. И при этом сложилась парадоксальная ситуация практического отсутствия у нас информационных услуг для массового потребителя. Причины известны – конституированный монополизм государства на пространственную информацию при его фактической незаинтересованности в предоставлении сервисов на ее основе. Секретность координатных определений с необходимой точностью до сих пор остается принципиальным препятствием. Для инвесторов и коммерческих компаний этот рынок очень интересен, но неопределенность правового поля и, соответственно, невозможность просчитать риски, отваживает их от вложений в это направление.[6]

# 3. АНАЛИЗ ФУНКЦИЙ СИСТЕМЫ

Как уже упоминалось, наша задача - исследование возможностей применения методов геоинформационной визуализации и визуального анализа в сфере торговли недвижимостью и при решении проблем городской инфраструктуры. В связи с этим возникает необходимость в анализе потребностей конечного пользователя системы. В нашем случае это продавцы и покупатели объектов недвижимости, а так же представители муниципальных органов власти. Следовательно, необходим анализ функций будущей системы.

На динамично развивающемся рынке торговли недвижимостью России аналитические отделы риэлторских фирм оперируют огромными объемами информации, поскольку от своевременности обработки этих данных во многом зависит успех развития бизнеса. Процесс автоматизации обработки информации в большинстве фирм рынка недвижимости построен на основе средств СУДБ и БД. Опыт работы этих компаний показывает, что табличные данные без привязки к географической основе не дают полной картины. Поэтому в последнее время специалисты, работающие на рынке торговой недвижимости, все активнее используют в аналитической работе ГИС-технологии.

Структура аналитического обеспечения работ на рынке недвижимости тесно связана с необходимостью учета пространственной информации. Это проявляется как на региональном уровне (престижность направления (шоссе), условия проезда, расположение относительно объектов инфраструктуры и др.), так и на локальном (наличие лесных деревьев, уклоны участков, наличие обременений и т.д.). Отображение всех объектов на карте одновременно практически невозможно с точки зрения ее восприятия и анализа. Следовательно необходимо разбить все интересующие объекты по уровням, разместив их на различных слоях карты. Многослойность предоставит пользователю возможность работать и анализировать только те данные, которые интересуют его в данный момент, начиная от целого района и заканчивая конкретным зданием или автобусной остановкой.

Важным моментом при оказании консультационных услуг является возможность отображения как исследуемого участка на карте города, так и среды вблизи него. Потенциальных покупателей могут интересовать различные факторы и особенности того или иного района. Отображение объектов карты, основанное на потребностях потребителя услуг, можно добиться с помощью системы фильтров.

В работе аналитика ГИС является одним из наиболее простых и удобных инструментов визуализации информации о состоянии рынка, существенно помогая выполнять прогнозирование и моделирование возможных ситуаций. Разрабатываемая нами информационная система должна позволять визуально оценить насыщенность рынка, определять зоны концентрации офисных центров, выделять наиболее интересные и перспективные районы на карте города. Все решения, принимаемые пользователями системы, должны основываться на визуальном анализе. Источником информации при этом является не только карта с набором географических объектов, но и статистические данные. Всё это позволит сделать правильный и надежный вывод.

Одновременно на карте может отображаться сотни и тысячи объектов разного характера. Необходимо предоставить пользователю возможность быстро осуществлять поиск по интересующим показателям. Это может быть район города, этаж, цена и т.д. Удобная система запросов должна позволить продавцу в кратчайшие сроки подобрать подходящий покупателю вариант.

Как уже говорилось, существующие ГИС многофункциональны и их основная задача – представление информации. При анализе инфраструктуры города требуются дополнительные инструменты, позволяющие понять причины тех или иных явлений в жизни города. Если рассматривать такое явление как дорожные пробки, то необходимо проанализировать возможные причины их возникновения, информативно отобразить их на карте города, тем самым дав представителям муниципальных органов проанализировать картину в целом. Пользователь должен иметь возможность, основываясь на своем опыте, расставлять приоритеты различных параметров, от которых зависит движение транспортного потока города. Следовательно необходимо разработать инструментарий, позволяющий пользователю как анализировать ситуацию в городе в целом, так и фокусироваться на каких-то отдельных моментах.

Любая ГИС обладает системой вывода. Пользователь всегда должен иметь возможность просмотреть информацию на экране монитора или распечатать на бумаге интересующие его данные. Поэтому одной их основных функций нашей системы должна стать возможность быстрой печати результата исследований. Учитывая специфику разрабатываемой ГИС особое внимание должно быть уделено добавлению на печатную форму дополнительной информации об объекте: фотографии, схемы, описания.

К сожалению, мы не обладаем всей полнотой статистической информации, на основании которой можно было бы построить точный визуальный анализ. Тем не менее, в качестве данных будем применять усредненные параметры, которые при дальнейшей работе могут быть заменены. Вместе с тем, есть значительный объем данных топографии районов города Екатеринбург, дорог, зданий и иных объектов инфраструктуры. Эту информацию можно получить из существующих ГИС, таких как «Яндекс.Карты» и 2Gis. Реальная система должна оперировать большим объёмом данных и доставлять их в визуальном виде.

На основе проделанного анализа потребностей конечного пользователя, можно сделать выводы об основных функциях и особенностях системы, необходимых для успешной информационной поддержки:

* многослойность карты;
* возможность получать об объекте дополнительную информацию: фото, схемы, рисунки, текстовое описание.
* система фильтров отображения объектов;
* удобная система запросов и поиска объектов на карте;
* возможность печать подробной информации об объекте;
* инструментарий для измерения расстояния и площади;
* система визуального анализа выдаваемой информации;

# ОСНОВНЫЕ ПРОБЛЕМЫ ТРАНСПОРТНОЙ СИСТЕМЫ

# ЕКАТЕРИНБУРГА

Сегодня пробки на дорогах Екатеринбурга стали настоящим бедствием: люди часами стоят в автомобильных заторах, опаздывая на работу или на важную деловую встречу, грузы не вовремя попадают на стройки и промышленные предприятия, резко ухудшается экологическая ситуация в городе.

Активным изучением и анализом пробок крупных городов занимается компания «Яндекс». В Екатеринбурге создан филиал аналитического центра, который занимается сбором и обработкой информации о транспортных потоках города. Основные источники информации центра «Яндекс.Пробки» - данные детекторов транспорта и камер наблюдения за транспортными потоками, установленные на сложных развязках города, мобильные треки от пользователей и партнеров, сообщения корреспондентов и т.д. Загруженность дорог измеряется в баллах:

1 – Дороги свободны

2 – Дороги почти свободны

3,4 – Местами затруднения

5 – Движение плотное

6 – Движение затрудненное

7 – Серьезные пробки

8 – Многокилометровые пробки

9 – Город стоит

10 – Лучше ехать на метро

Кроме того, используется комплексный показатель загруженности улиц, учитывающий количество транспортных средств, длину пробок и среднюю скорость движения (по данным видеодетекторов).

Одной из причин транспортных заторов на дороге является наличие так называемых «узких» мест.

«Узкие» места — такие участки дорог, движение на которых замедляется даже при относительно небольшом количестве автомобилей. Они возникают по самым разным причинам — из-за сужения трассы, сложных развязок и перекрёстков, где сливаются несколько потоков машин, парковки автомобилей вдоль трассы, плохо настроенных светофоров и т.д. Именно с таких участков начинаются пробки в часы пик — из-за того, что на них падает скорость автомобилей. Такие места ещё называют «бутылочными горлышками».

К сожалению выявлением таких мест на дорогах Екатеринбурга открыто никто не занимается. Более пристально этой проблеме уделяют внимание в Москве, где проблема пробок более актуальна и остра. Рассмотрим на примере Моссквы несколько примеров «узких» мест, опубликованных компанией «Яндекс»:

1. Часть дороги занимает опора моста. Сразу после моста есть разворот в сторону области, на котором скапливаются машины.

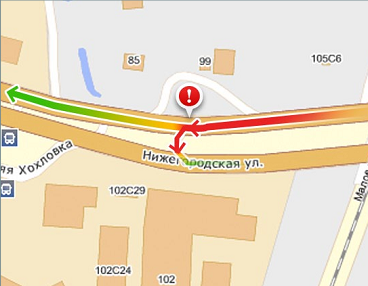


Рис.8 «Узкое» место на дороге, пример №1 [7]

1. Каширское шоссе вливается в Варшавское, при этом семь полос сливаются в пять. Через триста метров после этого расположен пешеходный светофор.

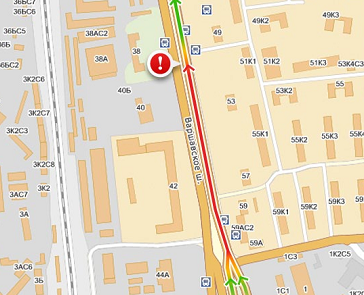


Рис.9 «Узкое» место на дороге, пример №2 [7]

1. На светофоре часто скапливается большой поток желающих повернуть налево. При этом карман для поворота слишком короткий — в результате автомобили занимают левую полосу перед карманом и полосы для движения прямо непосредственно перед перекрёстком.



Рис.10 «Узкое» место на дороге, пример №3 [7]

Компанией «Яндекс» обладает огромным опытом в создании качественных сервисов информационной поддержки пользователей. Каждый год аналитические центры «Яндекс.Пробки» публикуют годовой отчет по состоянию пробок на дорогах крупнейших городов. Статистическая информация, собираемая аналитическими центрами, находятся в открытом доступе и может быть использована в разрабатываемой ГИС.

# ОПИСАНИЕ ПРОЕКТА СИСТЕМЫ

## 5.1. Описание электронной карты

Как уже говорилось, основа любой ГИС – это электронная карта, состоящая из различных слоёв. Основные преимущества такого подхода были описаны выше в главе «Послойная организация данных». Поэтому, одним их основных этапов разработки системы было создание слоёв электронной карты Екатеринбурга. Ниже опишем основные слои, присутствующие в системе.

* City – самый первый слой, появляющийся при открытии приложения. Представляет из себя слой полиполигонов. Содержит 1 объект – город Екатеринбург. Был вынесен в отдельный слой с различными целями. Во-первых, предоставить пользователю общую информацию о городе (дата основания, общая площадь и население, официальный сайт города и т.д.). Во-вторых, это позволяет в дальнейшем добавлять на карту другие города области ( например города спутники: Березовский, верхняя Пышма и т.д.), не затрагивая при этом другие слои карты.



Рис.11 Слой «City»

* Area – слой районов города. Состоит из 7 полигональных объектов: Ленинский, Верх-Исетский, Железнодорожный, Орджоникидзевский, Кировский, Октябрьский и Железнодорожный районы. При отображении слоя можно выбрать один из районов и двойным щелчком мыши открыть информацию о нем. Район – немаловажный фактор на рынке торговли недвижимостью, часто являющийся основополагающим при принятии решения. Для визуального анализа объектов слоя, на данном этапе разработки системы, были выбраны такие параметры как население и плотность населения. Безусловно, такой анализ будет полезен для реальных пользователей лишь при решении очень узкого круга задач, но, к сожалении, на данном этапе разработки мы не обладаем какими-то конкретными статистическими данными, которые можно было бы использовать в системе.



Рис.12 Слой «Area»

* MainRoads, BigRoads, Roads – слои дорожной сети Екатеринбурга, появляющиеся при разном масштабе карте. Каждая улица представляет из себя ломаную полилинию, каждый слой – набор таких полилиний. Для визуального анализа используются такие параметры как уровень пробок и состояние дорожного полотна. Данные по пробкам были получены с помощью сервиса «Яндекс. Карты», поэтому реально отображают ситуацию с дорожной сетью в городе. Для примера был воспроизведен участок Ленинского района. Данные о состоянии дорог были получены опытным путем и являются относительными.
* Water – слой водоемов города: пруды, реки, озёра и т.д. объекты представлены полигонами. Визуальный анализ отсутствует.



Рис.13 Слои «MainRoads», «BigRoads», «Water»

* Buildings – слой зданий. Безусловно, воспроизвести все объекты в рамках разработки нашей системы не возможно. Поэтому был выбран участок Ленинского района, на карту которого были нанесены дома и другие здания. Объекты слоя – полигоны. Для анализа были выбраны такие параметр как: средняя цена за 1кв.м., этажность здания, количество проживающих людей.
* Icons – точечный слой различных объектов карты, имеющий потенциальный интерес для пользователя системы. На данный момент добавлены такие объекты как знаки дорожного движения, детские площадки во дворах, спортивные объекты и т.д. Каждому объекту (точке) соответствует запись атрибутивной таблицы, в которой хранится тип объекта, в зависимости от которого ему присваивается пиктограмма. Эта пиктограмма отображается на карте с центром в соответствующей точке. Слой может содержать сколь угодно много объектов различного типа, которые могут быть так же разделены по группам, и в зависимости от установленных фильтров отображаться на карте.

**5.2. Описание интерфейса программы**

Основное окно программы изображено на рисунке ниже.

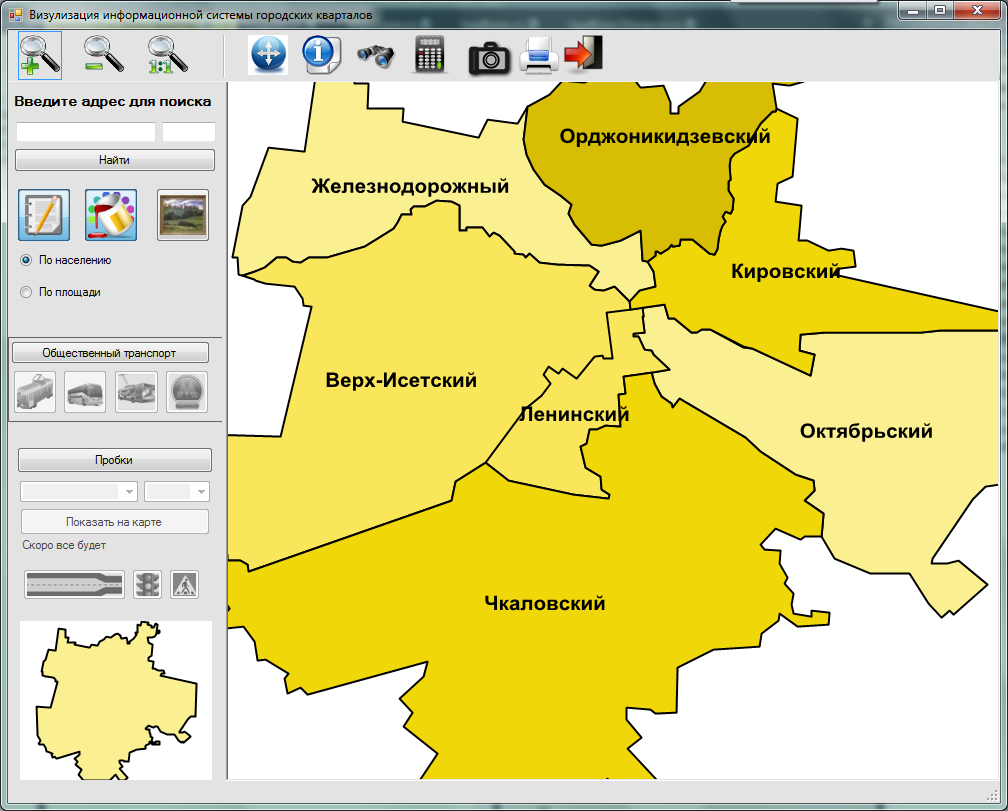


Рис.14 Интерфейс основного окна программы

Оно состоит из трёх основных частей. В основной части находится электронная карта, сверху – панель инструментов для работы с ней, слева – панель для настройки отображения объектов карты (подписи, визуальный анализ данных, фильтры). Структура электронной карты была описана выше, поэтому перейдем к описанию панелей инструментов и настроек.

Верхняя панель содержит следующие инструменты для работы с картой:



Рис.15 Панель инструментов

* «Приближение» – один из основных элементов управления картой. В системе используется, так называемое, семантическое масштабирование данных, то есть при определенном масштабе отображаются определенные слои электронной карты. В данном режиме работы нажатие левой кнопки мыши – приближение, правой – отдаление.
* «Отдаление» – режим работы с картой, аналогичный ZoomIn, отличающийся только назначением кнопок мыши: нажатие левой – отдаление, правой – приближение.
* «Начальный масштаб» – при нажатии карта возвращается к ее изначальному масштабу.
* «Перемещение» – режим перемещения карты. Для того, чтобы переместить карту, необходимо зажать левую кнопку мыши и двигать карту в нужном направлении.
* «Выбор объекта» – режим выбора объектов карты. Для просмотра подробной информации необходимо два раза нажать левой кнопкой мыши на объекте. При этом объект на карте выделяется цветом.
* «Поиск» - инструмент для поиска объектов на карте по заданным пользователям параметрам.
* «Измерить» - инструмент для измерения расстояния и площади на карте.
* «Фото» - режим работы с картой, превращающий ее в настоящую фото-галерею.
* «Печать» - инструмент для печати электронной карты.
* «Выход» - закрытие всех окон программы.

Инструменты по работе с картой будут подробно описаны в следующих главах.

Рассмотрим теперь левую панель программу. С помощью ее можно управлять различными свойствами отображения карты, осуществлять поиск домов по адресу, просматривать маршруты общественного транспорта, анализировать транспортную сеть города и т.д.

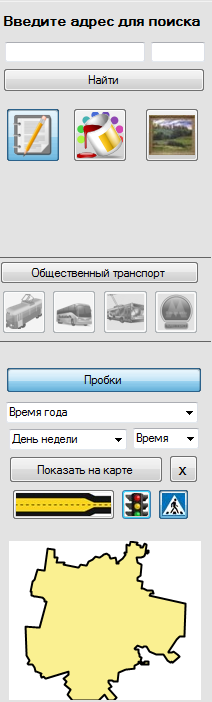


Рис.16 Панель управления отображения карты

Управление отображением карты включает в себя:

* «Подписи» - отображение на электронной карте подписей объектов. Это могут быть названия районов, номера домов и т.п.
* «Визуальный анализ» - визуализация объектов карты разными цветами, в зависимости от выбранного ниже параметра. Для объектов разных типов используются различные параметры визуализации.
* «Пиктограммы» - отображение на карте различных пиктограмм (дорожные знаки, светофоры, магазины, спортивные сооружения и т.д.)

При нажатии на кнопку «Пиктограммы» открывается окно выбора типов отображаемых объектов. При нажатии ан кнопку «Показать» они будут показаны на карте.

Панель «Общественный транспорт» позволяет включать на карте отображение маршрутов и остановок общественного транспорта Екатеринбурга. Включает в себя:

* Трамваи
* Автобусы
* Троллейбусы
* Метро

На данный момент разработаны методы отображения слоев такого типа, но сами слои находятся в разработке.

Панель «Пробки» показывает загруженность транспортной сети в различные дни недели в определенные промежутки времени. Так же имеется возможность отображения на карте «узких» мест на дорогах, светофоров и пешеходных переходов. Более подробно данный инструментарий будет описан в следующих главах.

Внизу панели расположена обзорная карта, позволяющая понять в какой области карты находится пользователь в данный момент.

## 5.3. Информация об объекте

При выборе определенного объекта в режиме работы с картой «Выбор объекта» можно просмотреть подробную информацию о нем в открывшемся окне. Как видно, оно состоит из трех частей:

* атрибутивная таблица объекта,
* изображение,
* текстовое описание.

Рассмотрим каждую из них подробнее.

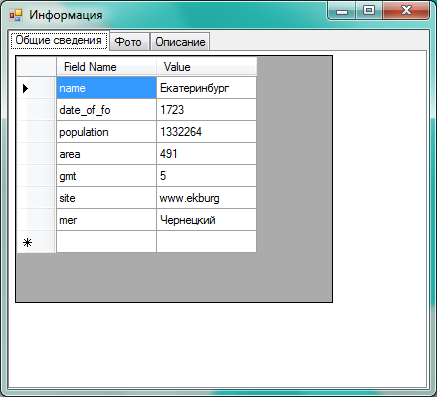


Рис.17 Атрибутивная таблица объектов

Атрибутивная таблица данных объекта хранится в файле с атрибутивной информацией векторного слоя. В зависимости от объекта может содержать различное число строк. Решение представлять информацию в виде таблицы связано с идеей дальнейшего развития системы и предоставления пользователю возможности редактировать данные об объекте и сохранять эти изменения.

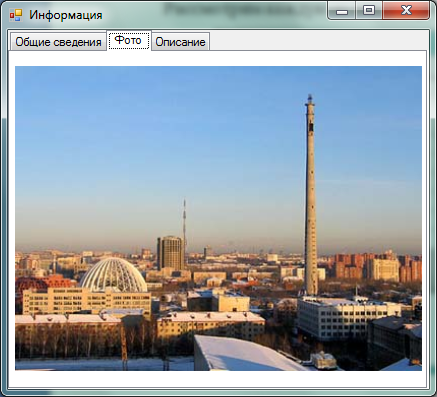


Рис.18 Изображение объекта

Изображение объекта загружается из файла, путь к которому указан также в атрибутивной таблице объекта, но не отображается пользователю. Все изображения на данный момент хранятся в папке Images.

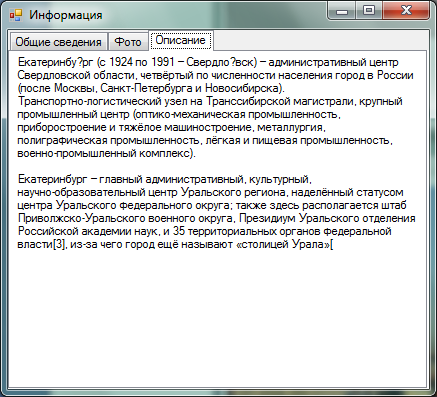


Рис.19 Текстовое описание объекта

Текстовое описание объекта так же хранится в файле, путь на который указан в атрибутивной таблице. Все текстовые файлы хранятся в папке Texts.

## 5.4. Описание средств поиска объектов на электронной карте

Основными объектами нашей электронной карты, представляющие наибольший интерес для потенциального пользователя, безусловно, являются жилые дома, административные постройки, офисные здания. У всех этих объектов есть свой адрес, который уникально идентифицирует их на карте города. Для нахождения этих объектов был реализован их поиск по адресу. Панель поиска изображена на рисунке ниже.

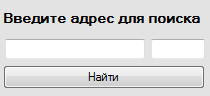
****

Рис.20 Панель поиска объекта по адресу

Как и в большинстве ГИС достаточно просто ввести улицу и номер дома и нажать кнопку «Найти». Если объект с таким адресом был найден, то карта автоматически центрируется на нем и увеличивается (или уменьшается) до нужного масштаба. При этом объект будет выделен цветом. Если объект на карте не найден, то появится информационное сообщение об этом, после которого можно попытаться найти его еще раз и дальше продолжить работу в системе.

Каждый объект обладает атрибутами, которые хранятся в атрибутивной таблице слоя карты. Основной функцией нашей системы является фильтрация и поиск объектов недвижимости, соответствующим определенным параметрам. Некоторые из параметров можно определить использовав саму карту. Например, район, в котором находится объект, близость к водоемам и паркам, относительное расположение относительно других объектов и т.д. Все эти нюансы можно увидеть на карте с помощью средств информационной визуализации. Но есть параметры, которые нельзя напрямую отобразить на карте. Это может быть тип объекта (жилая квартира или офисное помещение), его стоимость, тип недвижимости (продается или сдается в аренду) и т.д. Таких параметров может быть очень много. При этом для одних пользователей они могут иметь высокий приоритет при выборе недвижимости, а для других не иметь никакого значения. Большинство ГИС имеют предустановленный набор фильтров и параметров, по которым можно осуществлять поиск. При этом в основном поиск идет только объектов недвижимости. Что делать, если пользователь добавляет в атрибутивную таблицу параметр, которого не было раньше и хочет осуществлять поиск именно по нему? Как найти и показать на карте все круглосуточные магазины в Ленинском районе? Для данных целей был разработан более сложный инструментарий создания запросов, фильтрации и поиска объектов на карте.

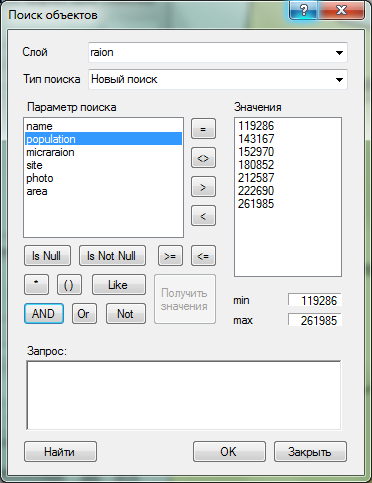
****

Рис.21 Окно создания запросов и поиска объектов

Подробно рассмотрим окно поиска и алгоритм создания запроса.

1. Список «Слой» содержит все загруженные на карты слои. Неважно, отображается на данный момент на карте этот слой или нет.
2. Список «Тип поиска». Реализованы следующие типы поиска:

* Новый поиск
* Добавить к найденному
* Удалить из найденного
* Поиск в найденном

1. Окно «Параметр поиска» позволяет выбрать параметр (или несколько параметров) атрибутивной таблицы, по которому будет осуществляться поиск.
2. Окно «Значения» отображает все уникальные значения выбранного параметра. Окно становится активным после нажатия кнопки «Получить значения». При выборе другого параметра и повторном нажатии этой кнопки – окно «Значения» обновляется. Ниже приведены максимальное и минимальное значения выбранного параметра.
3. Набор кнопок для построения логических выражений
4. Окно «Запрос» отображает сам запрос и позволяет его редактировать вручную.

Простой пример и результат запроса показаны на рисунке ниже.

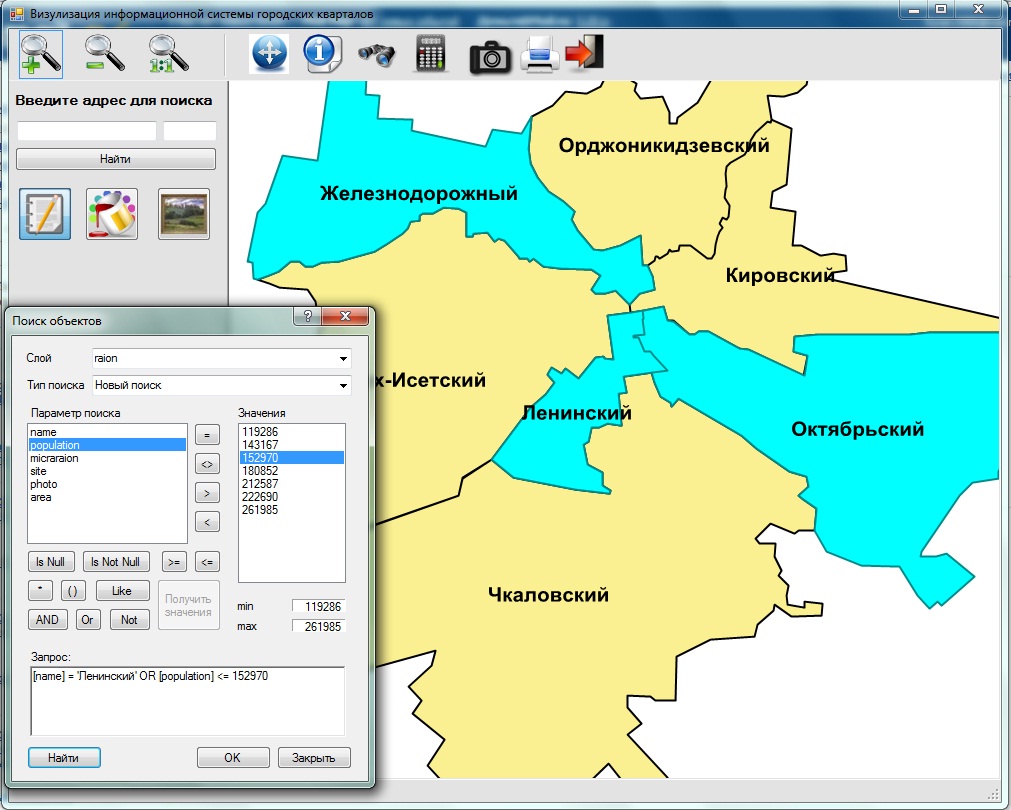
****

Рис.22 Пример поиска объектов по заданным параметрам

**5.5. Инструменты измерения расстояний и площади объектов**

Карта дает четкую информацию о расположении объектов относительно друг друга и их размере. Но часто этого бывает недостаточно и требуются точные данные. В таких случаях можно воспользоваться инструментарием для измерения расстояний и площади объектов.

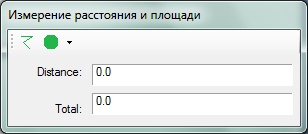
****

Рис.23 Окно измерения расстояний и площадей

Выделим основные направления, в которых может применяться данный инструментарий:

1. Измерения расстояния между объектами (расстояние от дома до магазина, больницы, школы и т.п.).

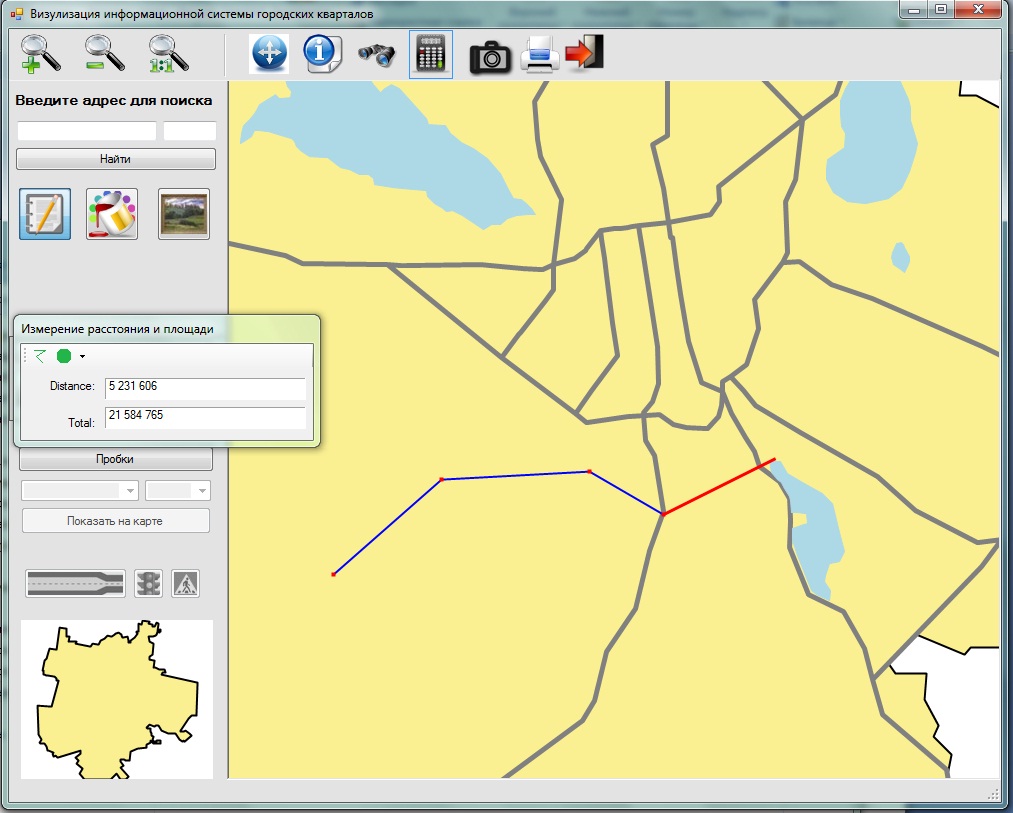


Рис.24 Пример измерение расстояния

1. Измерение площади прилегающей территории .

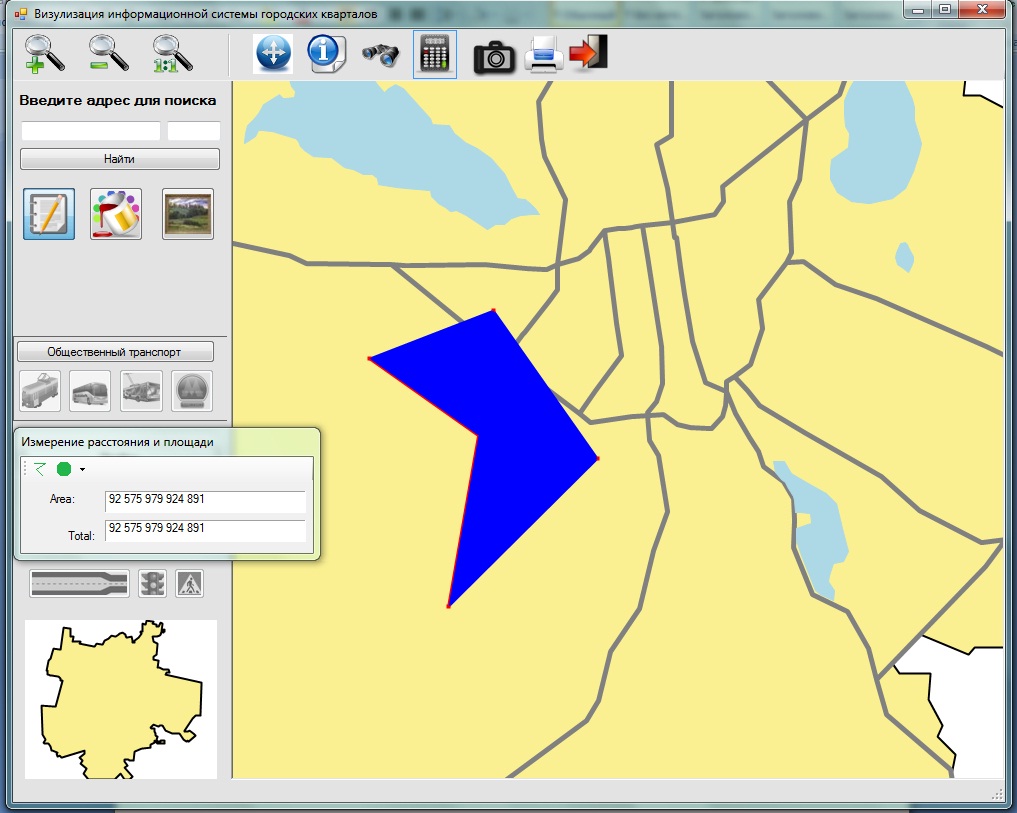


Рис.25 Пример измерения площади

## Инструменты анализа транспортной системы города. «Узкие» места

## на дорогах Екатеринбурга

Как уже говорилось ранее, система должна дать инструмент анализа, как для специалистов купли/продажи недвижимости, так и для представителей муниципальных органов власти. Ранее уже были описаны возможности применения ГИС в различных сферах современного общества. Особое внимание следует уделить проблеме пробок на дорогах крупных городов. ГИС идеально подходит для визуализации и анализа транспортной сети города.

Как уже было сказано выше, наиболее активно изучением пробок в крупных городах занимается «Яндекс». Сервисом «Яндекс.Пробки» пользуются тысячи автомобилистов. «Яндекс.Пробки» показывают пользователям картину загруженности дорог. Для этого сервис собирает из разных источников данные о загруженности улиц, анализирует их и отображает на «Яндекс.Картах». В наиболее крупных городах, где пробки — серьезная проблема, а не просто неприятность, сервис рассчитывает балл пробок — средний уровень загруженности.

Стоит отметить, что сервис «Яндекс.Пробки» в первую очередь создан для предоставления актуальной информации автомобилистам о загруженности дорожного движения и ДТП, которые, безусловно, могут парализовать транспортный поток. Нашей задачей было воспользоваться информацией, полученной из данного сервиса, и попытаться создать инструментарий для анализа дорожной сети Екатеринбурга. В качестве предмета исследования были взяты крупнейшие улицы города.

Подробно рассмотрим панель «Пробки», которая дает пользователю системы возможность оценивать и анализировать не только пробки, но и причины их возникновения.

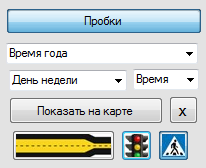
****

Рис. 26 Панель Пробки»

Пробки на дорогах могут появляться по многим причинам. Перечислим основные из них:

* ДТП. Может парализовать движение по нескольким улицам одновременно
* Ремонтные работы. Закрытие на ремонт небольшого участка дорожного полотна может серьезно сказаться на движении транспортного потока.
* Наличие «узких мест» .
* Поломка светофора или его некорректная работа. Отсутствие регулировщиков в таких случаях может создать большие заторы на небольших улицах.
* Отсутствие парковочных мест, большое количество торговых центров и магазинов приводит к сужению проезжей части.

Некоторые факторы не поддаются анализу (например ДТП), поэтому пробка может возникнуть совершенно в неожиданном месте в любое время дня. Но все-таки проанализировать большинство причин можно.

Для начала, необходимо выбрать сезон, день недели и временной промежуток, интересующий пользователя. На данный момент выбор осуществляется между выходными днями и будними, реализовано несколько временных промежутков. При нажатии на кнопку «Показать на карте» загруженность дорог будет показана на карте. При этом улицы разбиваются на участки, уровень пробок показывается для движений в обе стороны.

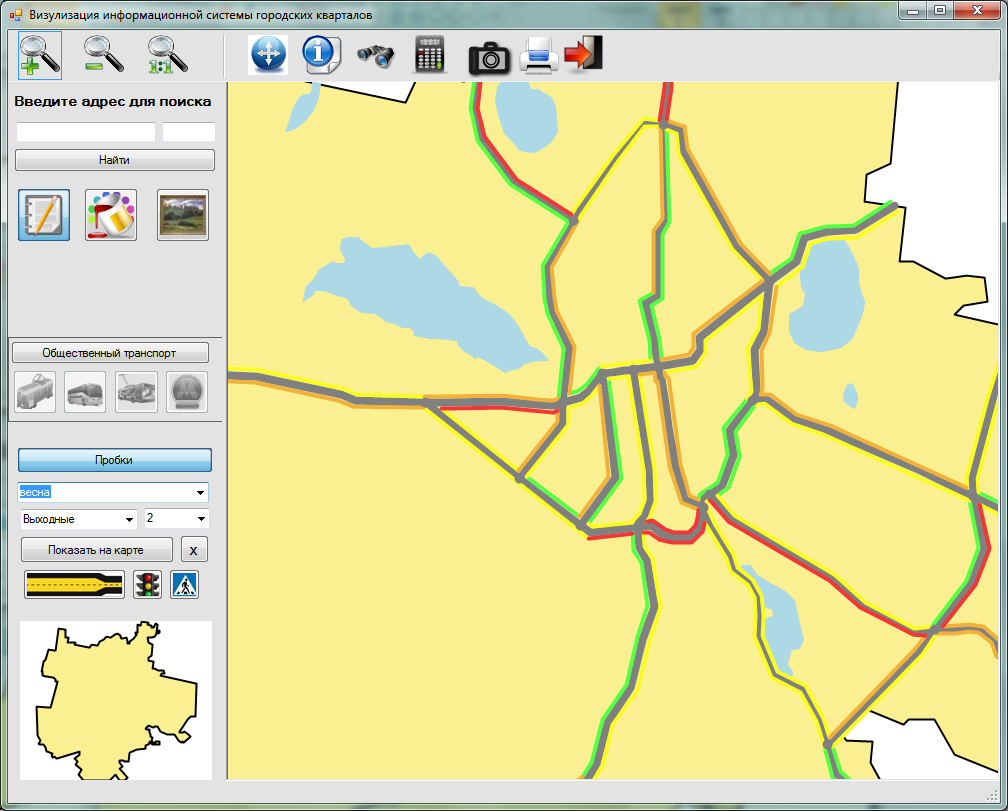


Рис.27 Визуализация пробок

Но данная информация о заторах не дает никакой информации о причинах их возникновения. В этом вопросе на помощь может придти визуализация дорожной сети по таким атрибутам как :

* Количество полос в одном направлении
* Качество дорожного полотна

Включение визуализации объектов и выбор интересующего параметра осуществляется вверху панели с помощью кнопки «Визуальный анализ».

Кроме того, на панели «Пробки» можно включить отображение на карте следующих объектов:

* «Узкие» места

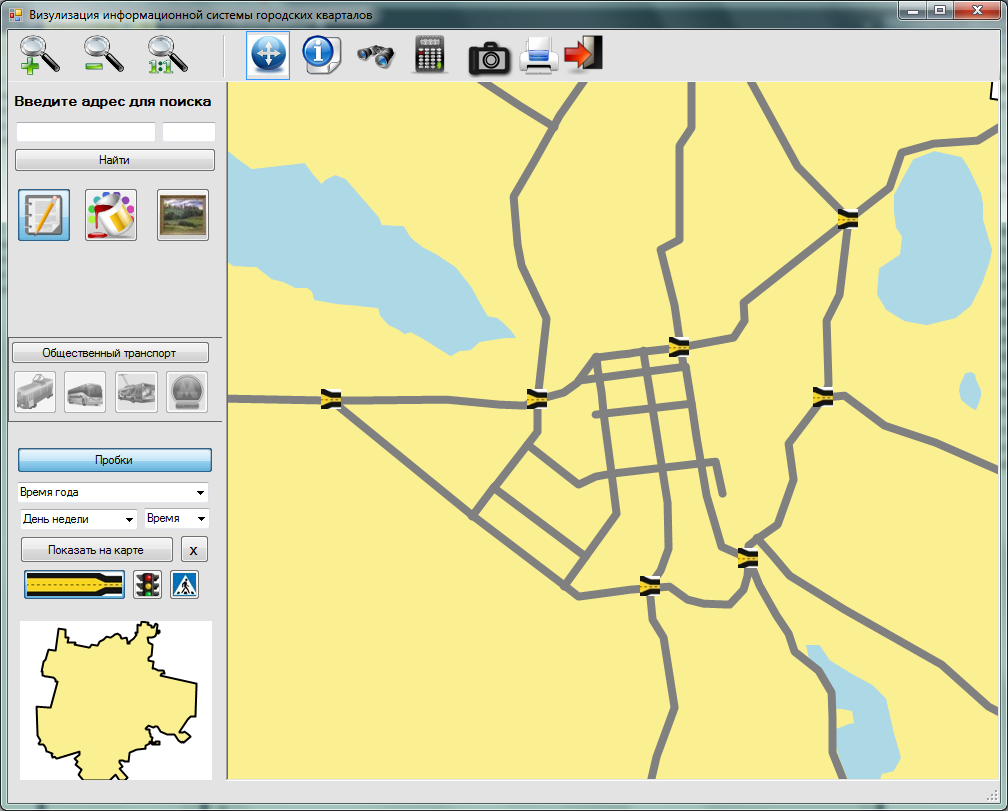


Рис.28 Отображение «узких» мест

* Светофоры

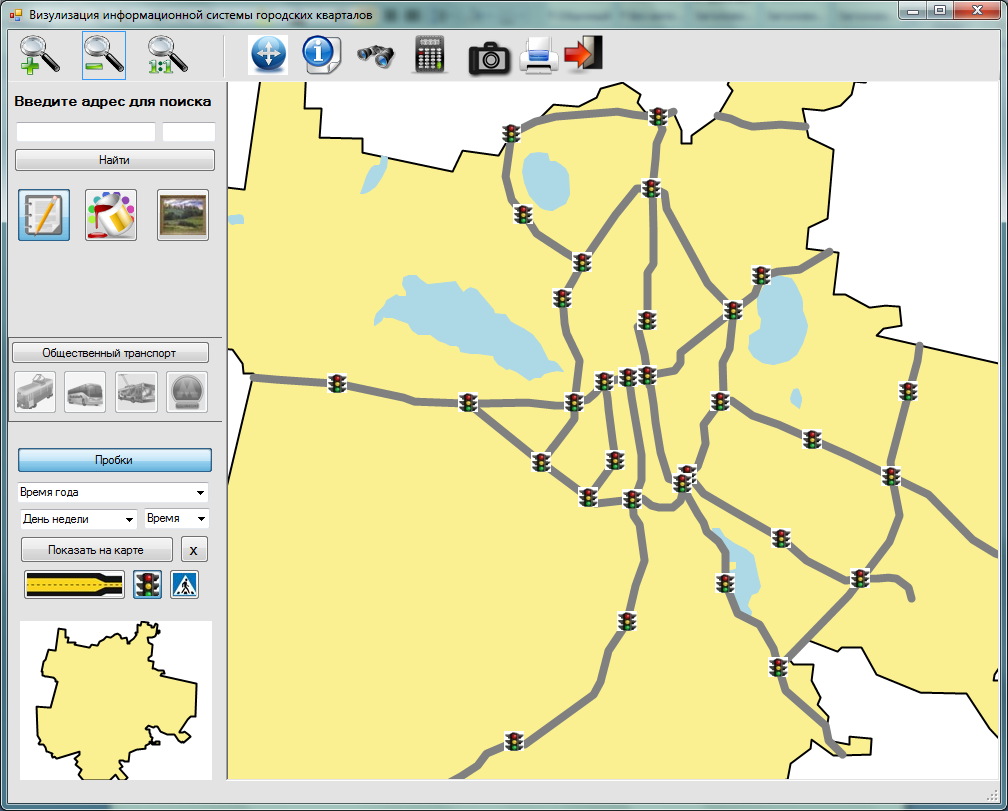


Рис. 29 Отображение на карте светофоров

* Пешеходные переходы и пешеходные светофоры.

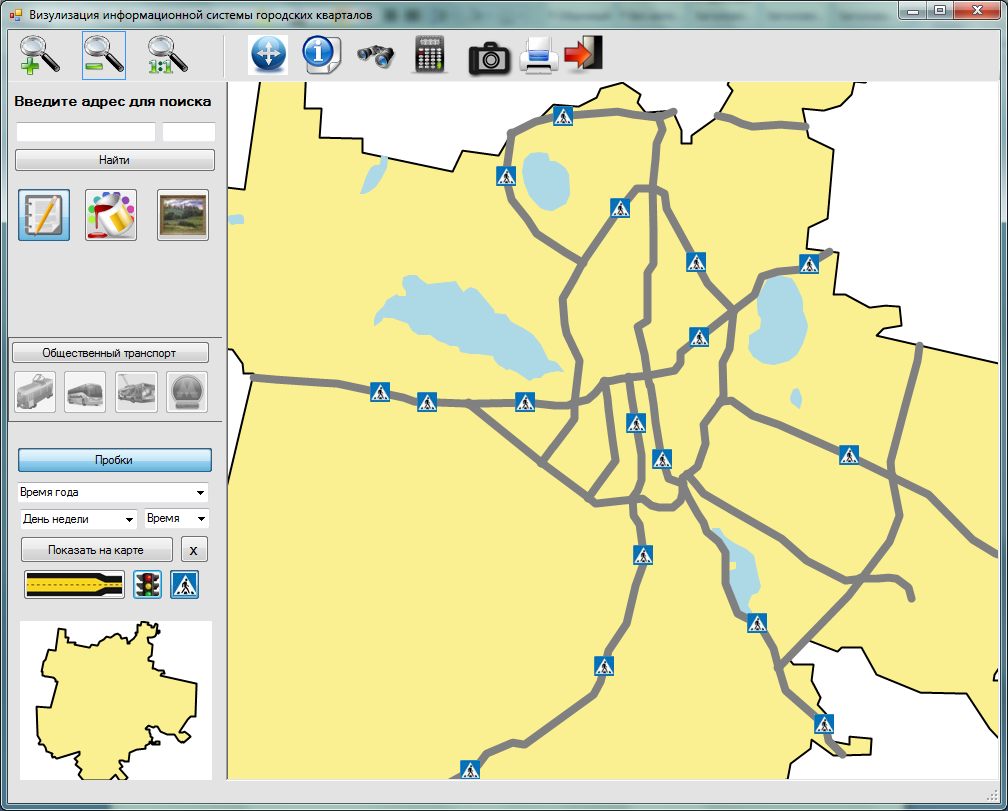


Рис. 30 Отображение на карте пешеходных переходов

**5.7.Средства печати**

Важной частью любой информационной системе является система вывода. Часть современных ГИС в качестве системы вывода используют только монитор компьютера, но часто этого бывает недостаточно. Разрабатываемая нами система должна давать возможность не только просмотра электронной карты, но и ее сохранения или печати. Инструмент для переноса карты на другие носители (бумага, файл) должен дать потенциальному пользователю системы возможность сохранять и сравнивать отдельные участки инфраструктуры города, передавать эту информацию другим людям.

Подробно рассмотрим окно печати.

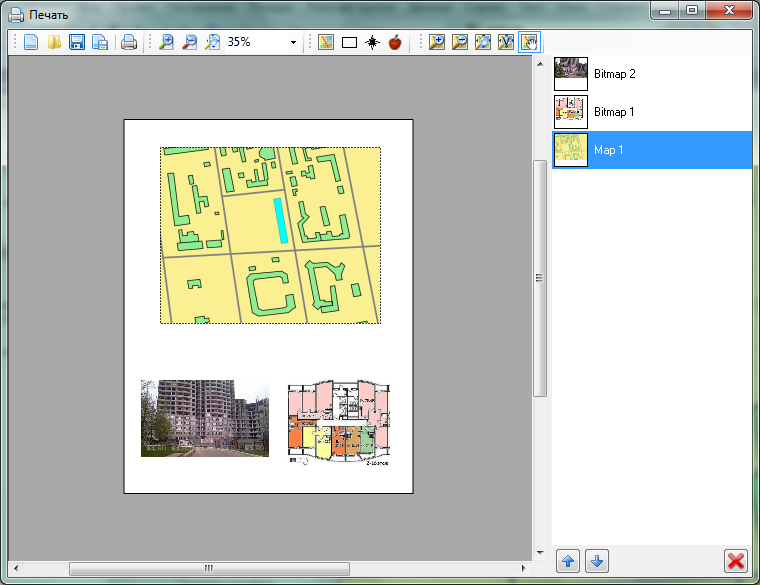
****

Рис. 31 Окно печати

Окно печати состоит из трех основных частей:

1. Панель управления печатной формой. Включает в себя следующие элементы управления:



Рис. 32 Панель управления печатью

* Новая печатная форма
* Открыть печатную форму
* Сохранить печатную форму
* Сохранить печатную форму как
* Печать
* Увеличить
* Уменьшить
* Изначальный размер
* Выбор масштаба
* Добавить карту
* Добавить рамку
* Добавить указатель
* Добавить изображение
* Увеличить масштаб добавленной карты
* Уменьшить масштаб добавленной карты
* Изначальный масштаб карты
* Максимальный масштаб карты
* Переместить карту

1. Печатная форма. На печатную форму с помощью панели управления можно добавить участок электронной карты, отображаемый в данный момент на ней. После добавления карты ей так же можно управлять с помощью специальных элементов (изменять масштаб, перемещать). Одновременно можно добавлять несколько участков карты, перемещать их по печатной форме. Это позволяет пользователю системы сравнивать объекты инфраструктуры. Кроме этого, на карту можно добавить изображение объекта из внешнего файла.
2. Панель объектов. В списке отображаются все элементы, которые добавлены на карту на данный момент (участки карт, рисунки, рамки). Объекты можно перемещать в списке, при этом их отображение на печатной форме при наложении друг на друга будет меняться. Выбранный объект можно удалить. Если на форму добавлено несколько карт, то данный список служит для переключения между ними.

Стоит отметить, что правильно сформированная печатная форма может дать полное представление об объекте недвижимости.

# ПРИНЦИПЫ ПОСТРОЕНИЯ ЭФФЕКТИВНЫХ ВИДОВ

# ОТОБРАЖЕНИЯ ИНФОРМАЦИИ

Одной из важных задач при выполнении работы являлось построение качественного вида отображения информации на карте. В работах по графическому дизайну описан ряд общих принципов компоновки графических средств представления информации. Среди них:

1. **Принцип лаконичности.**

Графическое средство представления информации должно содержать лишь те элементы, которые необходимы для сообщения наблюдателю существенной информации, точного понимания её значения или принятия соответствующего оптимального (в каком-то смысле) решения. Бесполезно стремиться направить внимание на важнейшие характеристики, если они окружены лишними, не относящимися к ним визуальными раздражителями, мешающими восприятию главного.

1. **Принцип обобщения и унификации.**

Основные формы графического средства представления информации не следует излишне дробить, включая в них элементы, обозначающие несущественные с точки зрения отображаемой информации детали изображаемых объектов; их форма должна быть рационально обобщена. Кроме этого, в пределах всего комплекса графических средств представления информации образы, использованные для обозначения одних и тех же объектов, должны быть обязательно унифицированы - иметь единое графическое решение.

1. **Принцип акцента на основных смысловых элементах.**

На графических средствах отображения информации следует выделять размерами, формой, цветом в первую очередь те элементы, которые наиболее существенны с точки зрения восприятия наблюдателем передаваемой информации. В отдельных случаях допустимо даже нарушение пропорций между размерами образов и соответствующих им реальных объектов.

1. **Принцип автономности.**

Части графического средства представления информации, передающие относительно автономные сообщения следует обособить и чётко отграничить от других частей. Разбиение сложной графической информации на отдельные простые изображения значительно облегчает её восприятие и понимание.

1. **Принцип структурности.**

Каждая автономная часть комплекса графических средств отображения информации, занимающая в общем изложении некоторое центральное положение, должна иметь чёткую, легко запоминающуюся и дифференцирующуюся от других структуру, отражающую характер каждого сообщения.

1. **Принцип стадийности.**

В зависимости от стадий изложения информации должен выбираться состав сообщений, отображаемых на отдельных графических средствах. Этот принцип основан на методах борьбы с вредной, лишней в конкретном случае информацией путём пространственного и временного разделения всей информации и её последовательного восприятия.

1. **Принцип использования привычных ассоциаций и стереотипов.**

При создании графических средств представления информации должны учитываться устойчивые, привычные ассоциации между образами и обозначаемыми ими объектами и явлениями. Желательно применять не абстрактные условные знаки, а образы, привычно ассоциирующиеся с соответствующими объектами и явлениями. Однако следует учитывать, что слишком натуралистичное изображение внешнего вида объектов фиксирует мысль наблюдателя именно на внешнем сходстве с объектом и мешает осознанию более существенных с точки зрения представляемой информации признаков данного объекта.

Существует требование строгого соответствия композиционного решения, формы графического информационного средства его функциональному назначению, содержанию отображаемой информации. При создании композиции существенно важны такие приёмы её гармонизации, как поиск равновесия (симметричные и ассиметричные композиции), метр и ритм. Важными композиционными факторами являются обрамление изобразительной поверхности, контраст и нюанс. Одной из важных композиционных оценок является масштабность графического средства представления информации.

Форма, пространство и визуальное взаимодействие являются средствами для визуализации идей. Визуальная идея, дизайн и исполнение - три главных этапа в процессе визуализации.

Все эти принципы использовались в ходе проектирования нашей геоинформационной системы при дизайне слоёв электронной карты и разработки интерфейса программы.[8]

# 7. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ

Данная глава описывает средства разработки ГИС и объясняет, чем был обоснован этот выбор. Итак, основным языком разработки был избран язык программирования С# и платформа .NET, средой программирования была выбрана Visual Studio 2010. По моему мнению, объединение лучших идей современных языков программирования (Java, C++, Visual Basic и др.) делает язык C# не просто суммой их достоинств, а языком программирования нового поколения.

## Формат карт shape

В данной главе описывается формат пространственных данных shape-файлов, а также упоминаются инструменты, которые могут быть использованы для создания и работы с shape-файлами.

В системе векторные нетопологические данные представляются в формате shape-файлов геоинформационной системы ArcView фирмы ESRI. Shape-файлы бывают трёх основных типов:

* Файл полиполилиний. Полиполилиния – это фигура, состоящая из одной или более полилиний.
* Файл полиполигонов. Полиполигон – это фигура, состоящая из одного или более полигонов.
* Файл мультиточек. Мультиточка – это фигура, состоящая из одной или более точек.

Все эти типы имеют по три подтипа в зависимости от размерности пространства, в котором задаются объекты. В простых shape-файлах точки задаются в двумерном пространстве в виде пары чисел (*X*,*Y*), в shape-файлах с мерой точки задаются в трёхмерном пространстве в виде тройки чисел (*X*,*Y*,*M*), в shape-файлах с мерой и высотой точки задаются в четырёхмерном пространстве в виде четверки чисел (*X*,*Y*,*Z*,*M*).

В shape-файлах хранится нетопологическая геометрическая и атрибутная информация пространственных объектов в базах данных. Геометрия составляющих фигур представляется как набор плоских координат.

В связи с тем, что shape-файлы не имеют топологических ограничений на взаимные связи, они имеют определённые преимущества над другими источниками данных, что в первую очередь проявляется в высокой скорости отрисовки и простоте редактирования. Все фигуры shape-файла полностью независимы друг от друга, в частности, они могут произвольно пересекаться между собой и даже совпадать. Кроме того, нетопологический формат shape-файлов обычно требует меньше дискового пространства по сравнению с топологическими форматами, а также он проще для чтения и записи.

Shape-файлы могут хранить точечные, линейные и площадные фигуры. Смешение разных типов фигур в одном файле не допускается. Атрибуты файлов содержаться в формате файла таблиц dBase®. Каждая атрибутная запись хранится в отношении «один-к-одному» с соответствующей записью геометрии фигуры.

В настоящее время формат shape-файлов широко поддерживается многими программными продуктами фирмы ESRI и других производителей. Shape-файлы используются в таких системах, как продукты ESRI: ArcInfo® GIS, ArcSDE ™ (Spatial Database Engine), ArcView ® GIS, Business Map, а также других производителей: MapInfo, Autodesk AutoMap, AutoCAD Map, Autodesk World, EASI/PACE, TNTmips, MapEdit и др.

Формат shape-файла ESRI состоит из главного файла, файла индекса, а также таблицы dBase. Все эти 3 файла должны иметь одинаковое имя, но различные расширения: .shp для главного, .shx для индексного и .dbf для файла атрибутов. Например, shape-файл, содержащий информацию о городах, должен состоять из файлов cities.shp, cities.shx и cities.dbf.

В shape-файлах используются два типа данных хранения информации: целые и вещественные, имеющие следующий формат:

* Целые: знаковые 32-разрядные целые числа, занимающие 4 байта.
* Вещественные: знаковые 64-разрядные вещественные числа двойной точности с плавающей запятой в стандарте IEEE, занимающие 8 байт.

В shape-файле используются два способа кодирования этих чисел, отличающихся порядком расположения байтов чисел. На платформе Intel® используется порядок от младшего байта к старшему, а на платформах Sun®, Motorola®, наоборот, – от старшего к младшему.

В ходе разработки нашей системы, мной были созданы различные shape-файлы, представляющие из себя слои электронной карты Екатеринбурга. Многие объекты, такие как например дорого, были разделены по слоям в зависимости от их значимости и других факторов.

Shape-файлы создавались в программе QGis, предназначенной для работы и создания электронных карт. Основные слои были описаны в главе «Описание проекта системы»

## Краткий обзор QGis

QGIS позволяет использовать большое количество распространенных ГИС функций, обеспечиваемых встроенными инструментами и модулями.

Рассмотрим две основные функции QGis, активно используемые при создании нашей системы:

1. **Просмотр данных**

Можно просматривать и накладывать друг на друга векторные и растровые данные в различных форматах и проекциях без преобразования во внутренний или общий формат. Поддерживаются следующие основные форматы:

* Пространственные таблицы PostgreSQL с использованием PostGIS, векторные форматы, поддерживаемые установленной библиотекой OGR, включая shape-файлы ESRI, MapInfo, SDTS (Spatial Data Transfer Standard) и GML (Geography Markup Language)
* Форматы растров и графики, поддерживаемые библиотекой GDAL (Geospatial Data Abstraction Library), такие, как GeoTIFF, Erdas IMG, ArcInfo ASCII Grid, JPEG, PNG

1. **Управление данными: создание, редактирование и экспорт**

В QGIS можно создавать и редактировать векторные данные, а также экспортировать их в разные форматы. Чтоб иметь возможность редактировать и экпортировать в другие форматы растровые данные, необходимосначала импортировать их в GRASS. QGIS предоставляет следующие возможности работы с данными, в частности:

* инструменты оцифровки для форматов, поддерживаемых библиотекой OGR, и векторных слоев GRASS
* создание и редактирование shape-файлов и векторных слоев GRASS
* геокодирование изображений с помощью модуля пространственной привязки
* визуализация и редактирование данных OpenStreetMap
* создание слоёв PostGIS из shape-файлов
* обработка слоёв PostGIS
* управление атрибутами векторных данных с помощью таблицы атрибутов
* сохранение снимков экрана как изображений с пространственной привязкой

## Краткий обзор С#

C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как основной язык разработки приложений для платформы Microsoft .NET. Компилятор с C# входит в стандартную установку самой .NET, поэтому программы на нём можно создавать и компилировать даже без инструментальных средств, вроде Visual Studio. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML. Переняв многое от своих предшественников — языков C++, Java, Delphi, Модула и Smalltalk — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем: так, C# не поддерживает множественное наследование классов (в отличие от C++).

C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем. (Однако эта закономерность была нарушена с выходом C# 3.0, представляющим собой расширения языка, не опирающиеся на расширения платформы .NET.) CLR предоставляет C#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J# и др. [9]

Следующим решением, которое необходимо было принять, был выбор сторонней библиотеки для работы с пространственными географическими данными. Рассматривались два варианта: SharpMap и MapWindow. Так как библиотека SharpMap не поддерживает работы с кириллицей, а разрабатываемая система должна отображать атрибутивные данные на русском языке, то от работы с ней пришлось отказаться. В следующей главе дадим краткий обзор основных элементов и возможностей библиотеки MapWindow, используемых при разработке нашей системы.

## Обзор библиотеки MapWindow

Библиотека MapWindow разрабатывается Geospatial Software Lab [университета штата Айдахо](http://www.isu.edu/). Её основа – элементы управления ActiveX, которые могут быть помещены на форму приложения для отображения, управления и иной работы с пространственными данными. Библиотека разработана для языков C# и VB.NET.

MapWindow поддерживает работу со следующими форматами данных :  
1) векторные: ESRI Shapefile, Postgis (через экспорт в shape-файл);  
2) растровые: ESRI ASCII Grid, GeoTiff, ESRI Grid, ESRI FLT, USGS STDS, PAUX, PIX, DTED, ECW, BIL, MrSID, ArcInfo Grid, ERDAS Imagine img, PNG, JPEG, GIF, EMF, BMP.

Рассмотрим основные классы библиотеки, которые будут часто использоваться в разрабатываемой системе.

* 1. **Описание структуры класса Map**

Объект класса Map библиотеки MapWindow – основа разрабатываемого приложения. Ниже представлены лишь некоторые основные процедуры этого класса, предоставляющие основные возможности работы с электронной картой.

***private void Configure* –** установка свойств карты. Определение доступных функций и режимов работы с картой. Вызывается при создании объекта класса.

***public List<MapLayer> AddLayers*** –добавления слоев на карту. Можно добавлять слои всех типов (векторные и растровые, слои с информацией и подписями). Используется при формировании электронной карты из существующего набора слоёв.

***public MapLineLayer[] GetLineLayers –*** возвращает массив всех векторных линейных слоёв карты.

***public MapPolygonLayer[] GetPolygonLayers*** - возвращает массив всех векторных полигональных слоёв карты.

***public MapPointLayer[] GetPointLayers*** - возвращает массив всех векторных точечных слоёв карты.

***public void AddLabels*** – добавляет на карту слой с подписями (так же возможно добавление через AddLayers). Возможен вывод различных полей атрибутивных таблиц объектов, а так же работа с регулярными выражениями и фильтрами.

***public void ClearLabels*** – очищает карту (все слои) от подписей объектов.

***public void ZoomToMaxExtent***  - возвращение карты в ее изначальный масштаб. В нашей программе вызывается при нажатии на кнопку «Начальный масштаб»

***public FunctionModes FunctionMode –*** устанавливает режим работы с картой.

На основе стандартных, благодаря возможности наследования, в системе были разработаны следующие режимы работы с картой:

* MyZoomInFunction – приближение карты;
* MyZoomOutFunction – отдаление карты;
* SelectInfoFunction – выделение объекта на карте и предоставление информации о нем.
* MyMeasureFunction – измерение расстояний и площадей на карте

Без изменений остался только режим «Перемещение» (PanFunction). Расширение базовых классов связано прежде всего с их ограниченной функциональностью и наличием ошибок в библиотеке MapWindow.

## 7.6.Описание структуры классов слоев

При помощи процедуры AddLayers на карту могут быть добавлены слои различного типа. Векторные слои с географическими данными определяются объектами классов MapPolygonLayer, MapPointLayer, MapLineLayer. Их интерфейсы описаны в классах IMapPolygonLayer, IMapPointLayer, IMapLineLayer, которые являются производными от IMapFigureLayer. Рассмотрим основные процедуры для работы со слоями, использующиеся в системе, на примере класса MapPointLayer, а так же его свойства.

***bool IsVisible*** – определяет видимость загруженного на карту слоя. Используется в системе для отображения нужных слоев при различном масштабе, а так же при использовании фильтров.

***bool SelectionEnabled*** – возможность выбора объектов слоя. При установке значения false слой, по сути, представляет из себя обыкновенное изображение, работа с объектами которого не доступна. Используется в системе для предоставления пользователю возможности работы только с необходимыми слоями, а так же для использования других слоев как фоновое изображение.

***bool EditMode*** – в системе используется для включения/выключения режима отображения визуальной аналитики.

***bool ShowLabels*** – отображение подписей объектов слоя.

***Symbolizer*** – свойтво слоя, задающее параметры визуального отображения объектов: цвет, ширина границ, параметры заливки и т.п. (присваиваются объекты классов PointSymbolizer, LineSymbolizer, PolygonSymbolizer)

# 8. СТРУКТУРНАЯ ОРГАНИЗАЦИЯ ПРОГРАММЫ

## 8.1. Основное окно программы. Форма «MainForm»

В данной главе опишем основные методы, процедуры и классы, используемые при создании основной формы приложения. Подробный листинг будет приведен в Приложении.

**Работа с картой и слоями.**

***private void loadArea -*** процедура добавления на карту слоя с районами города. Для каждого слоя карты написана своя процедура (***loadCity, loadMainStreet, loadBuildings и т.д.)***, так как каждый слой имеет свои особенности отображения. В процедуре устанавливаются различные параметры отображения объектов слоя, такие как цвет, параметры заливки, отображение границ, толщина линий и т.д. Так же загружается атрибутивная таблица слоя, на основе данных которой определяется отображение объектов для визуального анализа. Тут же задаются параметры отображения подписей слоя: параметр который будет отображаться как подпись, свойства шрифта и т.д.

***private void InvisibleLayers –*** процедура, делающая невидимым все слои, добавленные на карты. Т.к. слоев может быть очень много, то для удобства отображения на карте только нужных остальные делаются невидимыми. После вызова процедуры нужные слои делаются видимыми.

***private void selectedFalse –*** процедура, запрещающая выделять объекты всех слоев. После вызова процедуры нужные слои делаются активными для выделения.

***private void map\_click –*** процедура, обрабатывающая нажатие кнопок мыши по карте. В зависимости от выбранного режима работы с картой происходят различные события. Именно здесь происходит анализ масштаба карты и определение слоёв, которые должны отображаться в данный момент.

**Обработка событий верхней панели управления картой.**

***private void cmdZoomIn\_Click –*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Приблизить» на панели инструментов. Переводит карту в режим работы MyZoomInFunction.

***private void cmdZoomOut\_Click -*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Отдалить» на панели инструментов. Переводит карту в режим работы MyZoomOutFunction.

***private void cmdMaxExtents\_Click -*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Начальный масштаб» на панели инструментов. Возвращает карту в начальный масштаб.

***private void cmdPan\_Click -*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Перемещение» на панели инструментов. Переводит карту в режим работы PanFunction.

***private void cmdSelect\_Click-*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Информация об объекте» на панели инструментов. Переводит карту в режим работы SelectInfoFunction.

***private void cmdCalc\_Click-*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Калькулятор» на панели инструментов. Переводит карту в режим работы MyMeasureFunction.

***private void cmdPrint\_Click -*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Печать». Открывает новое окно печатной формы.

***private void cmdFind\_Click -*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Поиск». Открывает новое окно поиска объектов на карте.

***private void cmdExit\_Click-*** процедура, обрабатывающая событие происходящее при нажатие мышью на кнопке «Выход». Закрывает все окна программы.

**Обработка событий левой панели управлением отображения карты**

***private void btnFind\_Click-*** процедура, вызываема при нажатии на кнопку «Найти». Находит объект на карте по введенному адресу, приближает до нужного масштаба.

***private void checkLabels\_CheckedChanged***- процедура, вызываемая при нажатии на кнопку «Подписи». Включает/отключает отображение подписи объектов на карте.

***private void checkVisual\_CheckedChanged-*** процедура, вызываемая при нажатии на кнопку «Визуальный анализ». Включает/отключает визуализацию объектов на карте. При включении открывает панель выбора параметра визуализации.

***rivate void checIco\_CheckedChanged-*** процедуры, вызываемая при нажатии на кнопку «Пиктограммы». Включает/отключает отображение на карте пиктограмм различных объектов. При включении открывается окно выбора групп объектов, которые будут показаны на карте.

***private void tram\_CheckedChanged-*** процедура, вызываемая при нажатии на кнопку «Обществееный транспорт». Включает/отключает отображение на карте маршрутов и остановок общественного транспорта. При включении панель с выбором отображаемых типов транспорта становится активной.

***private void traffic\_CheckedChanged-*** процедура, вызываемая при нажатии на кнопку «Пробки». Включает/отключает отображение на карте объектов, которые выбраны на панели «Пробки». При включении панель становится активной.

## 8.2. Окно информации об объекте. Форма «MyFeatureIdentifire»

В данной главе опишем основные методы, процедуры и классы, используемые при создании формы информации о выбранном на карте объекте.

***public virtual void Add*** – добавление информации о выбранном объекте в окно формы. На данном этапе разработки программы, пользователь имеет возможность только просматривать информацию, не меняя её, поэтому данная процедура является единственной в классе MyFeatureIdentifire (не считая служебных, таких как InitializeComponent, OnClosing и т.п.)

Каждый объект карты связан с записью в атрибутивной таблице слоя. При добавлении слоя на карту изначально добавляются только пространственные данные. Для получения доступа к семантическим данным географических объектов необходимо создать объект , в котором и будет храниться эта информация, загружаемая из файла содержащем атрибутивную таблицу слоя. Такие объекты представлены в системе объектами класса DataSet. Именно объект этого класса используется в классе MyFeatureIdentifire для представления атрибутивных данных в таблице.

## 8.3. Окно печати. Форма «MyLayoutForm»

Основой формы является объект класса LayoutForm библиотеки MapWindow. В этой главе опишем основные процедуры и функции, обеспечивающие полноценную работу данного объекта. Большинство из них реализованы в библиотеке, но некоторые были доработаны и изменены, для обеспечения нужной функциональности.

***public void NewLayout –*** создание новой печатной формы.

***public void LoadLayout –*** загрузка существующей печатной формы, сохраненной ранее.

***public void SaveLayout –*** сохранение текущей печатной формы.

***public void Print –*** Печать текущей формы

***public void ZoomIn –*** Увеличение масштаба печатной формы

***public void ZoomOut –***Уменьшение масштаба печатной формы

***public void ZoomFitToScreen-*** Изначальный масштаб печатной формы

***public void ZoomInMap-*** Увеличение масштаба выбранного фрагмента карты

***public void ZoomOutMap-*** Уменьшение масштаба выбранного фрагмента карты

***public void ZoomFullExtentMap-*** Максимальный масштаб выбранного фрагмента карты, полное отображение слоя.

***public void ZoomFullViewExtentMap-*** Начальный масштаб выбранного фрагмента карты

***public void PanMap-*** Передвижение карты внутри добавленного на форму элемента

**ЗАКЛЮЧЕНИЕ**

В заключении мы хотим описать поставленную нами задачу визуализации информационной системы городской среды и информационной поддержки пользователей. Основной идеей была послойная организация географических и семантических данных на электронной карте. Вывод данных производится в зависимости от масштаба, установленных фильтров и настроек. Разработан прототип программы, реализующий нашу идею. Программа представляет из себя геоинформационную систему с электронной картой части Екатеринбурга и инструментарием по работе с ней. Реализованы следующие возможности:

* Поиск объекта по адресу, а так же поиск групп объектов по заданным параметрам (через систему запросов).
* Измерение расстояния между объектами, их площади.
* Печать фрагментов карты.
* Фильтрация отображающихся объектов на карте.
* Просмотр и анализ транспортной инфраструктуры города (уровень пробок, движение общественного транспорта).

На следующем этапе необходимы консультации со специалистами, занимающимися торговлей недвижимостью, и аккуратное опробование на реальных пользователях. Следует отметить, что данный прототип нуждается в развитии. Отметим основные моменты, требующие доработки:

* Использование реальных географических карт, связь с координатами.
* Возможность пользователю самостоятельно создавать и редактировать объекты.

# ЛИТЕРАТУРА

1. А.А. Питенко – Нейросетевой анализ в геоинформационных системах, диссертация, Красноярск, 2010г.
2. <http://geofizic.ru>
3. [www.dataplus.ru](http://www.dataplus.ru)
4. <http://www.gis-tech.ru/>
5. Андрианов В.- Гис для бизнеса, статься журнала ArcReview №1 (12) 2000г.
6. Андрианов В. – ГИС на транспорте, статься журнала ArcReview №1 (24) 2003г.
7. http://www.yaprobki.ru/
8. Авербух В.Л. – конспект лекций по спецкурсу «Визуализация программного обеспечения», Уральский Государственный Университет им. А.М.Горького, г.Екатеринбург, 2010г.
9. http://ru.wikipedia.org/wiki/C\_Sharp
10. [Белков А.В.](http://www.swsys.ru/index.php?page=infou&id=4478) , Грызлова Т.П. , [Шаров В.Г.](http://www.swsys.ru/index.php?page=infou&id=4480) – Проблемы и методы построения эффективных визуальных информационных систем, статья журнала «Програмные продукты и системы», выпуск №3 за 2002г.
11. Смирнов Н.В. – конспект лекций «Проектирование информационных систем», Балтийский Государственный Технический Университет, С.Петербург, 2008г.
12. **Г**охман В. – Общегородские ГИС, статья журнала [ArcReview № 3 (46) 2008](http://www.dataplus.ru/Arcrev/Number_46/Index.html)г.
13. Скатерщиков С. – ГИС в градостроительном проектировании и управлении территориями, статья журнала ArcReview №3 (18) 2001г.
14. Андрианов В. – ГИС в торговле и сфере услуг, статья журнала [ArcReview № 4 (35) 2005](http://www.dataplus.ru/Arcrev/Number_35/index.htm)г.
15. Luo Yingwei, Wang Xiaolin, Xu Zhuoqun - Gis Components And Map Visualization Objects, Dept. of Computer Science and Technology, Peking University, Beijing, P.R.China, 100871
16. [Hui Lin, Zhu Qing](http://www.directionsmag.com/author.php?author_id=238) - Data Visualization: Virtual Geographic Environments combining AEC and GIS, статья журнала « Directions Magazine», 2004г.

**ПРИЛОЖЕНИЕ А**

**Класс MainForm**

**public MainForm()**

{

InitializeComponent();

map.MapFunctions.Add("SelectInfo", new SelectInfoFunction(map));

map.MapFunctions.Add("MyZoomOut", new MyZoomOutFunction(map));

map.MapFunctions.Add("MyZoomIn", new MyZoomInFunction(map));

map.MapFunctions.Add("MyMeasure", new MyMeasureFunction(map));

map.FunctionMode = FunctionModes.None;

}

**private void MainForm\_Load(object sender, EventArgs e)**

{

loadform lf = new loadform();

fi = new Icons();

lf.Show();

loadCity();

lf.loadbar.Value = 10;

loadRaion();

lf.loadbar.Value = 20;

loadMainStreet();

lf.loadbar.Value = 40;

loadBigStreet();

lf.loadbar.Value = 60;

loadStreets();

lf.loadbar.Value = 80;

loadBuildings();

loadSvetofor();

loadPeshehod();

loadPlaces();

lf.loadbar.Value = 100;

loadIcons();

lf.Close();

InvisibleLayers();

editTrue();

City.IsVisible = true;

selectedFalse();

City.SelectionEnabled = true;

road\_scheme\_create();

MainStreetTraff.IsVisible = false;

MainStreetIco.IsVisible = false;

Svetofor.IsVisible = false;

Peshehod.IsVisible = false;

Places.IsVisible = false;

map.Invalidate();

Rectangle r = map.MapFrame.View;

int w = r.Width;

int h = r.Height;

r.Inflate((int)(r.Width / 4), (int)(r.Height / 4));

map.MapFrame.View = r;

map.MapFrame.ResetExtents();

}

**public void road\_scheme\_create()**

{

road\_vih1 = new LineScheme();

road\_vih2 = new LineScheme();

road\_vih3 = new LineScheme();

road\_bud1 = new LineScheme();

road\_bud2 = new LineScheme();

road\_bud3 = new LineScheme();

Color color1 = Color.FromArgb(86, 255, 66);

Color color2 = Color.FromArgb(251, 251, 36);

Color color3 = Color.FromArgb(242, 177, 57);

Color color4 = Color.FromArgb(242, 55, 57);

LineCategory traff11 = new LineCategory(color1, 4);

traff11.FilterExpression = "[vih1] < 3";

LineCategory traff12 = new LineCategory(color2, 4);

traff12.FilterExpression = "[vih1] >= 3";

LineCategory traff13 = new LineCategory(color3, 4);

traff13.FilterExpression = "[vih1] >= 6";

LineCategory traff14 = new LineCategory(color4, 4);

traff14.FilterExpression = "[vih1] >= 8";

road\_vih1.ClearCategories();

road\_vih1.AddCategory(traff11);

road\_vih1.AddCategory(traff12);

road\_vih1.AddCategory(traff13);

road\_vih1.AddCategory(traff14);

LineCategory traff21 = new LineCategory(color1, 4);

traff21.FilterExpression = "[vih2] < 3";

LineCategory traff22 = new LineCategory(color2, 4);

traff22.FilterExpression = "[vih2] >= 3";

LineCategory traff23 = new LineCategory(color3, 4);

traff23.FilterExpression = "[vih2] >= 4";

LineCategory traff24 = new LineCategory(color4, 4);

traff24.FilterExpression = "[vih2] >= 5";

road\_vih2.ClearCategories();

road\_vih2.AddCategory(traff21);

road\_vih2.AddCategory(traff22);

road\_vih2.AddCategory(traff23);

road\_vih2.AddCategory(traff24);

LineCategory traff31 = new LineCategory(color1, 5);

traff31.FilterExpression = "[vih3] < 3";

LineCategory traff32 = new LineCategory(color2, 5);

traff32.FilterExpression = "[vih3] >= 3";

LineCategory traff33 = new LineCategory(color3, 5);

traff33.FilterExpression = "[vih3] >= 6";

LineCategory traff34 = new LineCategory(color4, 5);

traff34.FilterExpression = "[vih3] >= 8";

road\_vih3.ClearCategories();

road\_vih3.AddCategory(traff31);

road\_vih3.AddCategory(traff32);

road\_vih3.AddCategory(traff33);

road\_vih3.AddCategory(traff34);

LineCategory traff41 = new LineCategory(color1, 4);

traff41.FilterExpression = "[bud1] < 3";

LineCategory traff42 = new LineCategory(color2, 4);

traff42.FilterExpression = "[bud1] >= 3";

LineCategory traff43 = new LineCategory(color3, 4);

traff43.FilterExpression = "[bud1] >= 6";

LineCategory traff44 = new LineCategory(color4, 4);

traff44.FilterExpression = "[bud1] >= 8";

road\_bud1.ClearCategories();

road\_bud1.AddCategory(traff41);

road\_bud1.AddCategory(traff42);

road\_bud1.AddCategory(traff43);

road\_bud1.AddCategory(traff44);

LineCategory traff51 = new LineCategory(color1, 4);

traff51.FilterExpression = "[bud2] < 3";

LineCategory traff52 = new LineCategory(color2, 4);

traff52.FilterExpression = "[bud2] >= 3";

LineCategory traff53 = new LineCategory(color3, 4);

traff53.FilterExpression = "[bud2] >= 6";

LineCategory traff54 = new LineCategory(color4, 4);

traff54.FilterExpression = "[bud2] >= 8";

road\_bud2.ClearCategories();

road\_bud2.AddCategory(traff41);

road\_bud2.AddCategory(traff42);

road\_bud2.AddCategory(traff43);

road\_bud2.AddCategory(traff44);

LineCategory traff61 = new LineCategory(color1, 4);

traff61.FilterExpression = "[bud1] < 3";

LineCategory traff62 = new LineCategory(color2, 4);

traff62.FilterExpression = "[bud1] >= 3";

LineCategory traff63 = new LineCategory(color3, 4);

traff63.FilterExpression = "[bud1] >= 6";

LineCategory traff64 = new LineCategory(color4, 4);

traff64.FilterExpression = "[bud1] >= 8";

road\_bud3.ClearCategories();

road\_bud3.AddCategory(traff41);

road\_bud3.AddCategory(traff42);

road\_bud3.AddCategory(traff43);

road\_bud3.AddCategory(traff44);

}

**private void selectedFalse()**

{

for (int i = 0; i < mapLayers.Count; i++)

{

mapLayers[i].SelectionEnabled = false;

}

}

**private void editFalse()**

{

for (int i = 0; i < mapLayers.Count; i++)

{

mapLayers[i].EditMode = false;

}

}

**private void editTrue()**

{

for (int i = 0; i < mapLayers.Count; i++)

{

mapLayers[i].EditMode = true;

}

}

**private void InvisibleLayers()**

{

for (int i = 0; i < mapLayers.Count; i++)

{

mapLayers[i].IsVisible = false;

}

}

**private void loadCity()**

{

Color color1 = Color.FromArgb(250, 240, 145);

PolygonSymbolizer style = new PolygonSymbolizer(color1);

style.OutlineSymbolizer = new LineSymbolizer(Color.Black, 2);

City = map.Layers.Add (DataManager.DefaultDataManager.OpenVector

(@"GisData\City.shp", false, prog));

preview = map1.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\City.shp", false, prog));

preview.Symbolizer = style;

mapLayers.Add(City);

IMapLabelLayer labelLayer = new MapLabelLayer();

lb = new LabelSymbolizer();

lb.FontSize = 15;

lb.FontStyle = FontStyle.Bold;

lb.FontFamily = "Arial";

ILabelCategory category = labelLayer.Symbology.Categories[0];

category.Expression = "[name]";

labelLayer.Symbolizer = lb;

category.Symbolizer.Orientation = ContentAlignment.MiddleCenter;

City.LabelLayer = labelLayer;

City.ShowLabels = true;

City.Symbolizer = style;

}

**private void CityAsImage()**

{

City.IsVisible = true;

City.SelectionEnabled = false;

City.EditMode = false;

City.ShowLabels = false;

}

**public void loadRaion()**

{

scheme = new PolygonScheme();

Color color1 = Color.FromArgb(250, 240, 145);

PolygonCategory pop1 = new PolygonCategory(color1, Color.Black, 2);

pop1.FilterExpression = "[population] < 150000";

Color color2 = Color.FromArgb(250, 230, 90);

PolygonCategory pop2 = new PolygonCategory(color2, Color.Black, 2);

pop2.FilterExpression = " [population] > 150000 ";

Color color3 = Color.FromArgb(240, 215, 10);

PolygonCategory pop3 = new PolygonCategory(color3, Color.Black, 2);

pop3.FilterExpression = " [population] > 200000 ";

Color color4 = Color.FromArgb(215, 190, 5);

PolygonCategory pop4 = new PolygonCategory(color4, Color.Black, 2);

pop4.FilterExpression = " [population] > 250000 ";

scheme.ClearCategories();

scheme.AddCategory(pop1);

scheme.AddCategory(pop2);

scheme.AddCategory(pop3);

scheme.AddCategory(pop4);

scheme2 = new PolygonScheme();

PolygonCategory pop12 = new PolygonCategory(color1, Color.Black, 2);

pop12.FilterExpression = "[area] < 3";

PolygonCategory pop22 = new PolygonCategory(color2, Color.Black, 2);

pop22.FilterExpression = " [area] > 2 ";

PolygonCategory pop32 = new PolygonCategory(color3, Color.Black, 2);

pop32.FilterExpression = " [area] > 3 ";

PolygonCategory pop42 = new PolygonCategory(color4, Color.Black, 2);

pop42.FilterExpression = " [area] > 5 ";

scheme2.ClearCategories();

scheme2.AddCategory(pop12);

scheme2.AddCategory(pop22);

scheme2.AddCategory(pop32);

scheme2.AddCategory(pop42);

Raions = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\Raion.shp", false, prog));

mapLayers.Add(Raions);

Raions.Symbology = scheme;

IMapLabelLayer labelLayer = new MapLabelLayer();

ILabelCategory category = labelLayer.Symbology.Categories[0];

category.Expression = "[name]";

labelLayer.Symbolizer = lb;

category.Symbolizer.Orientation = ContentAlignment.MiddleCenter;

IMapLabelLayer labelLayer2 = labelLayer;

PolygonSymbolizer style = new PolygonSymbolizer(color1);

style.OutlineSymbolizer = new LineSymbolizer(Color.Black, 2);

Raions.Symbolizer = style;

Raions.LabelLayer = labelLayer;

Raions.ShowLabels = true;

}

**private void loadMainStreet()**

{

PolygonSymbolizer water = new PolygonSymbolizer(Color.LightBlue);

water.OutlineSymbolizer = new LineSymbolizer(Color.LightBlue, 1);

Water = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\Water.shp", false, prog));

mapLayers.Add(Water);

Water.Symbolizer = water;

LineSymbolizer ls = new LineSymbolizer(Color.Gray, 8);

road\_scheme = new LineScheme();

Color color1 = Color.FromArgb(0, 0, 0);

LineCategory traff1 = new LineCategory(color1, 6);

traff1.FilterExpression = "[traffic] < 4";

Color color2 = Color.FromArgb(108, 108, 108);

LineCategory traff2 = new LineCategory(color2, 6);

traff2.FilterExpression = " [traffic] >= 4 ";

Color color3 = Color.FromArgb(165, 165, 165);

LineCategory traff3 = new LineCategory(color3, 6);

traff3.FilterExpression = " [traffic] > 5 ";

Color color4 = Color.FromArgb(212, 212, 212);

LineCategory traff4 = new LineCategory(color4, 6);

traff4.FilterExpression = " [traffic] > 8 ";

road\_scheme.ClearCategories();

road\_scheme.AddCategory(traff1);

road\_scheme.AddCategory(traff2);

road\_scheme.AddCategory(traff3);

road\_scheme.AddCategory(traff4);

road\_scheme2 = new LineScheme();

LineCategory traff12 = new LineCategory(color1, 6);

traff12.FilterExpression = "[quality] < 4";

LineCategory traff22 = new LineCategory(color2, 6);

traff22.FilterExpression = " [quality] >= 4 ";

LineCategory traff32 = new LineCategory(color3, 6);

traff32.FilterExpression = " [quality] > 5 ";

LineCategory traff42 = new LineCategory(color4, 6);

traff42.FilterExpression = " [quality] > 8 ";

road\_scheme2.ClearCategories();

road\_scheme2.AddCategory(traff12);

road\_scheme2.AddCategory(traff22);

road\_scheme2.AddCategory(traff32);

road\_scheme2.AddCategory(traff42);

MainStreet = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\MainStreet.shp", false, prog));

mapLayers.Add(MainStreet);

MainStreet.Symbology = road\_scheme;

MainStreet.Symbolizer = ls;

MainStreetTraff = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\traff.shp", false, prog));

LineSymbolizer ls2 = new LineSymbolizer(Color.Red, 5);

MainStreetTraff.Symbolizer = ls2;

PointSymbolizer icon = new PointSymbolizer(Color.Gray, PointShapes.Ellipse, 10);

MainStreetIco = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\MainStreetIco.shp", false, prog));

MainStreetIco.Symbolizer = icon;

}

**private void loadBigStreet()**

{

BigStreet = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\BigStreet.shp", false, prog));

BigStreet.Symbology = road\_scheme;

mapLayers.Add(BigStreet);

}

**private void loadStreets()**

{

Streets = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\Streets.shp", false, prog));

Streets.Symbology = road\_scheme;

mapLayers.Add(Streets);

}

**private void loadBuildings()**

{

PolygonSymbolizer style = new PolygonSymbolizer(Color.LightGreen);

style.OutlineSymbolizer = new LineSymbolizer(Color.Black, 2);

Buildings = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\Buildings.shp", false, prog));

mapLayers.Add(Buildings);

IMapLabelLayer labelLayer = new MapLabelLayer();

LabelSymbolizer lb = new LabelSymbolizer();

lb.FontSize = 15;

ILabelCategory category = labelLayer.Symbology.Categories[0];

category.Expression = "[number]";

labelLayer.Symbolizer = lb;

Buildings.LabelLayer = labelLayer;

Buildings.ShowLabels = true;

Buildings.Symbolizer = style;

}

**private void loadSvetofor()**

{

PointSymbolizer sveto = new PointSymbolizer(Images.sveto, 20);

Svetofor = map.Layers.Add(DataManager.DefaultDataManager.OpenVector(@"GisData\Svetofor.shp", false, prog));

Svetofor.Symbolizer = sveto;

}

**private void loadPeshehod()**

{

PointSymbolizer pesh = new PointSymbolizer(Images.znak, 20);

Peshehod = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\Peshehod.shp", false, prog));

Peshehod.Symbolizer = pesh;

}

**private void loadPlaces()**

{

PointSymbolizer place = new PointSymbolizer(Images.\_4051, 20);

Places = map.Layers.Add(DataManager.DefaultDataManager.OpenVector

(@"GisData\Places.shp", false, prog));

Places.Symbolizer = place;

}

**private void UncheckOtherButtonsButMe(ToolStripButton checkedButton)**

{

foreach (ToolStripItem item in this.toolStrip1.Items)

{

ToolStripButton buttonItem = item as ToolStripButton;

if (buttonItem != null)

{

if (buttonItem.Name != checkedButton.Name)

{

buttonItem.Checked = false;

}

}

}

}

**private void cmdZoomIn\_Click(object sender, EventArgs e)**

{

if (map == null) return;

if (cmdZoomIn.Checked == false)

{

UncheckOtherButtonsButMe(cmdZoomIn);

cmdZoomIn.Checked = true;

foreach (IMapFunction gt in map.MapFunctions.Values)

{

gt.Deactivate();

MyZoomInFunction zf = gt as MyZoomInFunction;

if (zf != null)

{

zf.Activate();

map.Cursor = Cursors.UpArrow;

this.mode = 1;

try

{

MemoryStream ms = new MemoryStream(Resource.cursorZoomIn);

map.Cursor = new Cursor(ms);

}

catch

{

map.Cursor = Cursors.Arrow;

}

continue;

}

}

}

}

**private void cmdZoomOut\_Click(object sender, EventArgs e)**

{

if (map == null) return;

if (cmdZoomOut.Checked == false)

{

UncheckOtherButtonsButMe(cmdZoomOut);

cmdZoomOut.Checked = true;

foreach (IMapFunction gt in map.MapFunctions.Values)

{

gt.Deactivate();

MyZoomOutFunction zf = gt as MyZoomOutFunction;

if (zf != null)

{

zf.Activate();

map.Cursor = Cursors.UpArrow;

this.mode = 2;

try

{

MemoryStream ms = new MemoryStream(Resource.cursorZoomOut);

map.Cursor = new Cursor(ms);

}

catch

{

map.Cursor = Cursors.Arrow;

}

continue;

}

}

}

}

**private void cmdMaxExtents\_Click(object sender, EventArgs e)**

{

if (map == null) return;

UncheckOtherButtonsButMe(cmdMaxExtents);

map.ZoomToMaxExtent();

map.FunctionMode = FunctionModes.None;

mode = 0;

zoomCount = 0;

checkVisual.Enabled = false;

InvisibleLayers();

City.IsVisible = true;

if (checkLabels.Checked) City.ShowLabels = true;

else City.ShowLabels = false;

City.SelectionEnabled = true;

map.Invalidate();

Rectangle r = map.MapFrame.View;

int w = r.Width;

int h = r.Height;

r.Inflate((int)(r.Width / 4), (int)(r.Height / 4));

map.MapFrame.View = r;

map.MapFrame.ResetExtents();

}

**private void cmdPan\_Click(object sender, EventArgs e)**

{

if (map == null) return;

if (cmdPan.Checked == false)

{

UncheckOtherButtonsButMe(cmdPan);

cmdPan.Checked = true;

map.FunctionMode = FunctionModes.Pan;

mode = 0;

}

}

**private void cmdSelect\_Click(object sender, EventArgs e)**

{

if (map == null) return;

if (cmdSelect.Checked == false)

{

UncheckOtherButtonsButMe(cmdSelect);

cmdSelect.Checked = true;

map.FunctionMode = FunctionModes.Info;

foreach (IMapFunction gt in map.MapFunctions.Values)

{

gt.Deactivate();

SelectInfoFunction zf = gt as SelectInfoFunction;

if (zf != null)

{

zf.Activate();

map.Cursor = Cursors.Hand;

mode = 0;

continue;

}

}

}

}

**private void cmdCalc\_Click(object sender, EventArgs e)**

{

if (map == null) return;

if (cmdCalc.Checked == false)

{

UncheckOtherButtonsButMe(cmdCalc);

cmdCalc.Checked = true;

foreach (IMapFunction gt in map.MapFunctions.Values)

{

gt.Deactivate();

MyMeasureFunction zf = gt as MyMeasureFunction;

if (zf != null)

{

zf.Activate();

map.Cursor = Cursors.Cross;

mode = 0;

continue;

}

}

}

if (cmdCalc.Checked == false)

{

UncheckOtherButtonsButMe(cmdCalc);

cmdCalc.Checked = true;

map.FunctionMode = FunctionModes.Measure;

}

}

**private void map\_click(object sender, EventArgs e)**

{

if (this.mode == 1 && ((MouseEventArgs)e).Button == MouseButtons.Left)

{

this.zoomCount += 1;

Console.Out.WriteLine(this.zoomCount);

}

else if (this.mode == 1 && ((MouseEventArgs)e).Button == MouseButtons.Right)

{

this.zoomCount -= 1;

Console.Out.WriteLine(this.zoomCount);

}

else if (this.mode == 2 && ((MouseEventArgs)e).Button == MouseButtons.Right)

{

this.zoomCount += 1;

Console.Out.WriteLine(this.zoomCount);

}

else if (this.mode == 2 && ((MouseEventArgs)e).Button == MouseButtons.Left)

{

this.zoomCount -= 1;

Console.Out.WriteLine(this.zoomCount);

}

if (zoomCount == 1)

{

InvisibleLayers();

City.IsVisible = true;

selectedFalse();

City.SelectionEnabled = true;

if (checkLabels.Checked) City.ShowLabels = true;

else City.ShowLabels = false;

panel1.Visible = false;

checkVisual.Enabled = false;

}

if (zoomCount >= 2)

{

InvisibleLayers();

Raions.IsVisible = true;

selectedFalse();

editTrue();

Raions.SelectionEnabled = true;

checkVisual.Enabled = true;

btnVis1.Text = "По населению";

btnVis2.Text = "По площади";

}

if (zoomCount >= 3)

{

map.MapFrame.ClearSelection();

InvisibleLayers();

CityAsImage();

Water.IsVisible = true;

MainStreet.IsVisible = true;

MainStreetIco.IsVisible = true;

selectedFalse();

editTrue();

Water.SelectionEnabled = true;

MainStreet.SelectionEnabled = true;

btnVis1.Text = "По качеству полотна";

btnVis2.Text = "По средней скорости";

}

if (zoomCount >= 6)

{

CityAsImage();

Raions.IsVisible = false;

MainStreet.IsVisible = true;

MainStreetIco.EditMode = false;

Water.IsVisible = true;

BigStreet.IsVisible = true;

selectedFalse();

MainStreet.SelectionEnabled = true;

Water.SelectionEnabled = true;

BigStreet.SelectionEnabled = true;

btnVis1.Text = "По уровню пробок";

btnVis2.Text = "По качеству покрытия";

}

if (zoomCount >= 13)

{

CityAsImage();

Raions.IsVisible = false;

MainStreet.IsVisible = true;

Water.IsVisible = true;

BigStreet.IsVisible = true;

Streets.IsVisible = true;

selectedFalse();

MainStreet.SelectionEnabled = true;

Water.SelectionEnabled = true;

BigStreet.SelectionEnabled = true;

Streets.SelectionEnabled = true;

btnVis1.Text = "По уровню пробок";

btnVis2.Text = "По качеству покрытия";

}

if (zoomCount >= 15)

{

CityAsImage();

Raions.IsVisible = false;

MainStreet.IsVisible = true;

Water.IsVisible = true;

BigStreet.IsVisible = true;

Streets.IsVisible = true;

Buildings.IsVisible = true;

selectedFalse();

MainStreet.SelectionEnabled = true;

Water.SelectionEnabled = true;

BigStreet.SelectionEnabled = true;

Streets.SelectionEnabled = true;

Buildings.SelectionEnabled = true;

}

}

**private void map\_2click(object sender, EventArgs e)**

{

map\_click(sender, e);

}

**private void btnFind\_Click(object sender, EventArgs e)**

{

String street;

int house;

street = textStreet.Text;

house = int.Parse(textHouse.Text);

zoomCount = 10;

map\_click(sender,e);

Buildings.IsVisible = true;

Buildings.SelectByAttribute("[street]= '"+street+"' AND [number]="+house);

Buildings.ZoomToSelectedFeatures();

Rectangle r = map.MapFrame.View;

int w = r.Width;

int h = r.Height;

r.Inflate((int)(r.Width / 10),(int)( r.Height / 10));

map.MapFrame.View = r;

map.MapFrame.ResetExtents();

}

**private void cmdPrint\_Click(object sender, EventArgs e)**

{

MyLayoutForm1 lf = new MyLayoutForm1();

lf.MapControl = map;

lf.Show();

}

**private void btnVis2\_CheckedChanged(object sender, EventArgs e)**

{

if (zoomCount==2) Raions.Symbology = scheme2;

if (zoomCount >= 3)

{

MainStreet.Symbology = road\_scheme2;

BigStreet.Symbology = road\_scheme2;

Streets.Symbology = road\_scheme2;

}

}

**private void btnVis1\_CheckedChanged(object sender, EventArgs e)**

{

if (zoomCount == 2) Raions.Symbology = scheme;

if (zoomCount >= 3)

{

MainStreet.Symbology = road\_scheme;

BigStreet.Symbology = road\_scheme;

Streets.Symbology = road\_scheme;

}

}

**private void traffic\_CheckedChanged(object sender, EventArgs e)**

{

if (traffic.Checked) traffic\_panel.Enabled = true;

else traffic\_panel.Enabled = false;

}

**private void traffic\_find\_Click(object sender, EventArgs e)**

{

MainStreetTraff.IsVisible = true;

if (day.SelectedItem == "Выходные")

{

if (time.SelectedItem == "1") MainStreetTraff.Symbology = road\_vih1;

else if (time.SelectedItem == "2") MainStreetTraff.Symbology = road\_vih2;

else if (time.SelectedItem == "3") MainStreetTraff.Symbology = road\_vih3;

}

else if (day.SelectedItem == "Бцдни")

{

if (time.SelectedItem == "1") MainStreetTraff.Symbology = road\_bud1;

else if (time.SelectedItem == "2") MainStreetTraff.Symbology = road\_bud2;

else if (time.SelectedItem == "3") MainStreetTraff.Symbology = road\_bud3;

}

}

**private void checkLabels\_CheckedChanged(object sender, EventArgs e)**

{

if (checkLabels.Checked)

{

for (int i = 0; i < mapLayers.Count; i++)

{

mapLayers[i].ShowLabels = true;

}

map.MapFrame.ResetExtents();

}

else

{

for (int i = 0; i < mapLayers.Count; i++)

{

mapLayers[i].ShowLabels = false;

}

map.MapFrame.ResetExtents();

}

}

**private void checkVisual\_CheckedChanged(object sender, EventArgs e)**

{

if (checkVisual.Checked)

{

Raions.EditMode = false;

MainStreet.EditMode = false;

BigStreet.EditMode = false;

Streets.EditMode = false;

map.MapFrame.ResetExtents();

panel1.Visible = true;

}

else

{

Raions.EditMode = true;

MainStreet.EditMode = true;

BigStreet.EditMode = true;

Streets.EditMode = true;

map.MapFrame.ResetExtents();

panel1.Visible = false;

}

}

**private void tram\_CheckedChanged(object sender, EventArgs e)**

{

if (tram.Checked) tram\_panel.Enabled = true;

else tram\_panel.Enabled = false;

}

**private void cmdFind\_Click(object sender, EventArgs e)**

{

MySelectByAttributes sb = new MySelectByAttributes(map.MapFrame);

sb.Show();

}

**private void cmdExit\_Click(object sender, EventArgs e)**

{

Application.Exit();

}

**private void checkBox5\_CheckedChanged(object sender, EventArgs e)**

{

if (checkSveto.Checked)

{

Svetofor.EditMode = false;

Svetofor.IsVisible = true;

map.MapFrame.ResetExtents();

}

else

{

Svetofor.IsVisible = false;

map.MapFrame.ResetExtents();

}

}

**private void checkBox6\_CheckedChanged(object sender, EventArgs e)**

{

if (checkPesh.Checked)

{

Peshehod.EditMode = false;

Peshehod.IsVisible = true;

map.MapFrame.ResetExtents();

}

else

{

Peshehod.IsVisible = false;

map.MapFrame.ResetExtents();

}

}

**private void button1\_Click(object sender, EventArgs e)**

{

MainStreetTraff.IsVisible = false;

}

**private void checIco\_CheckedChanged(object sender, EventArgs e)**

{

if (checkIco.Checked)

{

fi.Show();

}

else fi.Visible = false;

}

**private void checkPlace\_CheckedChanged(object sender, EventArgs e)**

{

if (checkPlace.Checked)

{

Places.EditMode = false;

Places.IsVisible = true;

map.MapFrame.ResetExtents();

}

else

{

Places.IsVisible = false;

map.MapFrame.ResetExtents();

}

}

**Класс MyFeatureIdentifier**

**public MyFeatureIdentifier()**

{

InitializeComponent();

}

**public void Add(IFeatureLayer layer, IFeature f)**

{

if (f == null) return;

DataTable dt = new DataTable();

dt.Columns.Add("Field Name");

dt.Columns.Add("Value");

if (f.DataRow == null)

{

f.ParentFeatureSet.FillAttributes();

}

DataColumn[] columns = f.ParentFeatureSet.GetColumns();

for (int i = 0; i < columns.Length - 2; i++)

{

DataColumn fld = columns[i];

DataRow dr = dt.NewRow();

dr["Field Name"] = fld.ColumnName;

if (f.DataRow != null) dr["Value"] =

f.DataRow[fld.ColumnName].ToString();

dt.Rows.Add(dr);

}

dgvAttributes.DataSource = dt;

String photo = f.DataRow[columns[columns.Length-2].ColumnName].ToString();

photo = @"Images\" + photo;

pictureBox1.ImageLocation = photo;

String text\_file =

f.DataRow[columns[columns.Length - 1].ColumnName].ToString();

text\_file = @"Texts\" + text\_file;

try

{

StreamReader fs =

new StreamReader(text\_file, System.Text.Encoding.Default);

String text = fs.ReadToEnd();

label1.Text = text;

}

catch (ArgumentException)

{

label1.Text = "Описание отсутствует";

}

catch (FileNotFoundException)

{

label1.Text = "Описание отсутствует";

}

catch (DirectoryNotFoundException)

{

label1.Text = "Описание отсутствует";

}

}

**protected override void OnClosing(CancelEventArgs e)**

{

e.Cancel = true;

Hide();

}

**Класс MyLayoutForm**

**private void toolStripMap\_Click(object sender, EventArgs e)**

{

\_layoutControl1.AddElementWithMouse(\_layoutControl1.CreateMapElement());

}

**private void toolStripRect\_Click(object sender, EventArgs e)**

{

\_layoutControl1.AddElementWithMouse(new Layout.Elements.LayoutRectangle());

}

**private void toolStripNorth\_Click(object sender, EventArgs e)**

{

\_layoutControl1.AddElementWithMouse(new Layout.Elements.LayoutNorthArrow());

}

**private void toolStripImg\_Click(object sender, EventArgs e)**

{

OpenFileDialog ofd = new OpenFileDialog();

ofd.Filter = "Images (\*.png, \*.jpg, \*.bmp, \*.gif, \*.tif)|\*.png;\*.jpg;\*.bmp;\*.gif;\*.tif";

ofd.FilterIndex = 1;

ofd.CheckFileExists = true;

if (ofd.ShowDialog() == DialogResult.OK)

{

Layout.Elements.LayoutBitmap newBitmap = new Layout.Elements.LayoutBitmap();

newBitmap.Size = new System.Drawing.SizeF(100, 100);

newBitmap.Filename = ofd.FileName;

\_layoutControl1.AddElementWithMouse(newBitmap);

}

}

**Класс MySelectByAttributes**

**public MySelectByAttributes()**

{

InitializeComponent();

Configure();

}

**public MySelectByAttributes(IFrame mapFrame)**

**{**

\_mapFrame = mapFrame;

InitializeComponent();

Configure();

}

**private void Configure()**

{

DataTable dt = new DataTable();

dt.Columns.Add("Name", typeof(string));

dt.Columns.Add("Value", typeof(IFeatureLayer));

foreach(ILayer layer in \_mapFrame)

{

IFeatureLayer fl = layer as IFeatureLayer;

if (fl != null)

{

DataRow dr = dt.NewRow();

dr["Name"] = fl.LegendText;

dr["Value"] = fl;

dt.Rows.Add(dr);

}

}

cmbLayers.DataSource = dt;

cmbLayers.DisplayMember = "Name";

cmbLayers.ValueMember = "Value";

cmbMethod.SelectedIndex = 0;

if(cmbLayers.Items.Count > 0) cmbLayers.SelectedIndex = 0;

}

**public IFrame MapFrame**

{

get { return \_mapFrame; }

set

{

\_mapFrame = value;

Configure();

}

}

**protected override void Dispose(bool disposing)**

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

**private void cmbLayers\_SelectedIndexChanged(object sender, EventArgs e)**

{

DataRowView drv = cmbLayers.SelectedValue as DataRowView;

if (drv != null)

{

\_activeLayer = drv.Row["Value"] as IFeatureLayer;

}

else

{

\_activeLayer = cmbLayers.SelectedValue as IFeatureLayer;

}

if (\_activeLayer == null) return;

if(!\_activeLayer.DataSet.AttributesPopulated && \_activeLayer.DataSet.NumRows() < 50000)

{

\_activeLayer.DataSet.FillAttributes();

}

if(\_activeLayer.EditMode || \_activeLayer.DataSet.AttributesPopulated)

{

sqlQueryControl1.Table = \_activeLayer.DataSet.DataTable;

}

else

{

sqlQueryControl1.AttributeSource = \_activeLayer.DataSet;

}

}

**private void btnApply\_Click(object sender, EventArgs e)**

{

ApplyFilter();

}

**private void ApplyFilter()**

{

string filter = sqlQueryControl1.ExpressionText;

if (\_activeLayer != null)

{

try

{

\_activeLayer.SelectByAttribute(filter, GetSelectMode());

}

catch(Exception ex)

{

Debug.WriteLine(ex.ToString());

MessageBox.Show("There was an error attempting to apply this expression.");

}

}

}

**ModifySelectionModes GetSelectMode()**

{

switch (cmbMethod.SelectedIndex)

{

case 0: return ModifySelectionModes.Replace;

case 1: return ModifySelectionModes.Append;

case 2: return ModifySelectionModes.Subtract;

case 3: return ModifySelectionModes.SelectFrom;

}

return ModifySelectionModes.Replace;

}

**private void btnOk\_Click(object sender, EventArgs e)**

{

ApplyFilter();

Close();

}

**private void btnClose\_Click(object sender, EventArgs e)**

{

Close();

}

**Класс MyZoomInFunction**

**protected override void OnMouseUp(GeoMouseArgs e)**

{

\_currentPoint = e.Location;

Map.Invalidate();

Rectangle r = e.Map.MapFrame.View;

int w = r.Width;

int h = r.Height;

if (e.Button == MouseButtons.Left)

{

r.Inflate(-r.Width / 10, -r.Height / 10);

e.Map.MapFrame.View = r;

e.Map.MapFrame.ResetExtents();

}

else

{

r.Inflate((int)(r.Width / 10), (int)(r.Height / 10));

e.Map.MapFrame.View = r;

e.Map.MapFrame.ResetExtents();

}

base.OnMouseUp(e);

}

**Класс MyZoomInFunction**

**protected override void OnMouseUp(GeoMouseArgs e)**

{

\_currentPoint = e.Location;

Map.Invalidate();

Rectangle r = e.Map.MapFrame.View;

int w = r.Width;

int h = r.Height;

if (e.Button == MouseButtons.Left)

{

r.Inflate((int)(r.Width / 10),(int)( r.Height / 10));

e.Map.MapFrame.View = r;

e.Map.MapFrame.ResetExtents();

}

else

{

r.Inflate(-r.Width / 10, -r.Height / 10);

e.Map.MapFrame.View = r;

e.Map.MapFrame.ResetExtents();

}

base.OnMouseUp(e);

}

**Класс SelectInfoFunction**

**protected override void OnMouseDoubleClick(GeoMouseArgs e)**

{

Coordinate c1 = e.Map.PixelToProj(new System.Drawing.Point(e.X - 2, e.Y - 2));

Coordinate c2 = e.Map.PixelToProj(new System.Drawing.Point(e.X + 2, e.Y + 2));

Coordinate s1 = e.Map.PixelToProj(new System.Drawing.Point(e.X - 1, e.Y - 1));

Coordinate s2 = e.Map.PixelToProj(new System.Drawing.Point(e.X + 1, e.Y + 1));

Coordinate center = e.Map.PixelToProj(new System.Drawing.Point(e.X, e.Y));

IEnvelope tolerant = new Envelope(c1, c2);

IEnvelope strict = new Envelope(s1, s2);

HandleSelection(tolerant, strict);

if (frmFeatureIdentifier == null)

{

frmFeatureIdentifier = new MyFeatureIdentifier();

}

if (Identify(e.Map.MapFrame.GetLayers(), strict, tolerant))

{

frmFeatureIdentifier.SuspendLayout();

System.Console.Out.WriteLine(e.Map.MapFrame.GetLayers().Count);

frmFeatureIdentifier.ResumeLayout();

frmFeatureIdentifier.Show();

base.OnMouseDoubleClick(e);

}

else System.Console.Out.WriteLine("1213");

}

**private bool Identify(IEnumerable<ILayer> layers, IEnvelope strict, IEnvelope tolerant)**

{

foreach (IMapLayer layer in layers)

{

IMapFeatureLayer gfl = layer as IMapFeatureLayer;

if (gfl != null)

{

if (gfl.SelectionEnabled)

{

IFeature ff = new Feature();

List<IFeature> result = gfl.Selection.ToFeatureList();

if (result.Count > 0)

{

ff = result[0];

frmFeatureIdentifier.Add(gfl, ff);

return true;

}

}

}

}

return false;

}

**private void HandleSelection(IEnvelope tolerant, IEnvelope strict)**

{

Map.Select(tolerant, strict);

Map.MapFrame.ResetExtents();

}

}