Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное
образовательное учреждение
Высшего образования
Алтайский государственный технический университет
им. И.И.Ползунова

НАУКА И МОЛОДЕЖЬ – 2016

XIII Всероссийская научно-техническая конференция студентов, аспирантов и молодых ученых

СЕКЦИЯ

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

подсекция

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

Барнаул – 2016

УДК 004

XIII Всероссийская научно-техническая конференция студентов, аспирантов и молодых ученых "Наука и молодежь – 2016". Секция «Информационные технологии». Подсекция «Программная инженерия». / Алт. гос. техн. ун-т им. И.И.Ползунова. – Барнаул: изд-во АлтГТУ, 2016. – 81 с.

В сборнике представлены работы научно-технической конференции студентов, аспирантов и молодых ученых, проходившей 22 апреля 2016 г.

СОДЕРЖАНИЕ

РАНЖИРОВАНИЕ ВЕРШИН СЕМАНТИЧЕСКИХ ГРАФОВ

Абрамов Д.В. – магистрант, Перепёлкин Е.А. – д.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Интернет развивается с неимоверной скоростью, и в наше время задачи поиска нужной информации в нем являются наиболее актуальными. PageRank, который лежит в основе поисковых движков Google и Yahoo!, и HITS - два наиболее значимых алгоритма для определения релевантности веб-странц. PageRank - это числовая величина, зависящая только от топологии интернета, которая не учитывает содержимое страницы или особенности запроса. Хорошее соответствие запросу обеспечивается путем сочетания оценки PageRank с другими эвристическими алгоритмами.

Алгоритм HITS, с другой стороны, сначала составляет сосредоточенный подграф из веб-страниц, которые должны быть наиболее подходящими под критерии поиска. Сингулярные векторы матрицы смежности этого подграфа определяют лучшие оценки авторитетности и посредничества для данного запроса. Но, к сожалению, данные оценки не всегда соответствуют критериям исходного запроса, т.е. вершины сосредоточенного подграфа не связаны с темой запроса.

Исследования семантических графов [1] вместо обычных могут частично решить данную проблему. Вершины и ребра семантических графов являются разнородными, что позволяет наиболее точно представлять сложные данные, такие как структура социальной сети.

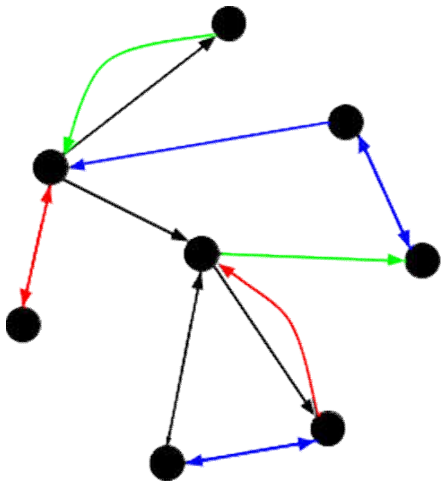


Рисунок 1 – Семантический граф

Для представления семантического графа с разнородными ребрами может быть использован трехмерный тензор [2], в котором каждый фронтальный двумерный срез является матрицей смежности для определенного типа ребер. Если в семантическом графе n вершин и m типов ребер, то для представления графа можно использовать трехмерный тензор (рисунок 2) размерности $n \times n \times m$, где элемент с индексами (i, j, l) принимает значение отличное от нуля, если вершина i соединена с вершиной j посредством ребра с типом связи l.

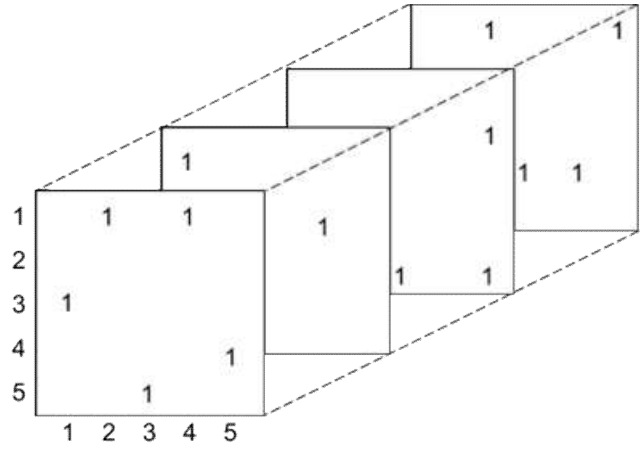


Рисунок 2 – Трехмерный тензор

Один из алгоритмов, применяемых к семантическим графам, является TOPHITS [3]. Этот алгоритм учитывает типы связей между вершинами графа, что позволяет получить более точные результаты. Он, аналогично алгоритму HITS, вычисляет авторитетные и посреднические оценки вместе с оценкой значимости типа связи.

Пусть n это количество вершин в сосредоточенном подграфе, а m - количество видов связей. Каждая вершина имеет оценку авторитетности (a), посредническую оценку (h) и оценку значимости типа связи (t), которые вычисляются согласно следующим формулам:

$$
\begin{aligned}
h_i^{(k+1)} &= \sum_{i \xrightarrow{l} j} a_j^{(k)} t_l^{(k)}, \quad i = 1,...,n, \\
a_j^{(k+1)} &= \sum_{i \xrightarrow{l} j} h_i^{(k+1)} t_l^{(k+1)}, \quad j = 1,...,n, \\
t_l^{(k+1)} &= \sum_{i \xrightarrow{l} j} a_j^{(k+1)} h_i^{(k+1)}, \quad l = 1,...,m.
\end{aligned}
\tag{1}
$$

Обозначение $i \xrightarrow{l} j$ в данном случае подразумевает, что вершина i связана с вершиной j ссылкой с типом связи l.

Как и в случае алгоритма HITS, после каждой итерации проводится нормализация векторов h, a и t. Таким образом, посредническая оценка вершины i равна сумме оценок авторитетных вершин, на которые она указывает, помноженной на соответствующую оценку значимости типа связи. Аналогично авторитетная оценка вершины j равна сумме значений оценок посреднических вершин, которые указывают на эту страницу, помноженной на соответствующую оценку значимости типа связи. В свою очередь, оценка значимости типа связи l вычисляется как сумма посреднических оценок вершины i, умноженных на авторитетные оценки вершины j по всем ссылкам $i \xrightarrow{l} j$, которые соответствуют типу связи l.

Тоже самые формулы можно записать в тензорной форме. Пусть A тензор смежности графа размерности $n \times n \times m$, для которого справедливо следующее:

1. $A_{ijl} = 1$, если есть ребро от вершины i в вершину j с типом связи l;
2. $A_{ijl} = 0$, в противном случае.

Тогда, формулы (1) можно представить в следующем виде:

$$h_i^{(k+1)} = \sum_{j=1}^{n}\sum_{l=1}^{m} A_{ijl} a_j^{(k)} t_l^{(k)}, \quad i = 1,...,n,$$

$$a_j^{(k+1)} = \sum_{i=1}^{n}\sum_{l=1}^{m} A_{ijl} h_i^{(k+1)} t_l^{(k)}, \quad j = 1,...,n, \qquad (2)$$

$$t_l^{(k+1)} = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ijl} h_i^{(k+1)} a_j^{(k+1)}, \quad l = 1,...,m.$$

Применяя алгоритм тензорного разложения PARAFAC [4], который представляет собой аппроксимацию ранга p исходного тензора, мы можем представить A как

$$A = \sum_{i=1}^{p} s^{(i)} u^{(i)} \otimes v^{(i)} \otimes w^{(i)}, \qquad (3)$$

где $a \otimes b \otimes c$ обозначает тензорное произведение трех векторов [5], т.е. тензор с элементами $(a \otimes b \otimes c)_{ijk} = a_i b_j c_k$. Иллюстрация такого разложения представлена на рисунке 3.

В отличие от сингулярного разложения матрицы, векторы, полученные при помощи алгоритма PARAFAC, не являются ортогональными, то есть $u^{(1)}$ не ортогонален $u^{(2)}$ и т.д.

Алгоритм PARAFAC при определенных условиях [4] вычисляет аппроксимацию ранга 1 тензора A. При этом предельные значения $h^{(k)}$, $a^{(k)}$, $t^{(k)}$ в алгоритме (2) равны соответственно $u^{(1)}$, $v^{(1)}$, $w^{(1)}$.

Наибольшие значения в векторе $w^{(1)}$ определяют доминирующие типы связи, тогда как наибольшие значения в векторах $u^{(1)}$ и $v^{(1)}$ определяют оценки авторитетности и посреднические оценки. Каждый триплет $\{u^{(i)}, v^{(i)}, w^{(i)}\}$ определяет другие типы связей и соответствующие им оценки авторитетности и посреднические оценки.

.

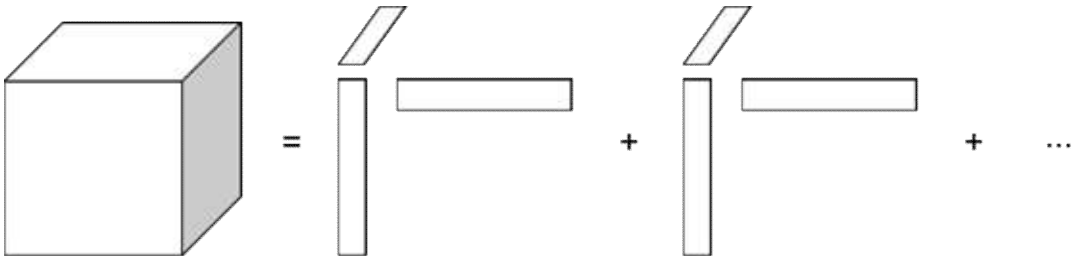


Рисунок 3 – Графическая иллюстрация тензорного разложения

Заключение. Задача ранжирования вершин семантических графов является актуальной в связи появлением больших архивов разнородных данных, таких как архивы поисковых систем, социальных сетей, систем коммуникаций. В настоящей работе описан алгоритм TOPHITS ранжирования вершин семантических графов и его реализация на основе алгоритма PARAFAC. В дальнейшем планируется разработка соответствующего

программного обеспечения.

Список литературы

1. Кузнецов И., Семантические представления/ И.П. Кузнецов, Е.В. Золотов - Москва: Наука, 1986. – 293 с.
2. Акивис А., Тензорное исчисление / М.А. Акивис, В.В. Гольдберг - Москва: Физматлит, 2003. – 304 с.
3. Kolda T., Higher-Order Web Link Analysis Using Multilinear Algebra / Kolda T., Bader B, Kenny J. // Proceedings of the Fifth IEEE International Conference on Data Mining. – Washington, 2005. - pp. 242-249.
4. Harshman R., Foundations of the PARAFAC procedure: models and conditions for an “explanatory” multimodal factor analysis / Harshman R. - UCLA working papers in phonetics, 1970. – pp 1-84c.
5. Kolda T., Tensor decompositions and applications / Kolda T., Bader B. // SIAM Review. - 2009. - №51(3) - pp. 455–500.

РАЗРАБОТКА ИНФОРМАЦИОННОГО ПО ИНТЕГРАЦИИ МИС ДИАГНОСТИЧЕСКОГО ЦЕНТРА АЛТАЙСКОГО КРАЯ С ЛИС АРИАДНА

Гавриченков Е.А. – студент, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский Государственный технический университет (г. Барнаул)

Описание терминологии

Медицинская информационная система (МИС) ДЦАК — система автоматизации документооборота, в которой объединены система поддержки принятия медицинских решений, электронные медицинские карты о пациентах, данные медицинских исследований в цифровой форме, данные мониторинга состояния пациента с медицинских приборов, средства общения между сотрудниками, финансовая и административная информация.

Лабораторная информационная система (ЛИС) — программное обеспечение, предназначенное для управления лабораторными потоками работ и документов.

Назначением ЛИС является получение достоверной информации по результатам исследований и оптимизации управления этой информацией с целью её использования для принятия корректных своевременных управленческих решений.

Назначение системы, интегрирующей МИС и ЛИС

МИС и ЛИС функционирую отдельно, независимо друг от друга, что создает большие трудности для медицинских учреждений. Поэтому в данной работе предлагается реализовать систему, интегрирующую их.

Функции разрабатываемой интегрирующей системы: [3]

1) Предоставлять интерфейс для формирования заказа на исследования.
2) Актуализировать справочную информацию, предоставляемую ЛИС и МИС.
3) Предоставлять интерфейс для сопоставления справочников, а также хранить эту информацию.
4) Преобразовывать заказы из терминов МИС в термины ЛИС с помощью сопоставлений.
5) Преобразовывать результаты из терминов ЛИС в термины МИС.

Диаграмма вариантов использования представлена на рисунке 1.

Врач или сотрудник регистратуры выписывают пациенту талон на сдачу биоматериала. Пациент в назначенное время приходит и сдает его, талон прикрепляется к контейнеру для идентификации биоматериала. Далее биоматериалы отсылаются в лабораторию для исследования. После проведения необходимых исследований результат возвращается в место сдачи или в то место, которое указано как получатель.

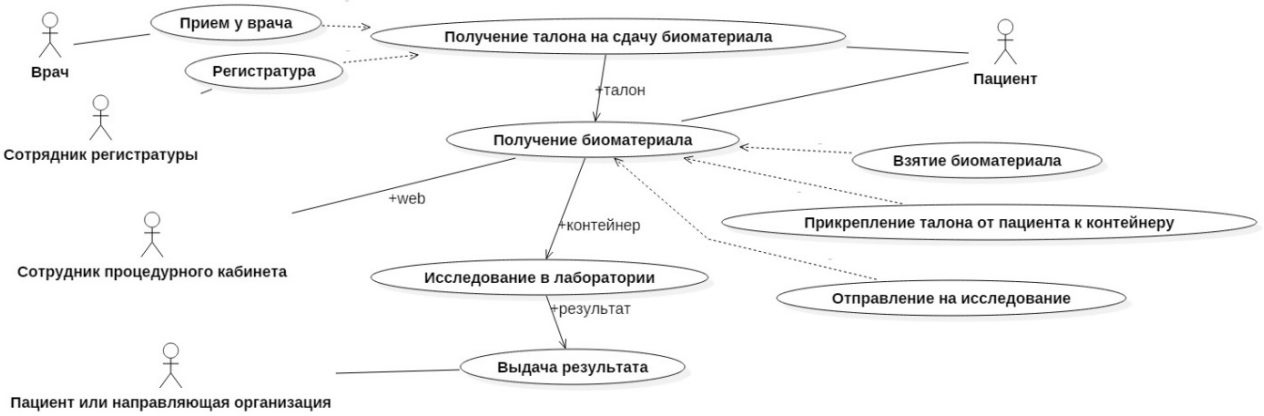


Рисунок 1 – Диаграмма вариантов использования

Диаграмма, описывающая информационные потоки, представлена на рисунке 2.

Для корректного проведения всего вышеописанного процесса необходима дополнительная справочная информация. Эта информация берется из ЛИС и МИС и обновляется по расписанию.

Однако эти системы имеют собственные, независимые друг от друга справочники, которые могут различаться между собой, а именно справочники услуг, организаций и категорий заказа. Из-за этого необходимо сопоставлять данные из ЛИС и МИС. Однако сопоставление — нетривиальный процесс, который невозможно автоматизировать. Поэтому сопоставлять данные будет администратор, для чего у него будет специальный интерфейс.

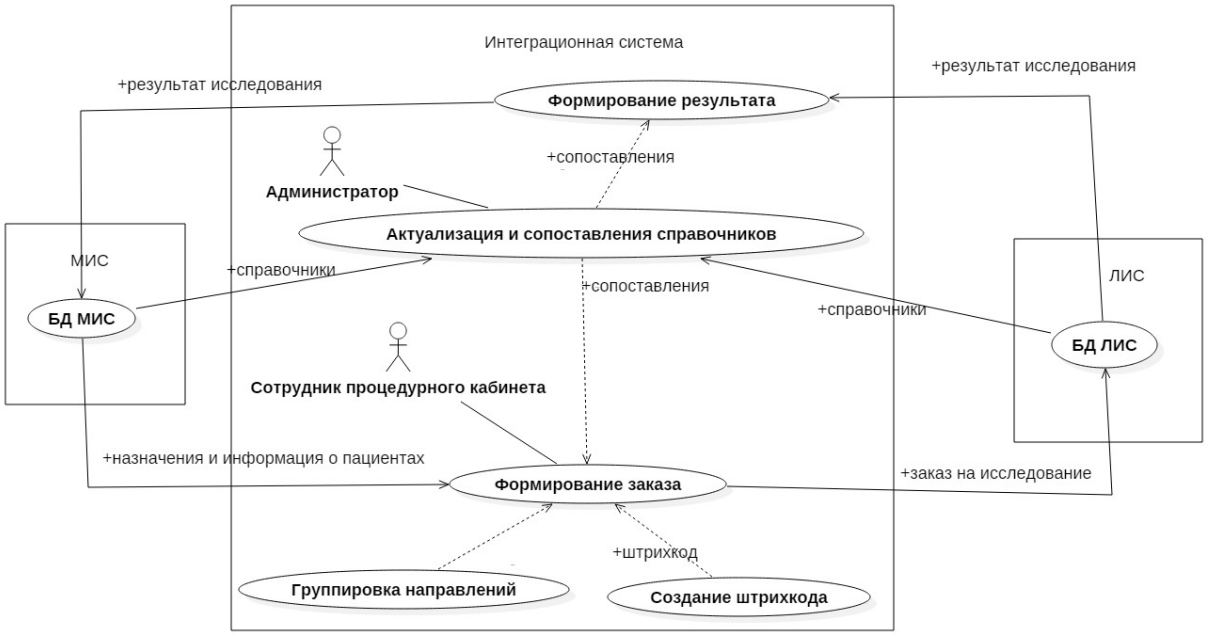


Рисунок 2 – описание информационных потоков

При выдаче талона МИС передает в интегрирующую систему направление, в котором

указываются некоторые данные о пациенте (ФИО, пол, возраст), а также данные об организации, в которой был совершен забор, организации, в которую необходимо вернуть результат, и т.п.

При заборе биоматериала сотрудник процедурного кабинета вносит в систему запись о биоматериале с указанием, какие исследования необходимо провести на ней. Также он может внести данные о состоянии пациента, например, указание о сроке беременности, принимает ли пациент антибиотики и т.п., так как эти данные влияют на интерпретацию результатов.

Далее эти данные передаются в ЛИС в виде заказа. Во время формирования заказа создается штрих код, по которому будет идентифицирован контейнер с биоматериалом.

После проведения исследований результаты возвращаются в интегрирующую систему, где проходят обратный процесс сопоставления. По завершению сопоставления результаты передаются в МИС.

Архитектура ПО

Диаграмма, описывающая архитектуру, представлена на рисунке 3.

Для реализации интегрирующей системы выбрана клиент-серверная архитектура.

Данные из МИС поступают в виде json-файлов, данные из ЛИС - из общей сетевой папки в виде XML-файлов. Таким образом, проектируются сервисы получения данных из соответствующих ИС, а также сервисы сериализации и десериализации. Теги всех файлов (справочников, направлений, заказов и результатов), а также структура общей папки, описаны в соответствующих спецификациях [2].

Для работы с БД разрабатываются специальные сервисы, унифицирующие добавление и редактирование. Удаление из БД данных система не производит для возможности впоследствии просмотреть все заказы и результаты.

Для работы с сопоставлением справочников предусмотрены специальные интерфейсы. Результаты сопоставлений записываются в БД, откуда потом будут браться при формировании заказов и результатов.

Клиентами являются браузеры компьютеров администратора и сотрудников процедурного кабинета.

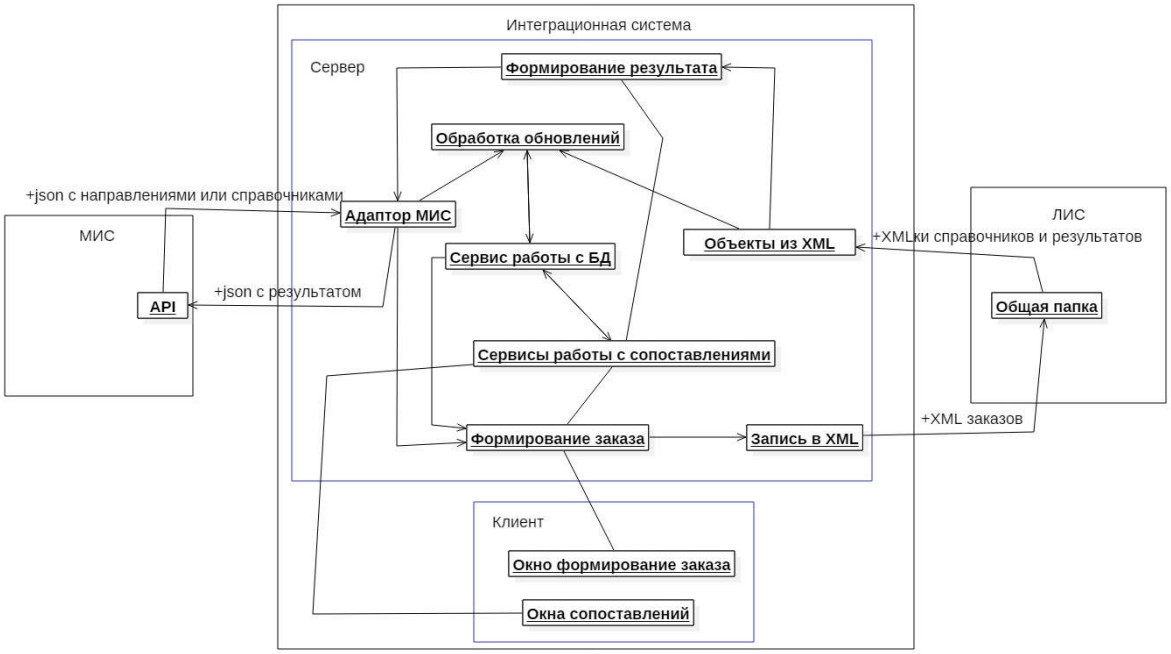


Рисунок 3 – архитектура приложения

Заключение

Предлагаемая система позволит уменьшить количество трудностей для медицинских учреждений, облегчить работу сотрудников процедурного кабинета. Также она позволит визуализировать сопоставление терминов, что уменьшит количество ошибок, связанных с разными справочниками МИС и ЛИС.

Список литературы

1. Health Level Seven International [Электронный ресурс]. — Режим доступа: http://www.hl7.org/index.cfm
2. Техническое задание на разработку интерфейса обмена данными между лабораторной информационной системой АРИАДНА в клинико-диагностической лаборатории и медицинской информационной системой [Текст]. - 2016.02.11. – 7 с.
3. Методические рекомендации по обеспечению функциональных возможностей медицинских информационных систем медицинских организаций (МИС МО) [Текст]: указание МИНЗДРАВ РФ от 05.02.2016. - 2016. – 83 с.

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ АВТОМАТИЗИРОВАННОЙ ЗАПИСИ ИНФОРМАЦИИ НА CD\DVD ДИСКИ

Каркавин М.А. – студент, Троицкий В.С. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Информационный обмен между предприятиями и организациями чаще всего происходит посредством телекоммуникационных сетей. Но в некоторых случаях, необходима передача именно материального носителя информации, например, по запросу судебных органов. Наиболее популярным носителем является оптический диск. Подготовкой передаваемых документов, их записью на оптический диск и собственно передачей занимаются различные сотрудники организации. Очевидно, что процесс записи оптических дисков это чисто техническая работа, которая может быть автоматизирована и из этой технологической цепочки сотрудник записывающий диски может быть исключен. Кроме собственно записи оптических дисков, предлагаемое нами программное обеспечение автоматизирует и сопутствующий ему документооборот.

Созданное в рамках данной работы ПО представляет собой специализированный WEB-сервис (Автоматизированная станция записи дисков). Сервер, где он развернут (монитор, системный блок с приводом DVD и принтер, клавиатура и мышь не нужны), предполагается разместить около секретаря организации, который будет его обслуживать и заниматься рассылкой записанных дисков.

Особенности ПО:

1. Для станции записи была выбрана операционная система MS WindowsServer 2008 R2;
2. Для рабочих мест пользователей была выбрана операционная система MS Windows 7;
3. По правам доступа пользователей к приложению используется ActiveDirectory;
4. Интерфейс станции записи разработан с учетом того, что есть только монитор (вывод информации, сколько дисков в очереди на запись и что необходимо сделать сейчас, привод DVD (собственно запись) и принтер (печать сопроводительного письма, описи и расписки), нет ни мыши, ни клавиатуры.

5. Так как пользователей много, а рекордер один,предусмотрен механизм очереди на запись дисков;
6. Все операции записываются во внутренние журналы (когда, кто и что сделал), а копии записанных файлов сохраняются в архиве;
7. Есть возможность просмотра этих журналов и файлов, возможность фильтрации, поиска, очистки за заданный период.
8. Интерфейс клиентской части предусматривает возможность записи файлов, просмотра состояния своих задач записи и историю записанных пользователем дисков.

Как работает приложение:

1. Пользователь заходит в веб-интерфейс приложения через браузер, и автоматически определяется кто зашел при помощи ActiveDirectory, выдается его список задач записи и история записанных пользователем дисков.
2. Пользователь создает новую задачу записи, формирует список файлов для записи на диск и сопроводительное письмо к ним и отправляет запрос на запись;
3. Подготовленная пользователем информация передается на станцию записи;
4. На станции записи, она помещается в очередь на запись дисков;
5. Станцией записи, в порядке очередности производится запись на диск, печать описи, расписки и сопроводительного письма к ним. Кроме того, она просит секретаря вставлять чистые диски и забирать уже записанные диски;
6. Секретарь отправляет записанные диски согласно сопроводительного письма к ним.

КОМПЛЕКС КЛИЕНТ-СЕВРЕНЫХ ПРИЛОЖЕНИЙ PRICE-CHECKER

Мальков С.А. – магистрант, Крючкова Е.Н. – к.ф-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В современном мире уровень технологичности с каждым годом повышается. Ведущие мировые торговые сети внедряют различные сервисы, чтобы привлечь больше клиентов, сделать их покупки проще, а товарооборот - эффективнее. Массовая реклама и продвижение товаров и услуг по каналу мобильной связи успешно применяются торговыми сетями.

Анализ уровня внедрения современных технологий на предприятиях России показал, что мобильный маркетинг находится на стадии развития, и текущие решения, применяемые в сфере розничной торговли не оптимальны. Предлагаемое решение позволяет увеличить эффективность мобильного маркетинга и вывести его на новый уровень, ныне не достигнутый.

Структура системы

На текущий момент реализована клиент-серверная система, в которой серверная часть состоит из двух серверов, один – это сервер Price Checker, иначе сервер статистики, взаимодействующий с базой данных статистики и проксирующий запросы от клиентского приложения на кассовый сервер и обратно, второй – кассовый сервер, взаимодействующий с базой данных товаров. Второй сервер является упрощенным вариантом реального кассового сервера, работающего в торговой точке. Кроме того, установлен JasperReports сервер, взаимодействующий с базой данных статистки, и разработаны формы отчетов для данного сервера.

Топ товаров

Количество	Штрих-код	Наименование	Цена
276	4607043790149	БАТОН НАРЕЗНОЙ 1С 350Г	27.6
155	2500000183195	СГУЩЕННОЕ МОЛОКО КОРОВКИНО ЦЕЛЬНОЕ С САХАРОМ 280Г ДОЙ/ПАК	44.9
98	4600536001005	ВОДА НАРЗАН МИНЕРАЛЬНАЯ ПРИРОДНЫЙ ГАЗ. 0.33Л ПЭТ	24.0
41	4605209000293	ВАНИЛИН 1Г	5.2
40	4607161625071	МОЛОКО ПРОСТОКВАШИНО ТОПЛЕНОЕ 4% 930МЛ	62.6

Рисунок 1 – Отчет топ товаров

Клиентская часть представлена мобильным приложением, работающим под управлением операционной системы android, реализующим следующие функции:

- сканирования штрих-кодов товаров;
- распознавания штрих-кодов;
- предоставление информации об отсканированном штрих-коде;
- подбор товаров в корзину с подсчетом стоимости покупки.

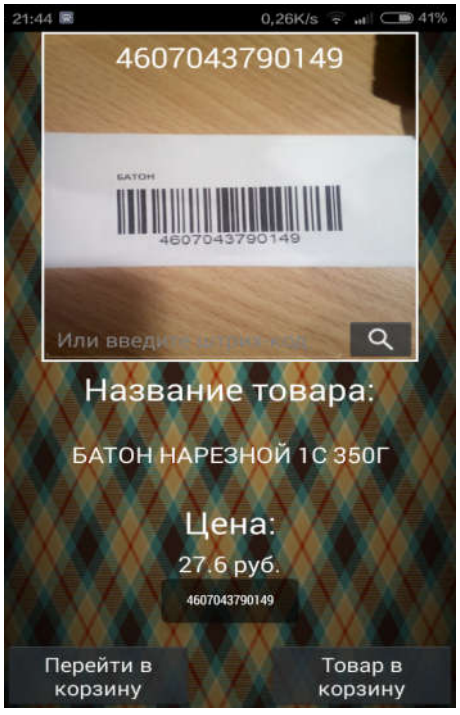


Рисунок 2 – Отсканирован товар

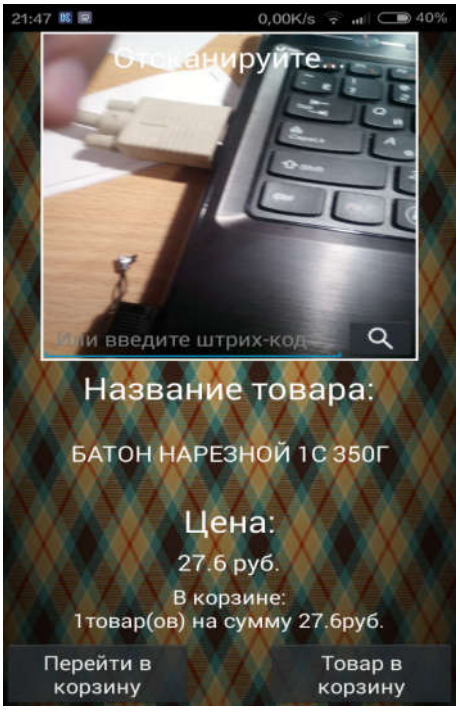


Рисунок 3 – Добавлен товар в корзину

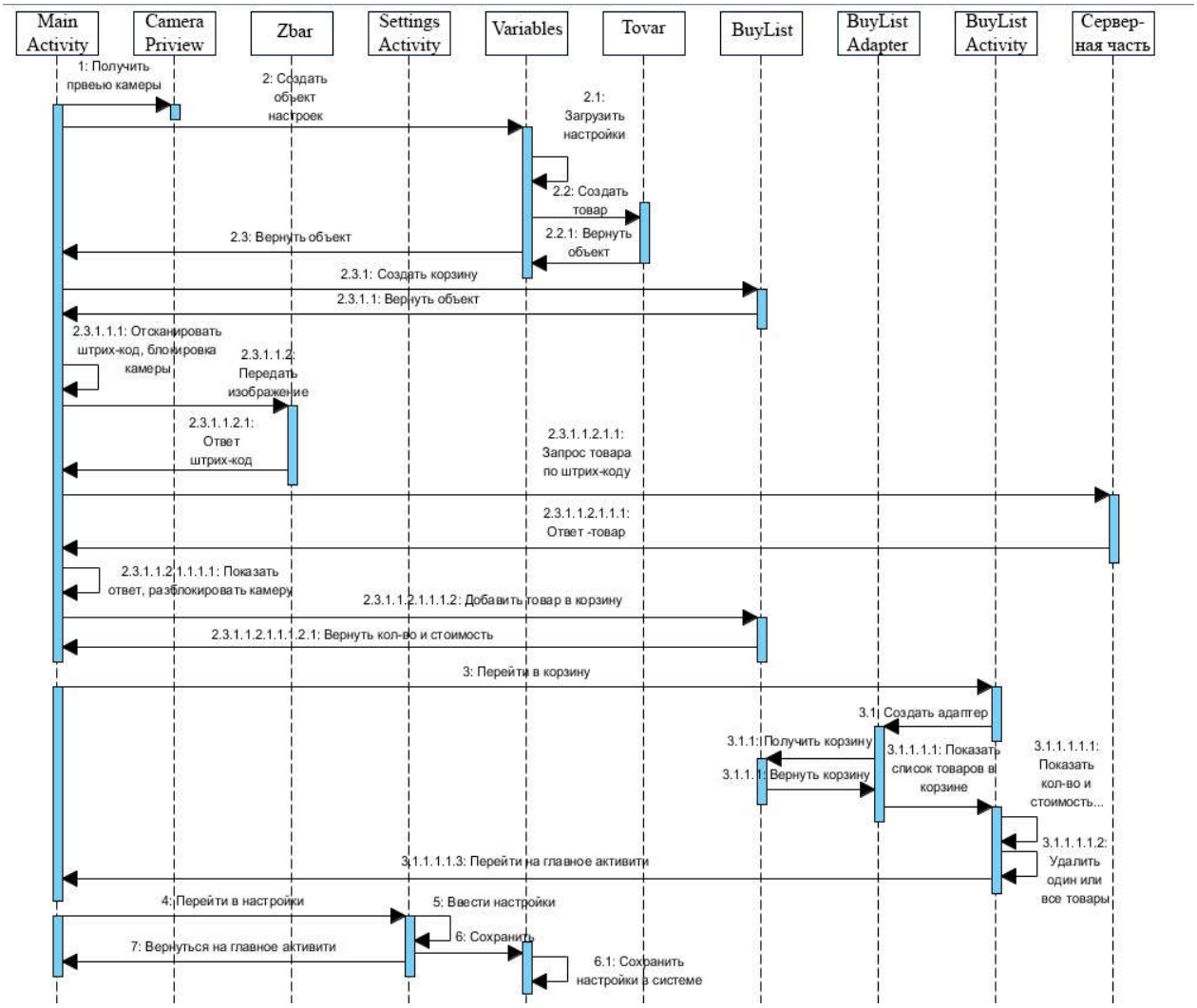


Рисунок 4 – Линии жизни клиентского приложения

Клиентская и серверная части написаны на языке программирования Java в средах разработки Android Studio и NetBeans с использованием технологий и средств разработки:

- Spring Framework;
- протокол Representational State Transfer (REST);
- библиотека Zbar;
- Jackson JSON Processor.

Разработанная система предназначена для внедрения в уже имеющуюся систему программного обеспечения торговых предприятий компании «ООО Ритейл Сервис».

Перспективы

Разработанная система имеет большие возможности к дальнейшему расширению. В дальнейшем может быть реализовано:

- Идентификация пользователей;
- Отображение акций, проводимых в магазине;
- Отображение рекламных предложений;
- Предложения связанных товаров;
- Пользовательская статистика;
- Индивидуальные предложения;
- Расширение предоставляемой информации о товарах;

- Различного рода реклама.

Список литературы

1. Статья «Мобильные приложения – модный тренд или необходимость?» [Электронный ресурс]. // RETAIL. – Режим доступа: http://www.retail.ru/interviews/79960, свободный.
2. Статья «Мобильные технологии на вооружении торговых продовольственных сетей» [Электронный ресурс]. // ADVERTOLOGY. – Режим доступа: http://www.advertology.ru/article124100.htm, свободный.
3. Голощапов, А. Google Android. Создание приложений для смартфонов и планшетных ПК [Текст] / А. Голощапов. — БХВ-Петербург. — 928 с.
4. Статья «Сканеры штрих кода» [Электронный ресурс]. //SCANCODE. – Режим доступа: http://www.scancode.ru/catalog/2, свободный.

СИСТЕМА КОМПЬЮТЕРНОГО ЗРЕНИЯ ДЛЯ АНАЛИЗА МИКРОФОТОГРАФИЙ СВАРНЫХ СОЕДИНЕНИЙ

Мягков В.В. – магистрант, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Компьютерный анализ и моделирование играют большую роль при проведении научных исследований. Исключением не является и область материаловедения. Ручной анализ микрофотографий весьма трудоемок или даже невозможен в случае достаточно большой выборки изображений. В связи с этим актуальную проблему представляют исследования, направленные на автоматизацию изучения структуры материалов и адаптацию методик параметризации этих структур.

Для изучения структуры материалов и исследования их эволюции эффективно используется аппарат фрактальной геометрии[1]. Вопрос о том, с какими физическими характеристиками и механизмами связаны фрактальные характеристики, не вполне очевиден, однако фрактальная размерность материала должна определять некоторые свойства материала. В связи с этим важной задачей является определение фрактальной размерности.

Целью данной работы является разработка системы расчета фрактальной размерности сварного соединения для микрофотографии целиком или же для выбранной пользователем части микрофотографии.

Основная последовательность работы системы следующая:

1. Пользователь системы выделяет часть микрофотографии, которая подвергнется дальнейшему анализу.
2. Производится кластеризация - выделение связных областей, занимаемых различными веществами, участвующими в сплаве.
3. Для связных областей, полученных на шаге 1, выделяются контуры.
4. По контурам рассчитывается фрактальная размерность.

Пользователь, используя настройки системы, может выбрать какой метод кластеризации применяется к изображению. На данный момент доступными вариантами являются методы Otsu Thresholding[2] и метод адаптивной бинаризации[3].

Метод Otsu Thresholding анализирует обрабатываемое изображение и автоматически вычисляет порог бинаризации - единый для всей микрофотографии. Метод использует гистограмму распределения значений яркости пикселей растрового изображения, которая рассчитывается на основе количества пикселей в группе с

определенным значением пикселя, для минимизации внутриклассовой дисперсии. Найденный порог подаётся на вход обыкновенной пороговой бинаризации[4].

Метод адаптивной бинаризации рассматривает значение не в одном пикселе, а в некоторой его окрестности. Это значение может быть просто средним значением пикселей окрестности(т.е. все пиксели равнозначны), либо пиксели окрестности умножаются на весовой коэффициент (взвешиваются) в соответствии с функцией, например с гауссовой функцией. Если значение оцениваемого пикселя больше рассчитанного для окрестности значения, то пикселю присваивается значение 1, иначе - 0.

Результат работы методов Otsu Thresholding и адаптивной бинаризации для исходного изображения, представленного на рисунке 1, можно видеть на рисунке 2.

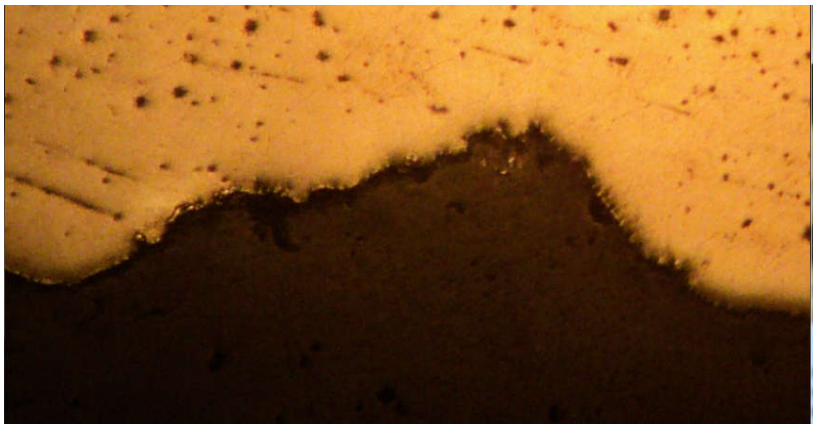


Рисунок 1 – Исходное изображение

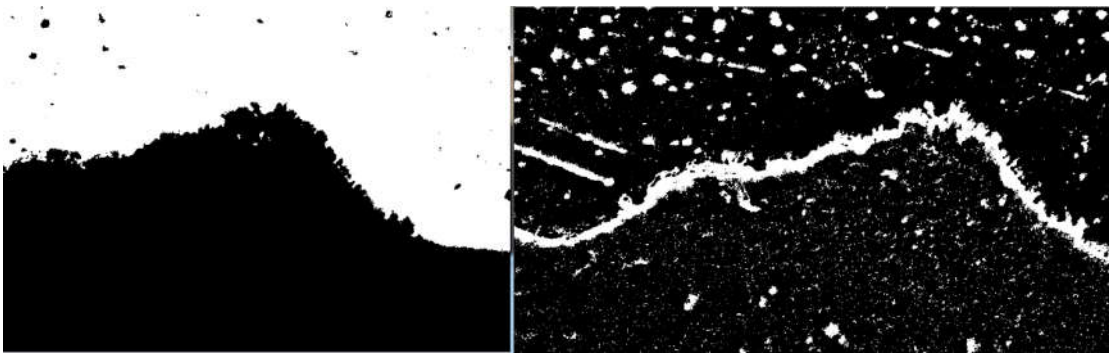


Рисунок 2 – Результат работы методов Otsu Thresholding и адаптивной бинаризации

Для нахождения контуров по полученным областям веществ был выбран алгоритм Topological Analysis by Border Following [5], как хорошо себя зарекомендовавший для подобных задач. На вход алгоритму поступает полученное на этапе кластеризации бинарное изображение, то есть изображение, каждый пиксель которого имеет значение нуля или единицы.

Далее производится построчное сканирование изображения, начиная с верхнего левого угла, до тех пор, пока не будет найден не равный нулю пиксель. Начиная с этого пикселя, контур достраивается путём просмотра соседних с текущим пикселей против часовой стрелки. Когда очередной контур достроен, построчное сканирование продолжается до тех пор, пока не будет достигнут правый нижний угол изображения

или не будет найден очередной не равный нулю пиксель. Результат работы алгоритма оконтуривания можно видеть на рисунке 3.

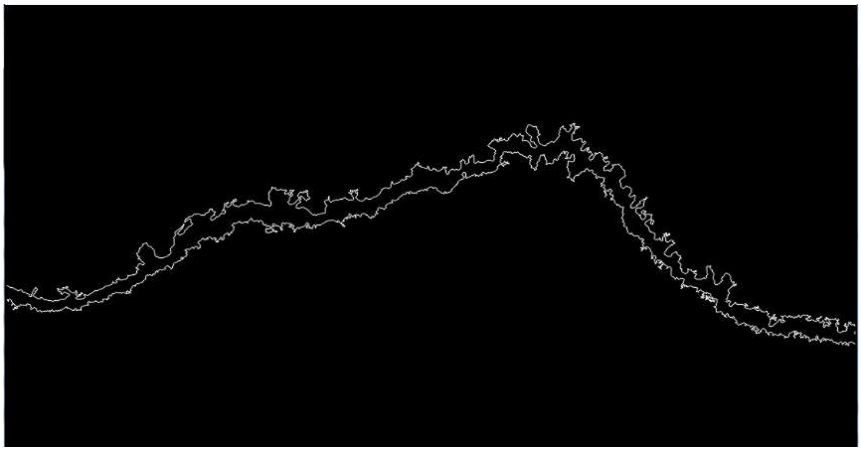


Рисунок 3 – Результат работы алгоритма Topological Analysis by Border Following

Случается, что не все из найденных контуров являются искомыми. Пользователь может отсеять такие выбросы простым щелчком мыши. Процесс такого отсеивания представлен на рисунке 4.

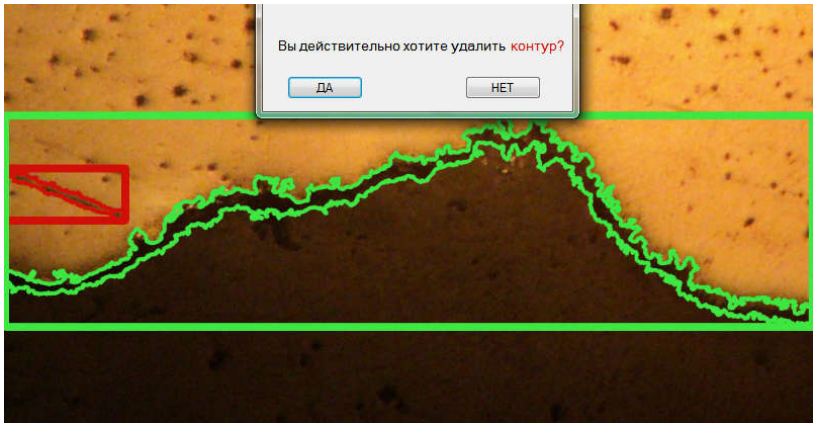


Рисунок 4 – Отсеивание ложных контуров

Заключительный шаг работы системы состоит в том, чтобы по выделенным контурам определить фрактальную размерность сварного соединения. Для вычисления фрактальной размерности используется алгоритм box-counting.[6].

Вычисление фрактальной размерности этим алгоритмом сводится к следующей процедуре:

1. Объект покрывается квадратной сеткой с ячейками известного размера
2. Подсчитывается количество ячеек, которые содержат ненулевые значения. Сохраняется пара значений (log(1/длина стороны ячейки), количество ячеек содержащих объект).
3. Сетка детализируется - т.е. размер ячеек уменьшается, и, соответственно, количество ячеек, содержащих объект, увеличивается. Сохраняется новая пара значений.
   Процедура детализации повторяется многократно.

По полученным парам значений производится построение линии регрессии. Значение углового коэффициента этой линии и будет искомой фрактальной размерностью.

В результате исследования был разработан программный комплекс, позволяющий рассчитывать фрактальную размерность сварного соединения на микрофотографии целиком или её избранной части. Планируется дальнейшая работа над поставленными задачи, испытание дополнительных методов кластеризации, обработки изображений и улучшение качества работы алгоритма.

Список литературы

1. Мандельброт Б. Фрактальная геометрия природы [Текст] / Б. Мандельброт. – М.: ИКИ, 2002. – 654 с.
2. Otsu N. A threshold selection method from gray-level histograms. IEEE Trans. Sys., 1979, Cyber. 9: pp. 62-66
3. OpenCV: Miscellaneous image transformation[Электронный ресурс]. // Scribd– 2014. Режим доступа: http://docs.opencv.org/modules/imgproc/doc/miscellaneous_transformations.html, свободный.
4. В Computer Science & Engineering[Электронный ресурс].// University of Washington - 2016. – Режим доступа: https://courses.cs.washington.edu/courses/cse576/book/ch3.pdf
5. Suzuki S. and Abe K. Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30 1, 1985, pp. 32-46.
6. Iannaccone K. Fractal Geometry in Biological Systems. CRC Press, p. 143

РАЗРАБОТКА УНИВЕРСАЛЬНОГО ВИРТУАЛЬНОГО ТЕРМИНАЛА ОПЛАТЫ ПО ПЛАСТИКОВЫМ КАРТАМ

Шахов Д.Е. – студент, Троицкий В.С. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Оплата услуг и товаров в сети интернет по пластиковым картам сегодня очень популярна. Пользователь выбирает товар или услугу, и производит оплату посредством международной пластиковой карты. Достаточно ввести необходимые реквизиты карты, и покупка совершена. Для оплаты товаров и услуг посредством международной пластиковой карты сайт магазина должен уметь взаимодействовать с процессинговым центром. Это процедура хоть и не сложна, но требует написания соответствующего программного кода, выполнения необходимых доработок на сайте магазина, обеспечения требований платежной системы в плане информационной безопасности. Для крупных магазинов, это не является проблемой, а вот небольшие организации позволить себе этого не могут. Упростить процесс оплаты не представляется возможным для классической схемы взаимодействия процессинговый центр-банк-клиент-магазин. При этой схеме все механизмы направлены на обеспечение безопасности оплаты, в частности на то, чтобы магазин получил достоверную информацию о факте оплаты в момент проведения платежа, т.е. сразу же после окончания проведения транзакции. Но существует множество случаев, когда магазину желательно предоставить покупателю возможность оплаты товара через интернет, но нет необходимости узнать о факте оплаты немедленно, а достаточно получить информацию о ней через некоторое время, например, в течение суток. Примером таких взаимоотношений является заказ номера в гостинице или покупка путевки в санаторий, покупка билета в кино на завтра или заказ товаров, доставка которых будет осуществляться почтой или транспортной компанией (т.к. магазин передает товары на доставку, только через несколько дней после оформления заказа). В этих случаях предоставление товара или услуги происходит не сразу, а с некоторой задержкой по времени и можно существенно упростить схему взаимодействия

для магазина. А именно, можно перенести работу виртуального терминала на территорию банка, и он будет один для множества магазинов. Таким образом, сайт магазина не будет напрямую взаимодействовать с процессингом, а будет переадресовывать клиента на сайт банка через упрощенный интерфейс взаимодействия. Информация о платежах клиентов будет доставляться магазину позже, с помощью специальных каналов связи, например, через систему Банк-Клиент. При такой организации процесса оплаты через интернет, магазину достаточно будет добавить упрощенный интерфейс взаимодействия на страницу своего сайта.

В данной работе описанная выше система спроектирована и создана. Она представляет собой специализированный веб-сервис, функционирующий на веб-сервере IIS 7 и позволяющий проводить платежи посредством упрощенного интерфейса взаимодействия. Инициировать платеж возможно из любого электронного документа (веб-страница, pdf-файл, документ Microsoft Office) или через гиперссылку, которую Вам прислали по электронной почте, мессенджеру или социальной сети. В программе предусмотрены механизмы обеспечения информационной безопасности, журнализация выполняемых операций и выпуск необходимых отчетов. На сегодняшний день сервис развернут на площадке ООО КБ Алтайкапиталбанк для тестовой эксплуатации.

ПРИМЕНЕНИЕ РАСПРЕДЕЛЕННОГО АЛГОРИТМА ВЫБОРА ЛИДЕРА ДЛЯ ЗАДАЧИ НАДЕЖНОЙ КОММУНИКАЦИИ

Ложкина Д.Д. – студент, Старолетов С.М. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Актуальность

В настоящее время при построении автоматизированных систем управления различного назначения широкое распространение получили распределенные вычислительные системы, одной из основных проблем которых остается задача обеспечения их продолжительного функционирования: высокого уровня отказоустойчивости и производительности с целью решения задач реального времени. Решение данной проблемы особенно актуально для бизнеса, где ошибка в работе системы или временной ее простой оборачивается более чем внушительными финансовыми потерями.

Для повышения надежности информационно-вычислительной системы идеальной схемой являются кластерные системы. Благодаря единому представлению отдельные неисправные узлы или компоненты кластера могут быть заменены без остановки работы и незаметно для пользователя, что обеспечивает непрерывность и безотказную работу вычислительной системы даже в таких сложных приложениях как базы данных [3].

Постановка задачи

В частности, такая отказоустойчивая вычислительная система может быть использована для решения задачи надежной коммуникации. Представленный продукт будет представлять чат между пользователями сети, решение которого основано на применении протокола выбора лидера. Основная идея чата: все пользователи взаимодействуют друг с другом через сервер, которым является один из участников чата. При возникновении неполадок роль сервера берет на себя другой пользователь, смена происходит незаметно для участников, работа чата продолжается без сбоев, все данные (переданные сообщения, информация о пользователях в сети) остаются доступны и не происходит их потери. Такое решение обеспечивает гарантированную передачу сообщений между пользователями, что зачастую

может быть очень важно, а также возможность отказа от существования отдельной машины, выполняемой роль сервера.

Основная схема работы

Первый пользователь, вошедший в сеть, принимает на себя роль сервера. Остальные пользователи подключаются к уже исполняющему свои функции серверу, образуя кластер. При возникновении сбоев на связывающей машине, какой-либо участник берет на себя обязанности сервера, сообщает об этом остальным участникам, и работа чата продолжается.

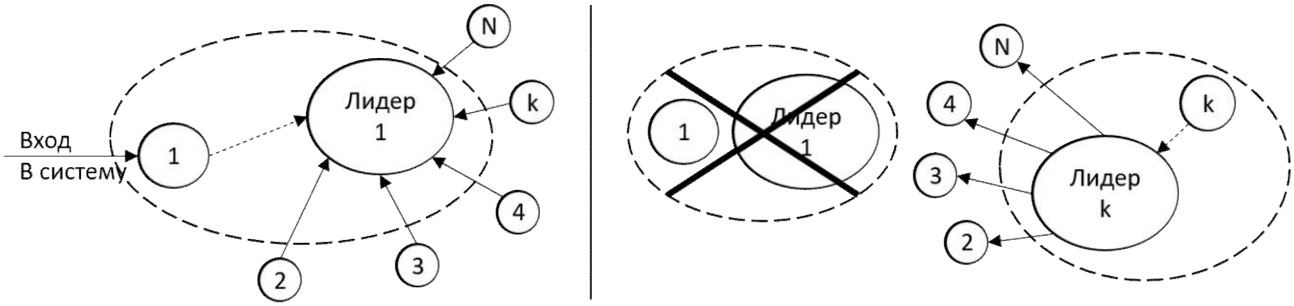


Рисунок 1 – Основная схема работы

Предложенный метод решения задачи

Алгоритм выбора лидера состоит в том, что распределенные процессы должны выбрать один процесс для координации их параллельной активности [1] в течение определенного отрезка времени. Для решения данной задачи были проанализированы различные алгоритмы, такие как Paxos, Raft, Viewstamped replication, и был выбран алгоритм Raft, предназначенный для решения задач консенсуса в сети надёжных вычислений, по следующим причинам. Raft разрабатывался с учётом недостатков более старого алгоритма Paxos [4]. При выборе ключевых идей, предпочтение отдавалось более простым и практичным решениям, делающим алгоритм эффективней и понятней: меньшее количество состояний и запросов, строго последовательная запись журнала, двухэтапный подход при изменении конфигурации кластера, минимальные действия со стороны обычных участников.

Основы алгоритма

Raft строится поверх кластера, на каждом из узлов которого работает определенная машина состояний. Обеспечивая надежную доставку сигналов на все узлы в заданном порядке, осуществляется переход всех машин состояний по одним и тем же последовательностям состояний. Таким образом, каждый узел гарантированно приходит в согласие с другими узлами.

Каждая машина может находиться в одном из трех состояний:

- Лидер (Leader)
  - принимает запросы от клиентов;
  - реплицирует журнал на всем кластере
  - сообщает участникам, когда можно безопасно применить ту или иную запись к их конечному автомату
  - поддерживает свою власть с помощью постоянной рассылки сообщений каждому участнику
- Кандидат (Candidate).
  - Инициирует выборы лидера
- Участник (Follower).

- Пассивен: не инициирует никаких RPC, только отвечает на запросы лидеров.

Таким образом участники следуют за лидером и не взаимодействуют между собой (за исключением фазы голосования).

Особенностью данного алгоритма является то, что управление кластером осуществляется с помощью двух видов запросов, а также оно разделено на две фазы:

1) Выборы лидера (голосование)

2) Репликация журнала (передача данных протокола) – нормальный режим работы

Все время работы разделено на периоды, каждый период начинается с выборов.

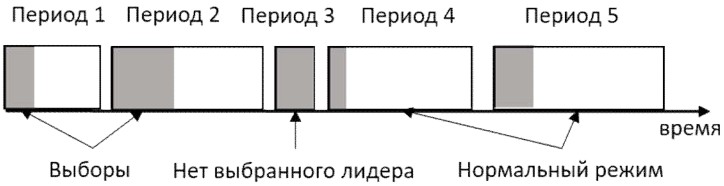


Рисунок 2 – Разделение фаз

Выборы лидера

Если какой-либо участник не получил в течение установленного времени (время выборов) сообщение от лидера, он считает, что лидер вышел из системы, переходит в состояние кандидата и отправляет всем участникам кластера запрос на начало голосования. Участники голосуют (единожды за один период времени) за кандидата, который имеет наиболее актуальный журнал (свойство безопасности Safety). Если кандидат набирает большинство голосов ($\frac{N}{2}+1$, где N – количество машин в кластере), он переходит в состояние лидера, оповещает об этом кластер и начинает процесс репликации журнала, иначе возвращается в состояние участника. Процедура голосования повторяется, пока не будет выбран лидер. Raft гарантирует, что в каждый момент времени возможно существование не более одного лидера (свойство безопасности Election Safety).

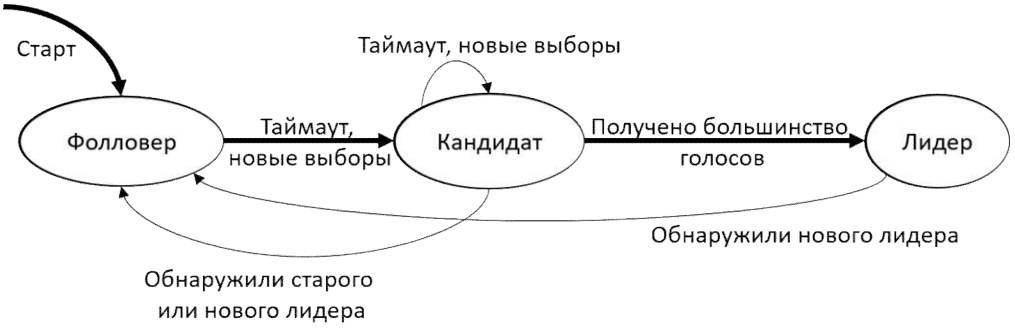


Рисунок 3 – Диаграмма состояний

Голосование опирается на случайные ожидания (время выборов). Диапазон допустимых значений определяется следующим образом:

$$broadcastTime \le electionTimeout \le MTBF ,$$

где broadcastTime - время трансляции; electionTimeout – время выборов; MTBF - среднее время безотказной работы. Время взаимодействия участников обычно измеряется миллисекундами (0,5 – 20 мс), а время нормальной работы между отказами — месяцами, поэтому на практике такое соотношение достигается легко: достаточно взять значение electionTimeout в промежутке 100…500 мс.

Репликация журнала

Лидер полностью отвечает за правильную репликацию протоколов. Он отправляет всем участникам запрос на добавление новой записи и считает транзакцию успешной только после того, как большинство узлов подтвердило, что данные были применены [4]. При этом, записи в журнале идут строго последовательно. По номерам записи определяется степень актуальности состояния узла, что имеет влияние при выборе нового лидера. Таким образом лидер никогда не перезапишет или удалит записи в журнале: обеспечивается высокая степень синхронизации журналов участников.

Индекс	1	2	3	4	5	6	7	8
Номер периода	**1**	**1**	**1**	**2**	**3**	**3**	**4**	**4**
Команда	add	rem	msg	add	rem	add	rem	msg

Рисунок 4 – Формат журнала

Особенности реализации

Реализация осуществляется на языке Java с использованием фреймворка Erlang/OTP (модель рабочих и главных процессов, предназначена для написания отказоустойчивых высоконагруженных приложений). Высокую надежность приложений обеспечивает встроенная обработка исключений, что делает код намного короче, а высокую эффективность – параллельные процессы. Erlang позволяет выполнять большое количество процессов одновременно, что создает хорошую модель многопоточности. Также есть свои средства для построения кластеров и общения между узлами.

Работа чата:

1) Для каждого участника чата создается узел и почтовый ящик, имеющий свой уникальный идентификатор, на котором хранятся (в порядке поступления) все полученные сообщения. На каждом узле хранится словарь, содержащий имена и идентификаторы всех участников. Когда участник переходит в состояние лидера, создается еще один ящик, регистрируемый под именем server, чтобы остальные пользователи могли подключиться к нему, отправив запрос на добавление в кластер.

2) Все сообщения передаются асинхронно [2]. Причем это не только запросы от лидера и ответы на него участников, а также сообщения с формы от одного пользователя другому. Процессы приема сообщений участника и лидера выполняются параллельно и независимо друг от друга. При отправке с формы текст этого сообщения и имя адресата передается через почтовый ящик серверу, а тот в свою очередь передает его другому пользователю.

3) При добавлении или удалении участников, лидер рассылаем всему кластеру запрос, содержащий соответствующую команду: add или rem. После успешной репликации участник применяет эту запись к своему конечному автомату, изменяя свой словарь и обновляя список участников на форме.

Для работы с объектами Erlang из Java используется JInterface (Java Interface). Это дает возможность создавать распределенную взаимодействующую систему, не используя ни одной строки сетевого кода, а также работать со структурами данных Erlang в ООП стиле. В таком случае мы можем реализовать логику вычисления на Erlang, а на Java - интерфейс ввода данных и визуализации результатов [5].

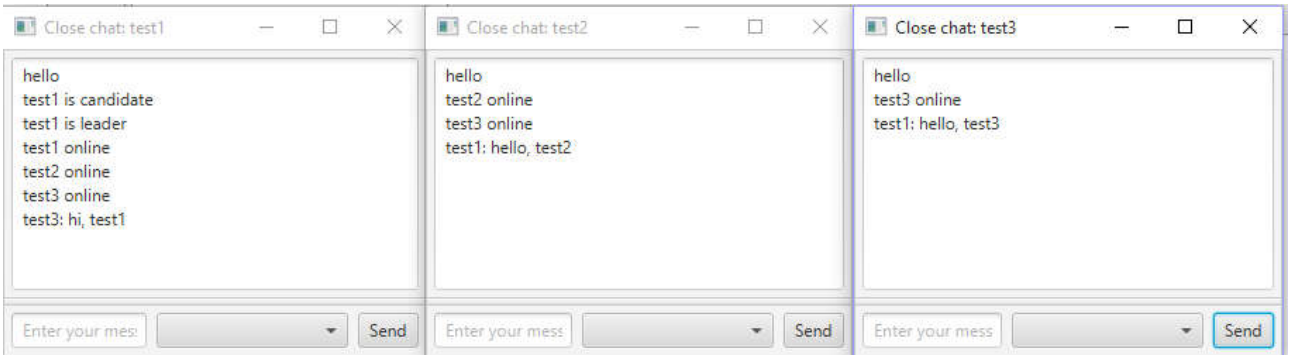


Рисунок 5 – Работа чата: порядок входа в сеть: test1 -> test2 -> test3

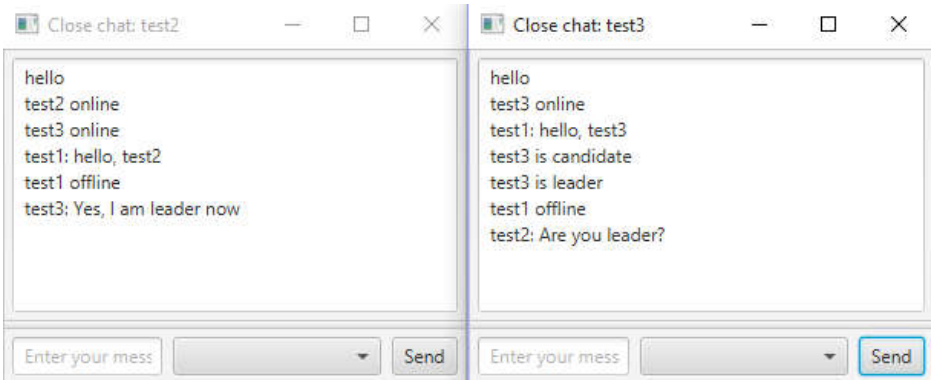


Рисунок 6 – Работа чата: test1 вышел из сети, test3 стал лидером

Заключение

На примере реализации чата было рассмотрено применение алгоритма выбора лидера для создания надежных вычислительных систем. Данное решение может быть применено для решения других прикладных задач, где необходимо обеспечить продолжительное функционирование системы и сохранность данных.

Список литературы

1. Model checking. Верификация параллельных и распределенных программных систем [текст]:Ю.Г. Карпов. - BHV, 2010. – 560 с.
2. Программирование в Erlang [текст]: С.Томпсон, Ф.Чезарини, А.О. Холомьева.- Москва:ДМК Пресс, 2012. – 488 с.
3. Надежность и отказоустойчивость МВС [электронный ресурс]: Режим доступа: http://www.intuit.ru/studies/courses/45/45/lecture/1358 - свободный.
4. In Search of an Understandable Consensus Algorithm [текст]: D. Ongaro, J. Ousterhout.- Stanford University. - 2014. -18 с.
5. Функциональные языки распределённых систем: Учебно-методическое пособие /С. М. Старолетов.- Барнаул : АлтГТУ, 2015 - 81 с.

ИСПОЛЬЗОВАНИЕ KOTLIN ДЛЯ ОПИСАНИЯ ДЕРЕВЬЕВ ПОВЕДЕНИЯ В ЦЕЛЯХ МОДЕЛИРОВАНИЯ ИИ В ОБЛАСТИ ОБУЧАЮЩЕГО ПО

Никитин А.А. – магистрант, Старолетов С.М. – к.ф.-м.н, доцент
Алтайский государственный технический университет (г. Барнаул)

Kotlin – статически типизированный язык программирования, компилирующийся в байткод для исполнения на JVM, разрабатывается компанией JetBrains с 2010 года. Этим языком поддерживается динамическая инжекция функций (далее расширение класса) [2], которая позволяет расширить функциональность класса путем добавления новых методов в класс. Для примера рассмотрим новый метод `hello()` для уже существующего класса `A`:

```
class A // определили класс
fun A.hello() = “Hello!” // инжекция метода в класс
fun main(args: Array<String>) {
    val a = A()
    println(a.hello()) // “Hello!”
}
```

В данном примере, метод `hello()` не является изначально созданным методом класса. Он создан как расширение для класса `A`. Расширение может быть описано в пределах некоторой области видимости:

```
class A(var message: String = “Hello”) // определили класс с одним
                                       // текстовым полем
fun initA(init: A.() -> Unit): A { // параметр функции является
                                   // расширением класса A
    val a = A()
    a.init() // вызываем определенное в данном контексте расширение
    return a
}
fun main(args: Array<String>) {
    val a = initA {
          message += “, world!” // this указывает на объект A
      }
      println(a.message) // “Hello, world!”
}
```

Здесь параметр функции `initA` является расширением для класса `A`, которое доступно для объектов класса `A` только в пределах функции `initA`. В главном методе программы этот параметр задается, используя лямбда-выражение, которое подменяет строковое поле message у объектов класса `A`. Следует заметить, что указатель `this` в теле лямбда-выражения, которое является расширением класса, указывает на объект этого класса.

Предметно-ориентированный язык (далее DSL) – язык программирования, специализированный для конкретной области применения [4]. Таковыми являются SQL для построения запросов к базе данных, HTML для разметки веб-страниц, TeX/LaTeX для разметки документов, и др.[3]. DSL в области деревьев поведения решает задачу построения стратегии.

Используя механизм расширений Kotlin, инициализацию класса можно реализовать в виде DSL, что позволит упростить синтаксис и облегчить чтение программы. На основе библиотеки [1] для создания деревьев поведения мы реализовали DSL с помощью Kotlin:

```
new Sequence(new UserAction(
  new Consumer(tick -> {
  System.out.println(“Hello!”);
  })
));
```

```
_sequence {
  userAction { tick, parent ->
        println(“Hello”)
  }
}
```

Слева пример создания дерева поведения с использованием Java 1.8, справа пример создания идентичного фрагмента дерева поведения с использованием Kotlin/DSL. Отсутствие операторов new, точек с запятой, круглых скобочек помогает, на наш взгляд, восприятию структуры дерева поведения и сокращает код, а также позволяет проще генерировать код средствами визуального редактирования.

Использование Kotlin для создания предметно ориентированного языка для организации деревьев поведения позволяет оперировать его богатым синтаксисом непосредственно в процессе строения стратегии, делая этот процесс динамическим. Для того, чтобы добавить возможность перестроения дерева поведения в процессе принятия решения, мы внесли средство отложенного добавления и удаления вершин:

```
fun main(agrs: Array<String>) {
    val root = _sequence { // корень дерева – композит-последовтельность
        userAction { tick, parent -> // единственной дочерней вершиной
                                     // будет являться вершина-действие
            parent.addChild(_userAction { // в процессе исполнения
                                          // добавим к списку детей
                                          // корневой вершины еще одну
                                          // вершину-действие
                action = Consumer {
                    println(uuid)
                    parent.removeChild(this)}})}} //, которая удалит себя
                                                  // при исполнении
    println(root.children.size) // 1 – на данный момент количество
                               // дочерних вершин у корня дерева поведения
    val bt = BehaviorTree(root) // создадим дерево поведения
    val bb = Blackboard() // создадим Blackboard
    bt.execute(bb) // первый userAction добавит еще один
                   // userAction к sequence
    println(root.children.size) // 2 – после первого исполнения дерева
                                // поведения количество дочерних вершин
                                // корня увеличилось
    bt.execute(bb) // первый userAction сделает то же самое, а второй
                   // userAction выведет свой id и удалит
                   // себя из списка дочерних вершин корня
    println(root.children.size) // 2

    bt.execute(bb) // --//--//--
    println(root.children.size) // 2
}
```

Таким образом, в данном примере показана возможность динамической реорганизации дерева поведения в процессе исполнения. Это совершенно новый подход к оперированию стратегиями, который позволит моделировать ИИ. Планируется использовать данную возможность для адаптивного построения стратегии обучения детей с речевыми отклонениями в обучающей системе. Нами в сотрудничестве с Алтайским педагогическим университетом разрабатываются методики корректировки некоторых дефектов произношения с учетом применения таковых в мобильных устройствах: планшетах и смартфонах. В таких условиях, система должна, относительно прогресса обучаемого и динамики восстановления, выбирать ту или иную стратегию обучения.

Список литературы

1. Никитин А.А, Старолетов С.М. Искусственный интеллект в моделировании поведения игровых персонажей на основе подхода «Behavior Tree» / А.А. Никитин, С.М. Старолетов // Сборник трудов конференц. «Наука и молодежь – 2015». Секция «Информационные технологии». Подсекция «Программная инженерия». – Барнаул, 2015 . – С. 33–37.
2. Исакова С. Kotlin in Action / С. Исакова, Д. Емеров . – 2015. – 325 с.
3. Ward M.P. Language Oriented Programming / M.P. Ward // Computer Science Department, Science Labs . – Durham . – 1994
4. Walid T. Domain-Specific Langianges / T. Walid // Department of Computer Science, Rice University . – Houston . - 2008

РАЗРАБОТКА ВЕБ-ПОРТАЛА АГРЕГАЦИИ КУЛЬТУРНЫХ И СПОРТИВНЫХ МЕРОПРИЯТИЙ

Токарев В.И. – студент, Крайванова В.А. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Разработка онлайн-приложений – активно развивающаяся отрасль, один из локомотивов ИТ-индустрии. В этой работе мы расскажем об одной такой разработке, о проблемах, с которыми мы столкнулись в процессе реализации, и о путях решения этих проблем. Проект представляет из себя сервис по продаже билетов на мероприятия в виде web-приложения. После регистрации пользователь может покупать билеты на мероприятия и тут же оплачивать их с банковской карточки. После покупки он получает персональный электронный билет. Приложение разработано на PHP Web фреймворке Yii2. Данный фреймворк является одним из лучших MVC фреймворков, так как в полной мере реализует этот паттерн программирования. В качестве базы данных была выбрана реляционная СУБД MySQL. Собирается проект при помощи установщика зависимостей Composer. Система сохранения версий Git, репозиторий для хранения данных BitBucket. Макет сверстан на каркасе Bootstrap 3.В качестве сторонних API использовались API смс шлюза, для отправки СМС сообщений, а также API Яндекс.Карты и Яндекс.Деньги

В процессе разработки приложения было обнаружено несколько проблем, часть их них были связаны с бизнес логикой, а часть – со спецификой используемой технической платформы.

Одновременная покупка одного билета двумя клиентами

Если два пользователя будут одновременно покупать один и тот же билет, то возникнет неопределенная ситуация. Проблема заключается в том, что может быть продано больше билетов, чем было запланировано. Если же билеты имеют свой уникальный номер, то либо один из пользователей окажется без билета, либо у двух пользователей окажется один и тот же билет. Решением данной проблемы является внедрение системы бронирования. То есть, сначала пользователь выбирает билет, затем система проверят возможность покупки этого билета, в режиме транзакции блокирует билет на определенное время и при отсутствии ошибок перенаправляет на сайт оплаты. Бронирование ограничено по времени.

Попытка купить уже купленный билет

Данная проблема вытекает из предыдущей проблемы и способа её решения. Становится возможным «подвисание» денег в системе, то есть пользователь перевел сервису деньги, но

не получил оплаченной им услуги.Опишем, как может произойти это самое «подвисание». Пользователь забронировал билет, перешел на страницу оплаты, но по тем или иным причинам не успел выкупить его во время бронирования (допустим, 1 час). После того, как билет стал доступным для покупки, его купил другой пользователь системы. Через 3 часа первый пользователь решил закончить процесс покупки и оплатил билет, но ни покупатель, ни Яндекс не в курсе, что билет уже был куплен, поэтому деньги отправляются на счет сервиса, но закрепить билет за первым пользователем невозможно, так как его уже купили.Проблема обостряется тем, что мы не можем произвести возврат, так как Яндекс не сообщает номер карточки плательщика и не позволяет вернуть деньги отправителю в автоматическом режиме. Для решения таких ситуаций необходимо фиксировать факт неудачного платежа, затем оповещать пользователя(в сервисе есть его телефон и e-mail) об ошибке покупки. После чего такие проблемы вручную решаются менеджером сервиса и пользователем.

Достоверность суммы, перечисленной через Yandex.Money

Проблема с одной стороны неоднозначная, так как Яндекс «подписывает» все транзакции при помощи специальных токенов, один из которых хранится на сервере, и при проверке входящего платежа, данные которые там пришли – достоверные. Но есть неучтенная опасность. Данные для оплаты пересылаются в открытом виде обычным POST запросом, поэтому перехватить и подменить сумму «к оплате» вполне реально, так как она никак не шифруется и не подписывается.Решениемявляетсяпроверка при получении платежа, соответствует ли он ожидаемой сумме, и если не соответствует, то записать платеж в отдельную таблицу и решать проблему вручную.

SQL запросы: медленный запрос

Запрос на вывод данных для виджета GridView брал основную таблицу, в которой было чуть больше 2000 записей, прицеплял к ней еще две на 1000 записей и 30 записей соответственно, а затем производил выборку по столбцам, у которых были установлены индексы, и сортировку. Запрос занимал 1317 мс. После оптимизации удалось добиться выполнения запроса за 117мс,то есть в 11 раз быстрее. Если же запрос повторить, то его результат приходит из кэша за 2.9мс, что уже в 450 раз быстрее. Так как первый запрос в принципе не попадал в кэш и всегда выполнялся от 1100 до 1700 мс, то это тоже достижение.

Таблица 1

	Запрос	**Время выполнения (мс)**
До	SELECT `list`.`record_id` AS `event_id`, `list`.`moderation_id` AS `id`, `list`.`name` AS `name`, `list`.`description` AS `description`, `list`.`afisha` AS `afisha`, `list`.`datetime` AS `datetime`, `list`.`moderation` AS `moderation`, `events`.`status` AS `status`, `list`.`moderation_date_off` AS `date_off`, `list`.`moderation_date_on` AS `date_on`, `category_events`.`name` AS `category` FROM `events` LEFT JOIN (SELECT * FROM `moderation_list` WHERE `moderation_id` IN (SELECT MAX(`moderation_id`) FROM `moderation_list` WHERE `section_id`=4 GROUP BY `record_id`)) AS list ON events.event_id=list.record_id LEFT JOIN `category_events` ON events.category_id=category_events.category_id WHERE (`events`.`user_id`=3) AND (NOT (`events`.`status`=3)) GROUP BY `list`.`record_id` ORDER BY `list`.`record_id` LIMIT 10	1317
После	SELECT STRAIGHT_JOIN `list`.`record_id` AS `event_id`, `list`.`moderation_id` AS `id`, `list`.`name` AS `name`, `list`.`description` AS `description`, `list`.`afisha` AS `afisha`, `list`.`datetime` AS `datetime`, `list`.`moderation` AS `moderation`,	117

	`events`.`status` AS `status`, `list`.`moderation_date_off` AS `date_off`, `list`.`moderation_date_on` AS `date_on`, `category_events`.`name` AS `category` FROM `moderation_list` `list`, `events`, `category_events` WHERE (((events.category_id = category_events.category_id) AND (list.record_id = events.event_id)) AND ((`events`.`user_id`='3') AND (`list`.`section_id`=4))) AND (NOT (`events`.`status`=3)) ORDER BY `list`.`record_id` DESC LIMIT 10	
После(кэш)	--//--	2.9

Дело в том, что когда одном запросе идет работа с несколькими таблицами, то MySQL объединяет таблицы так, как он считает оптимальным [1]. Индексы при этом берутся только по первой таблице, и если первой окажется таблица, в которой отсутствует необходимое поле и соответствующий индекс, то запрос может отрабатывать очень долго. Что бы указать СУБД, в каком порядке обрабатывать таблицы, после команды SELECT необходимо использовать ключевое слово STRAIGHT_JOIN. Оно «рекомендательно» указывает MySQL соединять таблицы именно в том порядке, в котором их перечислили мы и использовать индексы соответствующей первой таблицы.

SQL запросы: сложный запрос

Стояла задача, на странице мероприятий для каждого мероприятия указывать минимальную стоимость. У одного билета может быть указано до трех цен, то есть надо сначала выбрать минимальную из этих трех, а потом и из остальных билетов. Решить данную задачу помогла функция COALESCE. Суть функции в том, что она возвращает первое значение, которое не равно NULL. Поэтому запрос был таков «COALESCE(`sell_price`, `ruble`, `energy`) AS `cost`». В ходе разработки выяснилось, что невозможно составить такой SQL запрос, который адекватнов плане времени и производительности находил бы минимальную стоимость. Поэтому было принято решение изменить подход к задаче и хранить номер билета с минимальной ценой в записи мероприятия. В результате возникает 3 случая, когда номер этого билета надо обновить:

- добавили новый билет или удалили старый;
- продали последний билет какого-то типа;
- у билета закончился срок продажи.

В первом и втором случае при наступлении таковой ситуации надо найти новый билет. Но как следить за тем, что билет еще действителен? Так как при отображении билета, мы показываем сколько он еще будет продаваться, то почему бы перед выводим не проверить, просрочен оно или нет, и в случае если билет просрочен вызвать метод поиска нового билета.

Настройка кэширования

В Yii2 имеется широкий функционал для кэширования. Одним из них является кэширование тегами. Например, мы сделали некоторый запрос, сохраняем его в кэш и указываем 4 параметра: имя, данные для кэширования, срок кэширования, зависимость(от неё зависит досрочное удаление из кэша). В нашем случае, в качестве зависимости при запросе мы указываем тег, и если таковой уже имеется, то результат берется из кэша, иначе тег создается и кэш попадает в хранилище. Если же мы изменили данные, то мы можем удалить тег, тем самым удалив результат запроса их кэша.

Разработанное приложение проходит этап тестирования, в ближайшее время планируется ввод в эксплуатацию.

Список литературы

1. Оптимизация ORDER BY — о чем многие забывают [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/138163/
2. Yii PHP FrameworkVersion 2 [Электронный ресурс]. – Режим доступа: http://www.yiiframework.com/doc-2.0/index.html
3. TheDefinitiveGuidetoYii 2.0 [Электронный ресурс]. – Режим доступа: http://www.yiiframework.com/doc-2.0/guide-README.html
4. Documentation [Электронный ресурс]. – Режим доступа:http://php.net/docs.php

МЕТОДЫ АВТОМАТИЧЕСКОЙ КЛАССИФИКАЦИИ ДОКУМЕНТОВ

Самойлов С.С. – магистрант, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Стремительный рост сети интернет естественным образом породил проблемы поиска и упорядочивания информации. Сегодня в электронных хранилищах по всему миру содержатся терабайты информации. Информационных источников становится все больше и получить из этого океана нужные знания становится все труднее. Самостоятельно человек уже давно не в силах эффективно решать эту задачу. Задача классификации документов может быть рассмотрена как функция

$$\varphi: D \times C \rightarrow [0..1], \text{ где } D = \{d1, d2, \dots dk\} - \text{множество документов,}$$

$$C = \{c1, c2, \dots, cm\} - \text{множество возможных категорий.}$$

Классификатор для каждой категории должен быть обучен на заранее созданной обучающей выборке, в которой вручную подбираются соответствующие требуемой категории документы. Решение задачи автоматической классификации документов очень важно. Задача классификации состоит в разделении данных на классы. Исходные данные для классификации — это обучающая выборка, классы её элементов заранее известны. Цель классификации состоит в том, чтобы построить множество моделей, которые могут правильно установить класс различных объектов.

Категоризация при обработке тестовой информации требует применения алгоритмов машинного обучения и методов обработки естественного языка [1]. Некоторые системы машинного обучения пытаются устранить потребность в участии человека при анализе данных, в то время как другие принимают совместный подход между человеком и машиной. Конечно, человеческое участие не может быть полностью устранено, так как разработчику системы необходимо указать, каким образом данные должны быть представлены и какие механизмы будут использоваться для поиска характеристик данных. В данной работе представлен сравнительный анализ методов классификации документов

Дерево решений [2]. Каждый узел такого дерева это утверждение. В зависимости от правдивости этого утверждения относительно классифицируемого документа осуществляется переход к левому или правому потомку. В листовых узлах указан класс. Таким образом, пропустив документ от корня до листа можно узнать его класс. Однако перед использованием дерева решений его нужно построить. Обычно это нетривиальная задача.

Н*аивный байесовский классификатор* [3]. Он строится на предположении условной независимости. Это означает, что документ представляется как набор слов, вероятности

появления которых условно не зависят друг от друга. В соответствии с теоремой Байеса вероятность того, что документ принадлежит к какому-либо классу, рассчитывается по формуле 1.

$$c_{map} = \arg\max_{c \in C} \left[\log \frac{D_c}{D} + \sum_{i=1}^{n} \log \frac{W_{ic} + 1}{|V| + \sum_{i' \in V} W_{i'c}} \right] \quad (1)$$

- c_{map} – вероятность того, что документ принадлежит конкретному классу;
- D_c – количество документов принадлежащих классу c;
- D – общее количество документов в обучающей выборке;
- n – количество слов в документе;
- W_{ic} – количество раз сколько i-ое слово встречается в документах класса c;
- V – количество уникальных слов.

Метод k ближайших соседей. Для классификации документа его нужно преобразовать в вектор. Итоговый класс выбирается в зависимости от классов документов соответствующих k ближайшим векторам из обучающей выборки. Возникает проблема выбора метрики при определении функции расстояния между соседями, что при текстовой классификации существенно усложняется несбалансированностью количества объектов, принадлежащих классу и не принадлежащих ему.

Для классификации документов можно использовать искусственную нейронную сеть. Для этого нужно также преобразовать документ в вектор. При этом значение вектора пойдёт на вход нейронной сети, выходные нейроны будут соответствовать классам документа.

Метод опорных векторов может [4]. Если представить документы как точки в n-мерном пространстве, то метод опорных векторов предоставляет возможность построить гиперплоскость, разделяющую документы одного класса от документов другого. Для классификации достаточно рассчитать, с какой стороны плоскости окажется входной документ.

Один из простейших *методов векторизации документа - bag-of-words*. При использовании этого метода вектор строится на основе количества совпадений каждого слова в документе. Так как слов может встретиться очень много, размерность вектора может быть очень велика. Существуют различные улучшения метода для уменьшения размерности итогового вектора. При использовании этого метода для векторизации не учитывается семантический контекст и порядок слов.

Существуют более продвинутые методы векторизации документов, которые используют семантический контекст [5]. Процент ошибок при таких подходах значительно уменьшается.

Список литературы

1. Khan A., Baharudin B., Lee L.H., Khan K. A Review of Machine Learning Algorithms for Text-Documents Classification. // Journal of Advances in Information Technology, 2010, vol. 1, no. 1, pp. 4–20.
2. Деревья решений — общие принципы работы [Электронный ресурс]. Режим доступа: https://basegroup.ru/community/articles/description, свободный.
3. Наивный байесовский классификатор [Электронный ресурс]. Режим доступа: http://bazhenov.me/blog/2012/06/11/naive-bayes, свободный.
4. Метод опорных векторов [Электронный ресурс]. Режим доступа: http://webground.su/services.php?param=book&part=chapter+4_7.htm, свободный.
5. Le Q.V., Mikolov T.: Distributed representations of sentences and documents. In: Proceedings of the 31th International Conference on Machine Learning (ICML), 2014, pp. 1188–1196.

ПРИМЕНЕНИЕ СЕТИ РЕФЕРЕНЦНЫХ БАЗОВЫХ СТАНЦИЙ SMARTNET ДЛЯ ОПРЕДЕЛЕНИЯ ТЕКУЩИХ КООРДИНАТ КОЛЕСНОЙ СЕЛЬСКОХОЗЯЙСТВЕННОЙ МАШИНЫ

Ненайденко А.С. – аспирант, Поддубный В.И. – д.т.н., профессор
Алтайский государственный технический университет (г. Барнаул)

С каждым годом увеличивается сфера применение глобальных спутниковых навигационных систем (GPS, ГЛОНАСС, Galileo). Одним из приоритетных направлений их использования продолжает оставаться разработка систем точного земледелия. Применение этих систем в сельском хозяйстве позволяет обеспечивать с высокой точностью рабочее движение сельскохозяйственных машин по задаваемой траектории без участия водителя и/или с его частичным участием. Практика использования точного земледелия показала экономическую эффективность и целесообразность внедрения данной технологии. Однако, высокая стоимость импортного оборудования, а также плохая применимость к отечественному сельскохозяйственному парку затрудняют повсеместное внедрение данной технологии в России [1].

Одной из главных проблем использования более дешевых навигационных приемников является точность определения координат. Для нормального функционирования системы точного земледелия, которая должна работать в режиме Real Time Kinematic (RTK), необходима дециметровая, а при выполнении некоторых полевых работ и сантиметровая точность. Недорогие датчики, в том числе и отечественного производства, выдают точность порядка нескольких метров, что является недопустимым для задач точного земледелия. Разумеется, есть множество приемников, которые позволяют определять местоположение с высокой точность, но их стоимость исчисляется сотнями тысяч рублей (Trimble, Leica, Javad и другие производители). Такие датчики позволяют обеспечить точность до нескольких сантиметров.

Одним из вариантов решения проблемы точности недорогих навигационных приемников является использование RTK поправок от специальных сетей референцных базовых станций. Такая сеть представляет собой множество равномерно распределенных станций, каждая из которых построена с использованием высокоточного дорогостоящего навигационного оборудования. Каждая входящая в сеть базовая станция генерирует с заданной частотой поправочную информацию по фазовым измерениям. Эта информация передается посредством интернет соединения на NTRIP (Networked Transport of RTCM via Internet Protocol) сервера. В совокупности эти сервера программно объединяются в NTRIP Caster, поправочную информацию которого уже могут использовать с помощью клиентского ПО. Общая концепция взаимодействия NTRIP компонентов представлена на рисунке 1. Поправки передаются в формате RTCM SC 104-3.x.

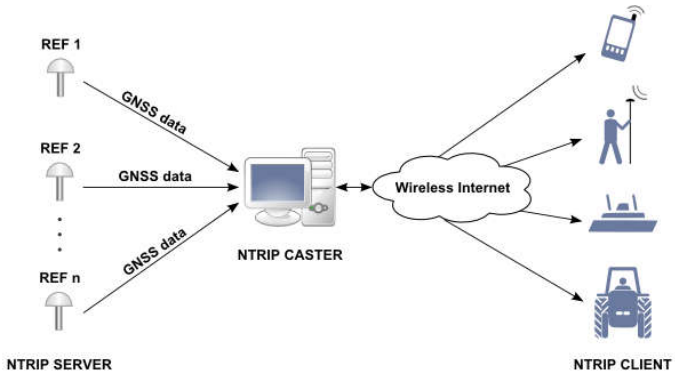


Рисунок 1 – Схема взаимодействия NTRIP компонентов

Самой крупной сетью референцных базовых станций в России является сеть SmartNet компании Навгеоком [2]. Она состоит на текущий момент из более чем 200 станций, расположенных по всей стране. На территории Алтайского края расположены 2 станции: в Барнауле и Тальменке. Стоит заметить, что услуги предоставления RTK поправок имеют не малую стоимость, но это гораздо меньше затрат необходимых для приобретения аналогичного высокоточного оборудования. При поддержке администрации Алтайского края, нами был получен безвозмездный тестовый доступ к сети SmartNet на 1 год для проведения всех необходимых экспериментов.

Для определения точных координат колесной машины было реализовано программное обеспечение на языке программирования C#. Схематично взаимодействие его составляющих, а также форматы обмена представлены на рисунке 2.

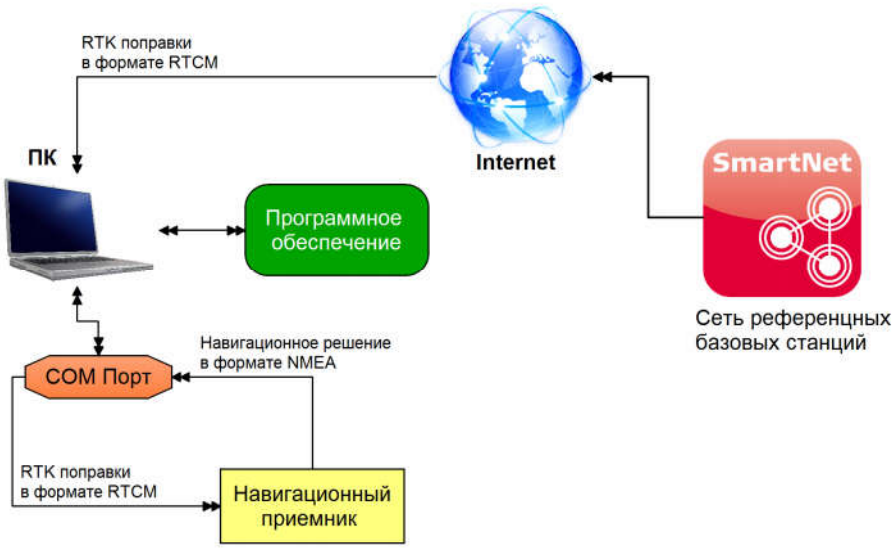


Рисунок 2 – Схема обмена информацией между компонентами системы

Программное обеспечение позволяет наладить обмен посредством сети интернет между сервером RTK поправок сети SmartNet и навигационным приемником, выступающим в роли ровера, а на выходе получить координаты текущего местоположения. Все функционирование происходит в режиме реального времени. Обмен между каждым компонентом системы выполняется в отдельном вычислительном потоке. Внешний вид основного окна программного обеспечения представлен на рисунке 3.

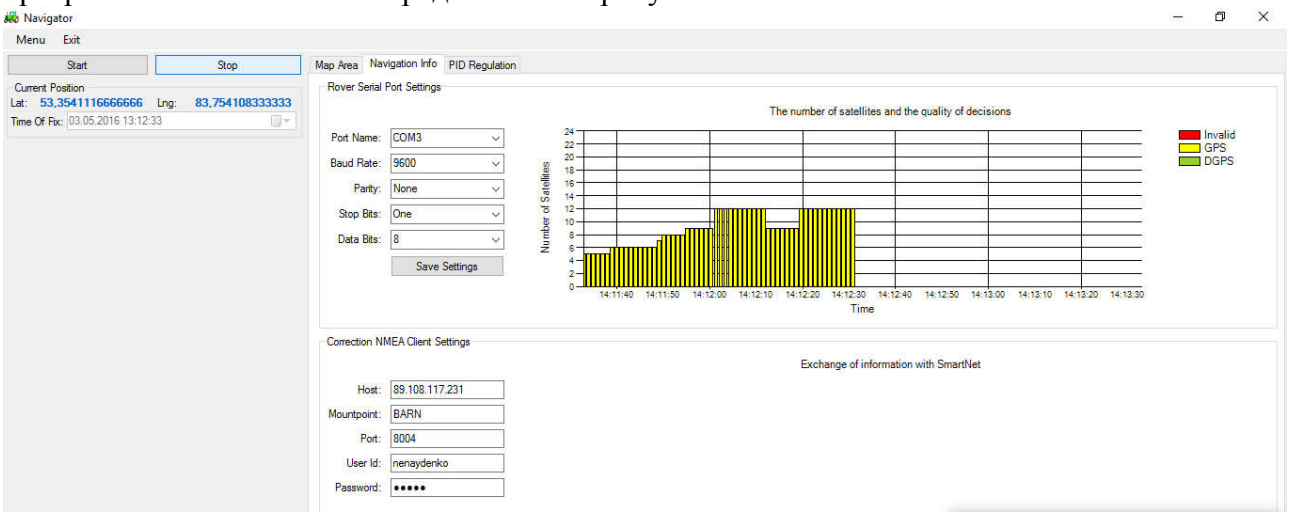


Рисунок 3 – Вид основного окна программного обеспечения

Полученный программный продукт является частью системы управления, которую планируется применять для обеспечения движения колесной машины по заданной траектории. Для получения RTK поправок от NTRIP сервера сети SmartNet реализован формат обмена RTCM, для считывания текущего решения с навигационного приемника - формат NMEA. Так как реализуемые форматы являются общепринятыми, то они применимы для любого двухчастотного RTK - приемника.

Список литературы

1. Зыков Р.В., Поддубный В.И. Разработка программного обеспечения для системы определения точных координат движущегося объекта с применением спутниковых радионавигационных систем // Горизонты образования. Приложение: Научно-техн. конф. "Наука и Молодежь-2014". Секция "Информационные технологии". Подсекция "Программное обеспечение вычислительной техники и автоматизированных систем", 24 Апреля 2014, С. 39-42. - Режим доступа: http://edu.secna.ru/media/f/povt2014.pdf
2. SmartNet | Принципы работы [электронный ресурс]. URL: http://smartnet-ru.com/principy-raboty_155.htm (дата обращения 20.04.2016).

ПРОЕКТИРОВАНИЕ СИСТЕМЫ АНАЛИЗА СТРУКТУРЫ МАТЕРИАЛОВ, НА ОСНОВЕ ДАННЫХ С АСМ-МИКРОСКОПА

Головин М.А. – магистрант, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Компьютерный анализ и моделирование играют большую роль при проведении научных исследований. Исключением не является и область материаловедения. Цифровые измерительные приборы позволяют изучать структуру материалов вплоть до молекулярного уровня, быстро получать и обрабатывать данные. Их обработка на вычислительной машине требует применения специализированных информационных систем. Такие системы должны предоставлять пользователю удобный графический интерфейс, иметь средства ввода информации в реальном формате, полученном от физических устройств, используемых в исследовательской аппаратуре. С точки зрения архитектуры система должна быть модифицируемой и расширяемой – исследователю могут потребоваться дополнительные методы анализа или новые модификации существующих алгоритмов. Очень часто такие алгоритмы обладают высокой вычислительной сложностью. Поэтому особый интерес представляют реализации вычислительных алгоритмов, предназначенных для быстрой обработки больших объемов данных, а также разработка методов эффективного уменьшения порядков вычислительной сложности расчетов, реализуемых в системе.

В настоящее время технологии формирования нано-структур широко используются в различных прикладных задачах. Для изучения топологии таких структур и исследования их эволюции эффективно используется аппарат фрактальной геометрии. Вопрос о том, с какими физическими характеристиками и механизмами связаны фрактальные характеристики поверхности, не вполне очевиден, однако фрактальная размерность материала должна определять некоторые его свойства. В связи с этим важной задачей является определение фрактальной размерности поверхностей, полученных различными методами и возможность прогнозирования их свойств на основе фрактального анализа. Задачей данной работы является проектирование системы анализа структуры материала на основе данных о его поверхности. Одной из важных задач, решаемых системой, является определение фрактальной размерности поверхности.

Система спроектирована на основе шаблона проектирования MVC – «модель, представление, контроллер».

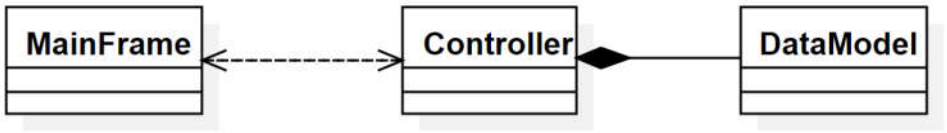


Рисунок 1 – Основные элементы системы

Представление используется для вывода данных и взаимодействия с пользователем. Модель представляет собой абстракцию данных, методов их обработки и визуализации. Контроллер реализует логику работы системы, используется для связи представления и модели. Такой подход позволяет разделить логику работы и подробности представления и обработки данных.

Модель данных включает в себя следующие основные элементы: парсер (анализатор входных данных), собственно модель поверхности, трехмерный визуализатор для вывода поверхности, визуализатор графиков для вывода результатов расчетов.

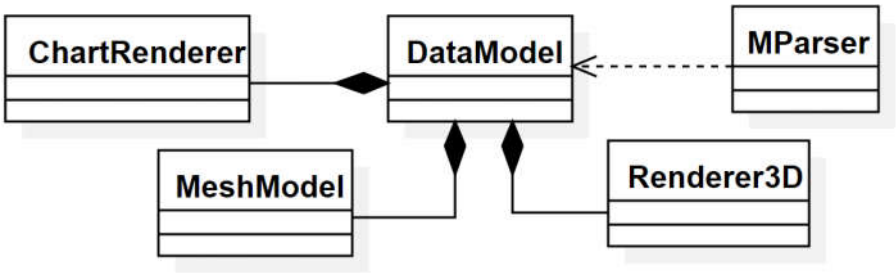


Рисунок 2 – Основные элементы модели данных

Процесс функционирования системы представлен на следующей UML-диаграмме последовательности (Рисунок 3).

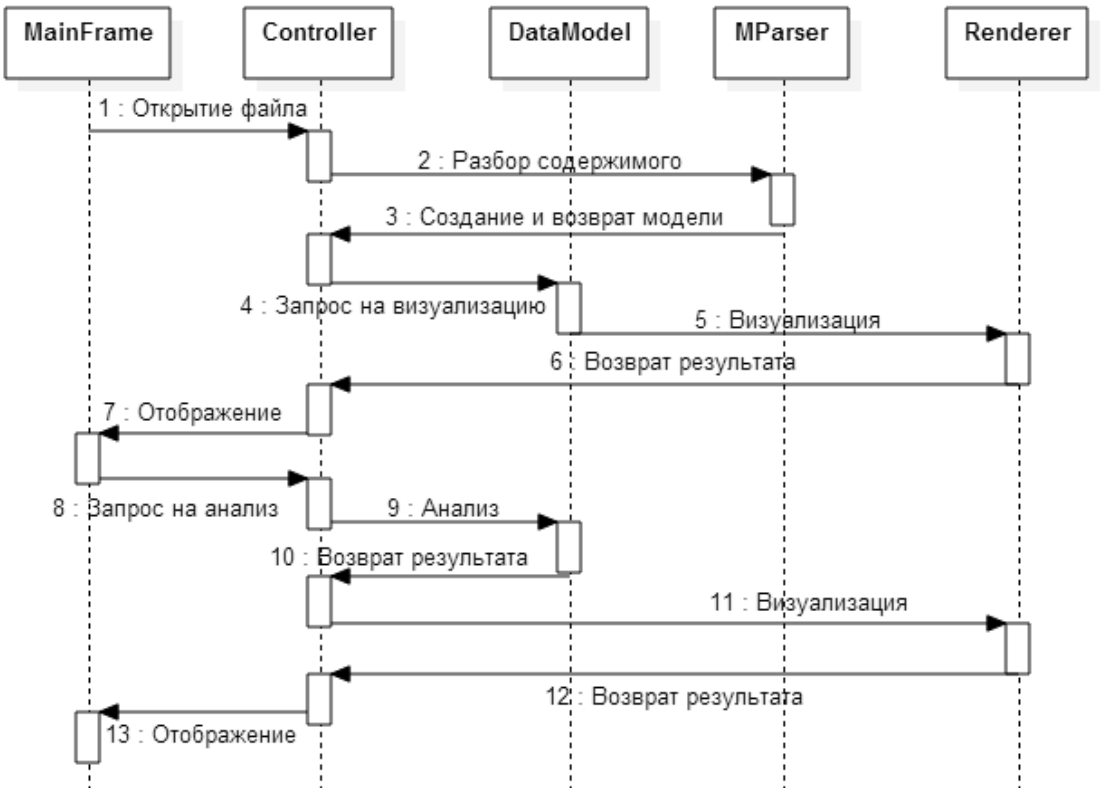


Рисунок 3 – UML-диаграмма последовательности

Обработка полученного с микроскопа файла начинается с передачи имени файла от представления к контроллеру. Контроллер инициирует разбор файла анализатором. Анализатор строит на основе содержимого файла модель данных и возвращает её контроллеру. Далее контроллер осуществляет запрос к модели на визуализацию поверхности. Модель задействует соответствующий модуль визуализации и возвращает полученное изображение контроллеру. Контроллер передает изображение представлению для отображения.

Когда пользователь хочет провести фрактальный анализ, представление отправляет контроллеру запрос. Контроллер делает соответствующие запросы на анализ и визуализацию и передает результат представлению для отображения.

С целью сокращения зависимостей спроектирована собственная система визуализации трехмерных изображений, не использующая внешних библиотек. Для этого разработаны модули для работы с трехмерными однородными координатами и аффинными преобразованиями в матричной форме. Они же используются и при построении графиков.

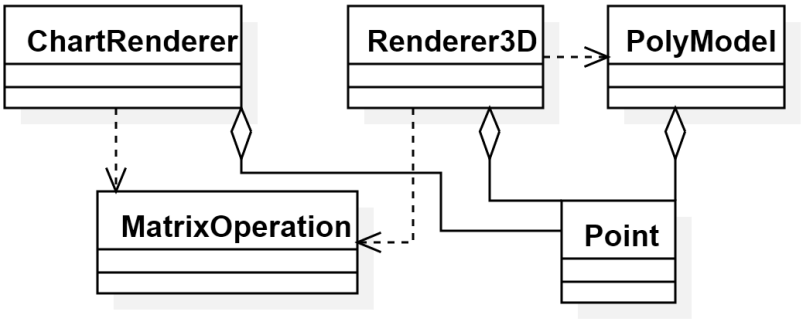


Рисунок 4 – Схема подсистемы визуализации

Для визуализации трехмерной поверхности используется её полигональное представление.

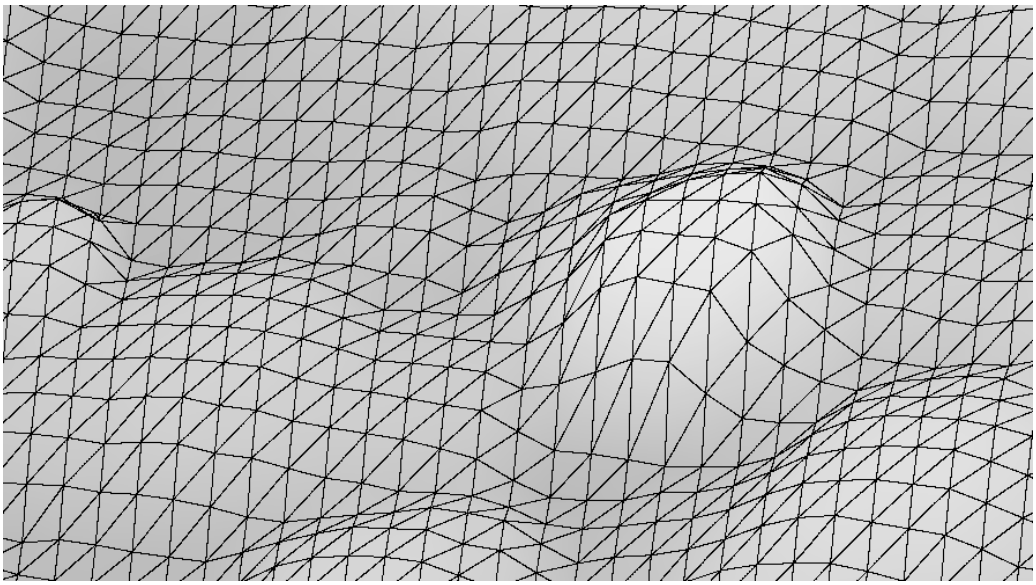


Рисунок 5 – Полигональное представление поверхности

При проектировании подсистемы работы с поверхностью был использован шаблон «стратегия» для интерполяции точек поверхности и определения площади, что повышает вариативность применения алгоритмов обработки и способствует расширяемости системы. Реализуемая в данный момент стратегия вычисления площади поверхности основана на методе триангуляции.

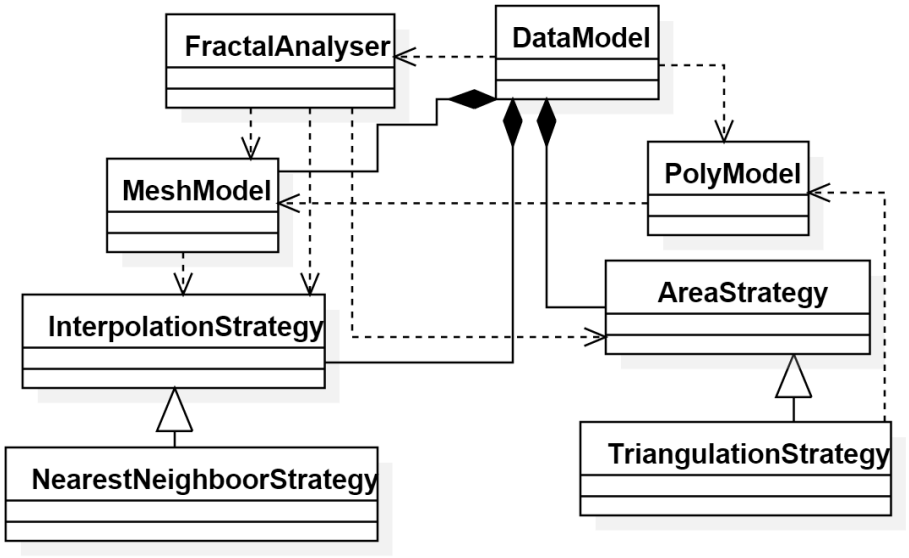


Рисунок 6 – Схема подсистемы работы с поверхностью

Анализ структуры был выделен в отдельный модуль, который использует внешние по отношению к себе стратегии.

Система была реализована с использованием языка Java, что обеспечивает ее кроссплатформенность. Реализованная в рамках системы подсистема визуализации не требует специальных аппаратных и программных интерфейсов и без проблем работает на большинстве компьютеров.

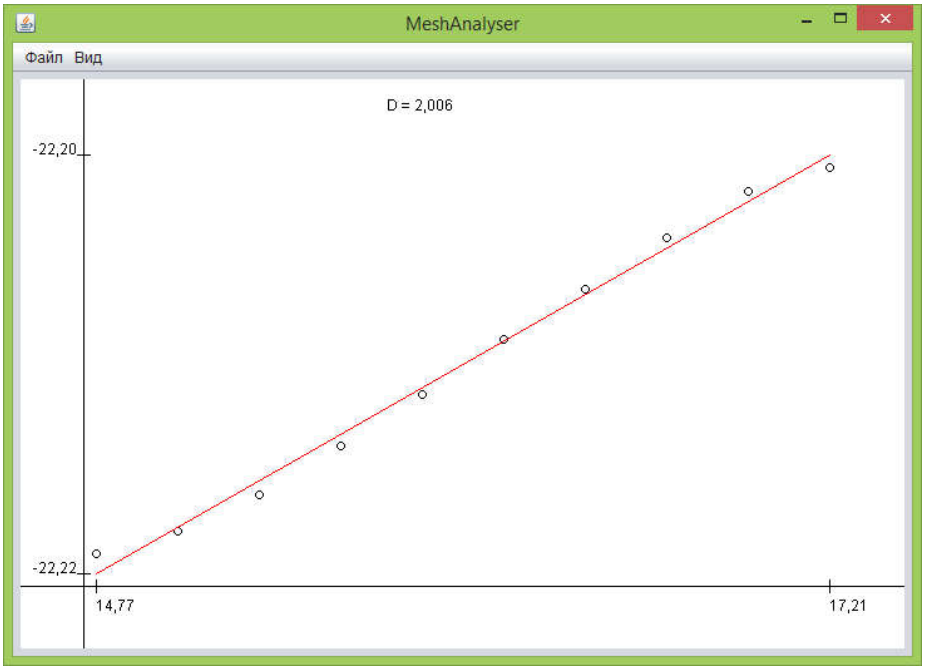


Рисунок 7 – Результаты фрактального анализа

Архитектура системы разрабатывалась с учетом дальнейшего расширения и усовершенствования, дополнения другими методами анализа. В качестве дальнейшего развития системы планируется расширение интерфейса, в частности возможностью управлять параметрами анализа и визуализации. Применение шаблонов при проектировании повышает расширяемость системы, а также делает код более организованным и понятным для программиста.

Список литературы

1. Мандельброт Б. Фрактальная геометрия природы [Текст] / Б. Мандельброт. – М.: ИКИ, 2002. – 654 с.
2. Федер Е. Фракталы [Текст] / Е. Федер. – М. : Мир, 1991. – 261 с.
3. Краткий курс компьютерной графики [Электронный ресурс] // Сайт Хабрахабр [Сайт]. – Режим доступа: http://habrahabr.ru/post/248153/
4. Фримен Э. Паттерны проектирования [Текст] / Э. Фримен, К. Сьерра, Б. Бейтс – СПб. : Питер, 2011. – 656 с.
5. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] / Э. Гамма [и др.] – СПб. : Питер, 2001. – 368 с.

ВОССТАНОВЛЕНИЕ 3D ГЕОМЕТРИИ

Борисов В.В. – магистрант, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Любое изображение является плоским по определению. Но, как и всякий объект или явление в реальности, изображение можно интерпретировать в терминах информации, и оно приобретёт третье измерение - смысловую глубину. Человек без всякого труда воспринимает эту глубину, с лёгкостью восстанавливая в своём воображении трёхмерный мир - реальный или вымышленный - фактически всего лишь из большого числа цветных точек, пятен, линий.

Об окончательной формализации этого процесса восстановления сегодня говорить ещё рано: существует лишь множество отдельных алгоритмов и методик, показывающих отличные результаты на одних изображениях, и в то же время посредственные - на других. Большую помощь в воссоздании полного алгоритма "преобразования" 2D > 3D могло бы оказать знание действительного механизма создания трёхмерного образа предмета по той информации, которую мозг человека получает через зрение, но этот механизм, как, впрочем, и принципы функционирования самого мозга, остаётся неизвестным.

В настоящее время разработано большое количество разнообразных подходов к решению задачи восстановления трехмерных сцен по их двумерным изображениям. Эти подходы можно подразделить на:

- Shape from Stereo, восстановление 3D по стерео паре.
- Shape from Motion - восстановление 3D из большого количества изображений этой сцены. Интерес к этим подходам связан с тем, что в отличие от стерео можно восстановить положение и ориентацию камер, внутренние параметры камер (фокусное расстояние), т.е. параметры, которые очень часто неизвестны, особенно при съемке фотоаппаратом с рук. соответствующие именно данной сцене!
- Shape from Manhattan - в случае когда известно, что сцена в основном формируется прямыми линиями и плоскостями, можно много информации извлечь из одного изображения.
- Shape from Shading - восстановление 3D по одному изображению, грубо говоря - на основе закона рассеяния Ламберта. Для сложных сцен не работает, на для изображений типа гипсовых статуй - по-видимому единственный способ (выше перечисленные не будут работать, так как не удастся найти характеристических особенностей).
- Shape from Focusing and Defocusing - если имеется ряд изображений, снятых с одной точки, но с различной фокусировкой, то в зависимости от удаленности, разные части сцены на изображении имеют различную степень резкости. По фокусировке - меняя

фокусировки находят какая часть в данный момент резкая, расстояние до этих частей равно расстоянию фокусировке.

- Shape from Texture - если известно, что сцена имеет поверхности с одинаковой текстурой (например оклеена обоями в клеточку), то анализируя изменение видимого размера элементов текстуры можно восстановить форму.
- Exotic: Shape from Scattering etc. Например, есть алгоритм определения расстояний по атмосферной дымке. Иногда такие методы могут быть полезны, но область применения их обычно существенно ограничена.

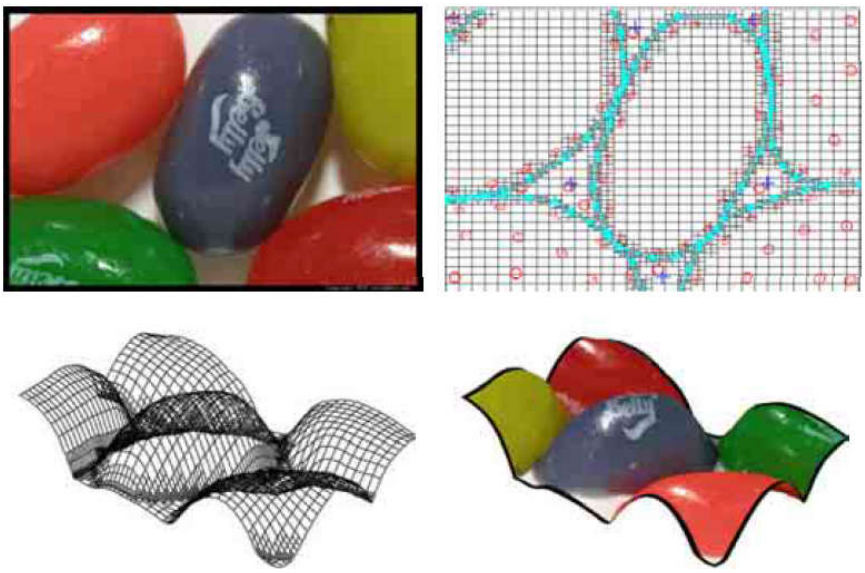


Рисунок 1 – Преобразование 2D изображения в 3D модель

Наиболее известные приложения для преобразования 2D изображений в 3D.

SketchUp – программа для моделирования относительно простых трёхмерных объектов – строений, мебели, интерьера.

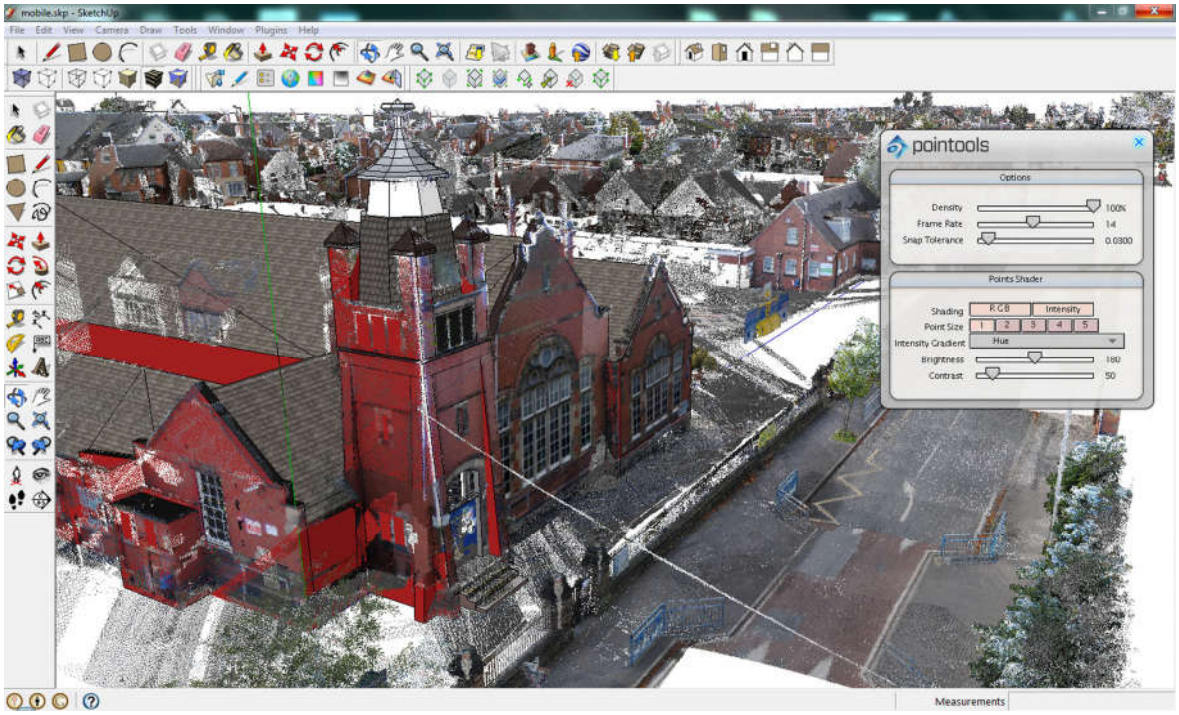


Рисунок 2 – Пример моделирования здания в SketchUp

Принадлежит компании Google. Разрабатывалась для грубого моделирования зданий и сооружений в Google Earth. Отлично показала себя при построении 3D моделей для

виртуальных экскурсий по достопримечательностям. Последовая версия вышла 5 февраля 2015 года, с тех пор не разрабатывается и не поддерживается.

RealViz Vtour (ныне Autodesk ImageModeler) — программный продукт разрабатываемый с 1998 года для преобразования изображений из 2D в 3D. Используется для медицинского моделирования тканей, скелетов и организмов; автоматизированного проектирования – создание 3D моделей зданий, сооружений и ландшафта основываясь на фотографиях местности; видеоигр; создание моделей для киноиндустрии (Гарри Поттер и кубок огня).

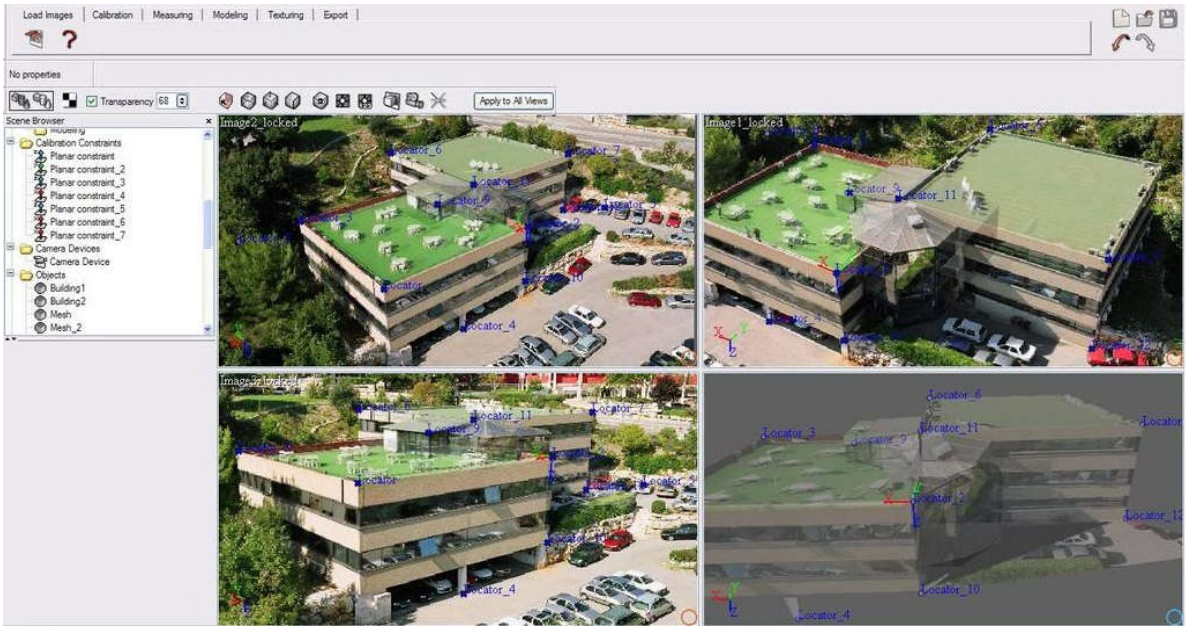


Рисунок 3 – Пример моделирования здания в RealViz Vtour

Разрабатываемая программа будет восстанавливать 3D изображение по стерео паре. Восстановление 3D по стерео паре заключается в том, что имеется два изображения, на каждом изображении производится поиск интересных точек, методом Харриса и поиск блобов (blobs) — одинаковых частей изображений на разных масштабах данных изображений.

После нахождения точек интереса вычисляются дескрипторы интересных точек, причём, изображения могут быть разным масштабом (инвариантность к масштабированию, используются бобы) и изображения могут быть повёрнуты на разные углы (инвариантность к повороту, используются блобы и Харрис). Далее, после нахождения дескрипторов на обоих изображениях они (дескрипторы) сопоставляются (производится поиск похожих частей на обоих изображениях).

Изображения деформируются (масштабирующий, сжимаются/растягиваются части изображения), чтобы предать им вид стерео пары. Вычисляется направление перспективы обоих изображений. В итоге, исходя из смещения объектов на изображениях вычисляется координата Z данных изображений.

Список литературы

1. Hartley R.I., Zisserman A. Multiple View Geometry. Cambridge, UK: Cambridge University Press, 2000.
2. Governi L., Furferi R., Palai M., Volpe Y. 3D geometry reconstruction from orthographic views: A method based on 3D image processing and data flitting. Computers in industry, 2013, № 64 pp.1290-1300.
3. Dimri J. Gurumoorthy B Handling sectional views in volume-based approach to automatically construct 3D solid from 2D views. Computer Aided Design. 2005. № 37. pp.485-495
4. Aldefeld B. On Automatic Recognition of 3D Structures from 2D Representations. Computer Aided Design, 1983, № 15 (2), pp.59-72.

ОСНОВНЫЕ ТРЕБОВАНИЯ К АРХИТЕКТУРЕ СИСТЕМЫ ОТСЛЕЖИВАНИЯ ПОЗЫ ЧЕЛОВЕКА НА ВИДЕОИЗОБРАЖЕНИИ

Есипенко С.П. – аспирант, Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Отслеживание позы человека необходимо в самых разных областях человеческой деятельности. Например, такая задача ставится при создании умных систем слежения, беспилотных летательных аппаратов и самоуправляемых автомобилей.

Некоторые подходы к решению данной задачи уже рассматривалась нами ранее [1, 2]. Однако там, как и во множестве других публикаций на тему отслеживания позы человека и захвата движения (например, [3,4,5,6]), основное внимание уделяется, непосредственно, решению задачи. Вопросы же, касающиеся архитектуры системы либо упоминаются вскользь, либо не упоминаются вовсе.

Конечно, разработка качественного решения задачи отслеживания позы человека может показаться гораздо важнее проектирования архитектуры системы. Однако, как показывает практика, без проработки хотя бы основных архитектурных вопросов, процесс разработки решения задачи отслеживания позы человека на видеоизображении значительно затрудняется. Действительно, система с хорошо продуманной архитектурой позволяет быстро менять алгоритмы отслеживания, проверять качество их работы в автоматическом режиме, проводить глубокий анализ сильных и слабых сторон конкретного решения. В системе с плохо продуманной архитектурой даже малейшее изменение в одном компоненте решения может повлечь за собой длинную цепочку изменений в других компонентах.

В рамках данной работы основное внимание будет уделено требованиям к архитектуре системы отслеживания позы человека на видеоизображении. Также будут перечислены основные компоненты системы.

Постановка задачи

В рамках исследовательской работы, целесообразно предъявить такие основные требования к системе, как гибкость, расширяемость и верифицируемость. Также можно упомянуть несколько менее важных требований, а именно – наглядность, компактность и производительность. Таким образом, задача заключается в разработке такой архитектуры, которая бы в наибольшей степени удовлетворяла перечисленным выше требованиям.

Рассмотрим требования подробнее. Под гибкостью системы подразумевается то, на сколько ее просто изменять. Обычно, гибкость системы напрямую зависит от степени связности ее компонентов: чем слабее связи между отдельными компонентами, тем проще вносить изменения в систему. С точки зрения задачи отслеживания позы, необходимо, чтобы можно было легко менять формат входных и выходных данных, алгоритмы и их отдельные стадии, модели тех или иных объектов и процессов. Особенно важно иметь возможность переключаться между различными моделями одних и тех же объектов. Например, то же тело человека можно моделировать просто набором отрезков в пространстве, цилиндрами и даже полноценными трехмерными сетками. Существует множество динамических моделей системы, моделей поверхности, моделей камеры, освещения и пр. Было бы удобно быстро настраивать систему на работу с теми или иными моделями.

Расширяемость тесно связана с гибкостью, но, в связи с ее чрезвычайной важностью для задачи отслеживания позы человека, целесообразно вынести ее как отдельное требование. Расширяемость системы – способность в кратчайшие сроки и с минимальными затратами дополнить систему новыми компонентами и функциями. В рамках задачи отслеживания постоянно требуется добавлять новые, более точные модели объектов и процессов, расширять старые алгоритмы новыми эвристиками, или же добавлять совершенно новые алгоритмы. При этом, хотелось бы максимально использовать уже имеющиеся наработки.

Чтобы сделать систему гибкой и расширяемой, необходимо провести тщательный анализ предметной области, особое внимание уделить процессу декомпозиции, то есть выделению ключевых компонентов, объектов и процессов, а также стадий отслеживания позы. Затем необходимо сфокусироваться на связях между компонентами, проработать интерфейсы взаимодействия между различными частями системы. Более подробно эти вопросы будут рассмотрены в последующих работах.

Под верифицируемостью понимается возможность проверки системы и анализа качества ее работы. При всей краткости данной формулировки, качественная проверка работы системы отслеживания позы человека является крайне нетривиальной задачей. Основная проблема заключается в том, чтобы понять насколько же ошиблась система при отслеживании позы того или иного субъекта? Откуда взять истинное расположение костей человека на видеоизображении? Какие метрики использовать для вычисления погрешности решения? Как проверить устойчивость отслеживания к различным факторам (смена освещения, тени, перекрытие объектов сцены и пр.)? Причем, хотелось бы, чтобы качество работы системы проверялось в автоматическом режиме, с предоставлением наглядного отчета для последующего анализа человеком.

Наглядность системы связана с тем, на сколько просто понять работу всей системы или ее частей для человека. Это скорее субъективное качество системы, которое сложно сформулировать формально, но, все же, ему требуется уделить некоторое внимание. Наличие данного качества могло бы значительно упростить работу нескольких исследователей над развитием системы отслеживания позы человека.

Компактность подразумевает, что архитектура системы не перегружена лишними компонентами и абстракциями. Это тоже несколько субъективное требование. По сути, оно определяет компромисс между гибкостью-расширяемостью системы с одной стороны и наглядностью – с другой. Предметную область можно дробить на компоненты практически неограниченно. Скелет можно сделать анатомически точным и смоделировать каждую косточку, причем со всеми законами физики, а в том же теле можно еще выделить такие компоненты как мускулатура, кожа, волосы и пр. Такой подход не только резко ударит по производительности, но и сделает систему чересчур сложной, перегруженной лишними сущностями, которые может и влияют на качество решения, но, скорее всего, незначительно.

Наконец, производительность показывает на сколько быстро система работает. Задача отслеживания позы человека является крайне ресурсоемкой. Поэтому, на этапе проектирования системы необходимо продумать вопросы повышения производительности, в частности, возможность распараллеливания вычислений и возможность выполнении вычислений на графических сопроцессорах.

Предлагаемое решение

В данном разделе будут описаны лишь основные компоненты системы отслеживания позы человека на видеоизображении, которые должны присутствовать в той или иной степени в любой подобной системе. Наглядная схема системы приведена на рисунке 1.

Основная система состоит из подсистемы получения входных данный, непосредственно алгоритма отслеживания и подсистемы выдачи выходных данных. Алгоритм может конфигурироваться через подсистемы настройки, позволяющей задавать параметры алгоритма и подсистемы эвристик, которая позволяет настроить, какие эвристики доступны алгоритму отслеживания. Основная система непосредственно решает задачу.

Верификация системы происходит за счет внешних компонентов. Проверочный набор содержит данные (наборы кадров) с правильным ответом (положения костей на каждом кадре). Данные подаются на вход системе, полученные ответы сверяются с правильным ответом с помощью верификатора, который создает отчет. Исследователь анализирует отчет и корректирует настройки, эвристики или, возможно, весь алгоритм отслеживания.

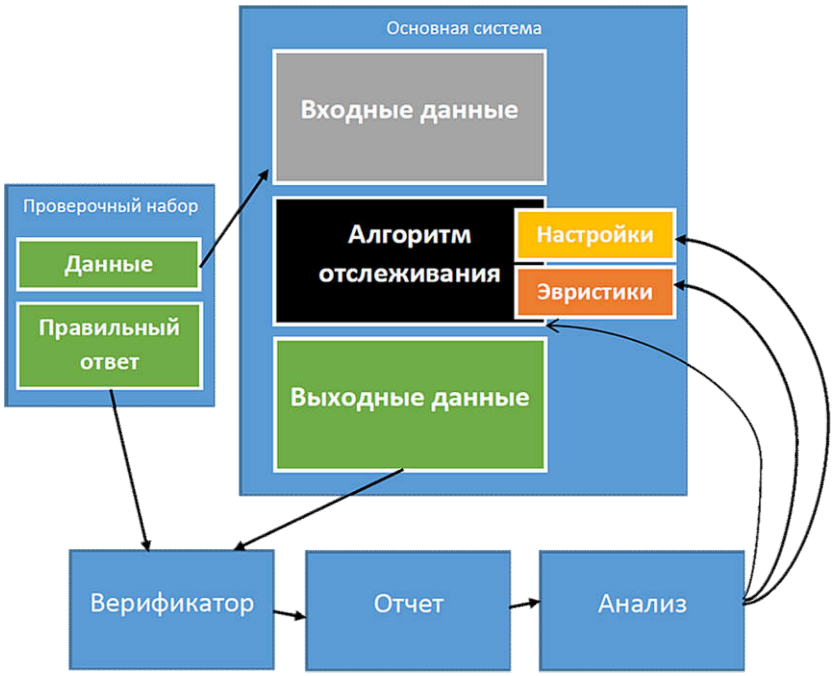


Рисунок 1 – Основные компоненты системы отслеживания позы человека

Конечно, это всего лишь общий набросок архитектуры системы. Подробное описание гибкой архитектуры проектируемой системы выходит за рамки данной работы, и будет дано нами в последующих работах.

Список литературы

1. Есипенко С.П., Крючкова Е.Н. Проблема скелетной реконструкции произвольных биологических и механических объектов в системах компьютерного зрения / С. П. Есипенко, Е.Н. Крючкова // XI Всероссийская научно-техническая конференция студентов, аспирантов и молодых ученых "Наука и молодежь – 2014". Секция «Информационные технологии». Подсекция «Программное обеспечение вычислительной техники и автоматизированных систем». / Алт. гос. техн. ун-т им. И.И. Ползунова. – Барнаул: изд-во АлтГТУ, 2014. – С. 11-13.
2. Есипенко С.П., Крючкова Е.Н. Применение алгоритма фильтрации частиц в задаче отслеживания позы человека / С. П. Есипенко, Е.Н. Крючкова // XII Всероссийская научно-техническая конференция студентов, аспирантов и молодых ученых "Наука и молодежь – 2015". Секция «Информационные технологии». Подсекция «Программное инженерия». / Алт. гос. техн. ун-т им. И.И. Ползунова. – Барнаул: изд-во АлтГТУ, 2015. – С. 37-40.
3. Tian J., Li L., Liu W. Monocular Human Motion Tracking with Non-Connected Body Part Dependency / J. Tian, L. Li, W. Liu // Digital Image Computing: Techniques and Applications. – IEEE, 2015. – С. 1–7.
4. Wandt B. 3D human motion capture from monocular image sequences / B. Wandt, H. Ackermann, B. Rosenhahn // IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). – IEEE, 2015. – С. 1–8.
5. Pfister T. Flowing ConvNets for Human Pose Estimation in Videos/ T. Pfister, J. Charles, A. Zisserman // Proceedings of the IEEE International Conference on Computer Vision. – IEEE, 2015. – С. 1913–1921.

6. Carreira J. Human Pose Estimation with Iterative Error Feedback [Электронный ресурс] / J. Carreira [и др.]. / Computing Research Repository. – Режим доступа: http://arxiv.org/abs/1507.06550. (Дата обращения: 16.04.2016 г.)

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ АНАЛИЗА И ВИЗУАЛИЗАЦИИ ЗАПИСЕЙ ОПЕРАЦИОННОГО ЖУРНАЛА АБС RS-BANK ДЛЯ ООО КБ «АЛТАЙКАПИТАЛБАНК»

Дерешева Д.С. – студент, Троицкий В.С. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Описание терминологии

АБС – Автоматизированная Банковская Система – комплекс программного и технического обеспечения, направленный на автоматизацию банковской деятельности.

RS-Bank – автоматизированная банковская система, разработанная фирмой R-Style Softlab, в данный момент используется ООО КБ «Алтайкапиталбанк».

Операционный журнал банка – электронный журнал, отражающий данные о всех операциях, как пользователей, так и системы, происходящих в банковской системе.

Постановка задачи

В банковском деле, возможность быстро и эффективно разрешать конфликтные ситуации в повседневной внутренней работе банков является очень востребованной. Часть конфликтов разрешается автоматическими системами за считанные доли секунды, что позволяет клиентам ожидать быстрое разрешение конфликтов. Следуя ожиданиям клиентов, добавляется необходимость быстрого и экономного расследования более сложных конфликтов, которые разрешаются только вручную. Одна только компания Visa, Inc. проводит через себя сотни миллионов транзакций в сутки [4], и эта цифра включает в себя только пользователей, соответствующих банковским картам. Внутренняя работа банка добавляет к этой цифре существенный прирост, таким образом делая расследование каждого конфликта человеческими ресурсами всё более дорогостоящим и времязатратным для банка.

В данной работе представлен визуальный подход к банковским данным, позволяющий сфокусироваться на происходящих событиях в системе в целом, нежели на конкретном пользователе, файле или типе операции. Благодаря такому подходу работники банка, занимающиеся разрешением конфликтов внутри системы, будут иметь полное представление о состоянии системы в заданный промежуток времени и смогут более быстро и эффективно отследить подозрительную активность, не затрачивая время и ресурсы на анализ каждого пользователя, файла и операции по отдельности. Так же снижается риск человеческого фактора, так как в текущих решениях сопоставлять данные в единое целое для разрешения конфликта приходится человеку.

Обзор известных подходов

Данная работа базируется на продукте RS-Bank от компании R-Style Softlab – ведущем российском разработчике и интеграторе банковского ПО [1]. Другие разработчики банковского ПО, например, Диасофт, IBM, Siebel Systems и Cognitive Technologies, так же занимаются продвижением своих продуктов с различной функциональностью [2]. Тем не менее, обсуждение стилистики и функциональности операционного журнала событий, на

базе которого проводится разрешение внутренних конфликтов не находится на повестке дня. Сами журналы выглядят весьма примитивно, сравнимо с таблицей в программе Microsoft Excel, однако предоставляют намного более ограниченные возможности визуализации данных, помимо простой фильтрации.

Собственное решение

Предлагаемое в этой работе решение фокусируется на приведении данных из операционного журнала событий банка к более простому для усвоения и чтения людьми виду, применяя техники визуализации и позволяя интуитивно перемещаться между пользователями, файлами и событиями, произошедшими в банковской системе в заданный промежуток времени. Такой визуальный подход не только позволяет быстро найти необходимую информацию, опираясь на естественное визуальное восприятие человека, но и объединяет множество строк данных журнала в единую систему, которую намного проще фильтровать, чтобы добраться до необходимой информации.

Краткое описание ПО

Программное обеспечение, решающее поставленную задачу, будет реализовано как десктопное приложение для Windows. Основной задачей данного ПО является упорядочивание и фильтрация данных операционного журнала. Фрагмент операционного журнала системы RS-Bank на предприятии ООО КБ «Алтайкапиталбанк» представлен на рисунке 1.

Фили	Дата	Время	Имя файла	Описание файла в словаре	Название операции
0	18.09.2014	20:16:01			Окончание процедуры
0	18.09.2014	20:16:03			Выход из программы
0	18.09.2014	21:24:09			Вход в программу
0	18.09.2014	21:25:01			Начало процедуры
0	18.09.2014	21:25:16	depparm.dbt	Параметры филиалов	Обновление записи
0	18.09.2014	21:25:16			Окончание процедуры
0	18.09.2014	21:25:16			Начало процедуры
0	18.09.2014	21:25:22	depparm.dbt	Параметры филиалов	Обновление записи
0	18.09.2014	21:25:22			Окончание процедуры
0	18.09.2014	21:25:24			Начало процедуры
0	18.09.2014	21:25:40	rtgroup.dbt	Файл групп пользователей	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	18.09.2014	21:25:40	groupmem.dbt	Состав группы	Транзакция ввода
0	27.02.2015	11:54:05	ofkntinf.dbt	Информация для извещения	Ввод записи
0	27.02.2015	11:54:05	document.dbt	Проведенные документы	Транзакция ввода
0	27.02.2015	11:54:05	document.dbt	Проведенные документы	Транзакция ввода
0	27.02.2015	11:54:05	dlinidoc.dbt	Связь сделок и сложных провод	Транзакция ввода
0	27.02.2015	11:54:05	dlinid.dbt	Бал.исх.документы сложн.прово	Транзакция ввода
0	27.02.2015	11:54:05	deal.dbt	Сделки	Транзакция изменения
0	27.02.2015	11:54:05	deal.dbt	Сделки	Транзакция ввода
0	27.02.2015	11:54:00			Начало процедуры
0	27.02.2015	11:53:58	lufvalue.dbt	Значеия дополнительных полей	Транзакция ввода
0	27.02.2015	11:53:58	lufvalue.dbt	Значеия дополнительных полей	Транзакция ввода
0	27.02.2015	11:53:58	ofkntinf.dbt	Информация для извещения	Ввод записи
0	27.02.2015	11:53:56	crd_op.dbt	Операции по кредиту	Транзакция изменения
0	27.02.2015	11:53:56	doc_acc.dbt	Документ по счету	Транзакция изменения
0	27.02.2015	11:53:56	copstage.dbt	операция по кредиту <-> стадия	Транзакция ввода
0	27.02.2015	11:53:56	doc_acc.dbt	Документ по счету	Транзакция изменения
0	27.02.2015	11:53:56	turn_acc.dbt	Оборот по счету	Транзакция ввода
0	27.02.2015	11:53:56	sb_acc.dbt	Лицевой счет	Транзакция изменения
0	27.02.2015	11:53:56	turn_acc.dbt	Оборот по счету	Транзакция ввода
0	27.02.2015	11:53:56	sb_acc.dbt	Лицевой счет	Транзакция изменения

Рисунок 1 – Фрагмент операционного журнала банка

Так как данные операционного журнала находятся во внутренней базе данных RS-Bank, к которой нет лицензионного доступа сторонним приложениям, необходимо обеспечить копирование данных в локальную базу, которую и будет использовать разрабатываемое ПО. Такое копирование будет производить администратор предположительно раз в сутки или по требованию. Схема копирования данных из базы RS-Bank в локальную представлена на рисунке 2. Визуализация данных предполагается в виде столбцевых графиков, отображающих фильтрацию данных по заданному промежутку времени, операциям, пользователям или группам пользователей, филиалам и используемым файлам. Так же будет предусмотрено масштабирование, для лучшего понимания представленных данных.

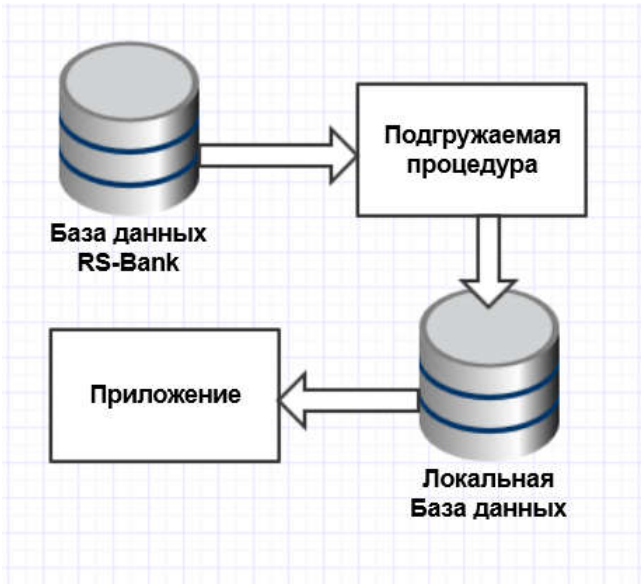


Рисунок 2 – Схема копирования данных

Заключение

Предлагаемая система поможет при разрешении внутрибанковских конфликтов, а также при составлении статистики о работе пользователей. В будущем, такая система позволит расширить применение журнала событий для других отделов банка, например, позволяя отделу по работе с персоналом определить ранние симптомы стресса у работников банка, что позволит сэкономить средства, потерянные из-за низкой продуктивности персонала [3].

Список литературы

1. R-Style Softlab R-Style Softlab | О компании [В Интернете] // R-Style Softlab | R-Style Softlab. - R-Style Softlab, 12 Апрель 2016 г.. - 12 Апрель 2016 г. - http://www.softlab.ru/company/.
2. Р. Г. Банковское ПО [Журнал]. - Москва : Открытые системы, 2005 г.. - Т. 02.
3. Сорокина Н.В. Искусство управления персоналом банка [Книга]. - [б.м.] : Бизнес-портал "Бизнес-Учебники.РФ", 1997.
4. Филипов Ю. Побит новый рекорд в 100 тысяч транзакций в день [В Интернете] // Coinside.ru - Bitcoin для простых смертных. - Coinside.ru, 4 Декабрь 2014 г.. - 12 Апрель 2016 г.. - http://www.coinside.ru/2014/12/04/pobit-novyj-rekord-v-100-tysyach-tranzaktsij-v-den/.

РАЗРАБОТКА ГЕНЕТИЧЕСКОГО АЛГОРИТМА ДЛЯ РЕШЕНИЯ ЗАДАЧИ ОПТИМИЗАЦИИ ГРАФИКА ДОСТАВКИ ГРУЗОВ

Костин К.В. – студент, Кантор С.А. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

Эффективность работы некоторых предприятий в значительной степени зависит от решения задачи маршрутизации, которое позволяет сократить транспортные издержки. В настоящей работе рассматривается задача построения графика доставки грузов, возникшая при оптимизации бизнес-процессов одной компании из г. Барнаула.

Постановка задачи может быть сформулирована следующим образом. Необходимо используя заранее предопределенное количество транспортных средств выполнить заявки на перевозку грузов либо со склада компании потребителю, либо наоборот. Для каждой заявки имеется следующая информация: тип доставка (доставка потребителю или от потребителя); временные промежутки, в которые необходимо совершить загрузку и разгрузку; вес и объем груза; время загрузки и разгрузки. Заявка является невыполненной, если не была произведена перевозка груза. Время просрочки вычисляется как длительность работ по погрузке/выгрузке, не попадающих в промежутки времени, оговоренные в заявке. Если это время равно нулю, то заявка считается выполненной, иначе, просроченной.

Для каждого транспортного известна следующая информация: начало и конец рабочего дня; ограничения на массу и объем перевозимого груза. Вне рабочего времени все транспортные средства должны находиться на складе компании.

Считается, что во время перевозки транспортным средством груз нельзя разделять на части, то есть необходимо выполнить его перевозку за один раз.

Построенный график доставки должен удовлетворять следующим условиям:

- для каждого транспортного средства загрузки и разгрузки укладываются в выделенное для него рабочее время;
- в транспортном средстве одновременно не перевозится груз, суммарная масса или объем которого больше допустимого.

Так же график должен быть оптимизирован по следующим критериям, перечисленным в порядке приоритета:

- минимизация количества невыполненных заявок;
- минимизация общего времени просрочки;
- минимизация суммарного времени работы всех транспортных средств.

Данная задача является NP-полной [1], так как ее частный случай без учета времени доставки и с одним транспортным средством, грузоподъемность и вместимость которого позволяет одновременно перевезти все грузы, является задачей коммивояжера [2]. Следовательно, поставленная задача не имеет эффективных алгоритмов поиска точного решения, поэтому необходимо найти приемлемое решение за заданное время, так как от этого напрямую зависит качество работы организации.

Существует целый класс алгоритмов, позволяющих находить приближенное решение данной задачи, называемых эвристическими [3]. Эвристический алгоритм – это алгоритм, правильность которого в общем случае не доказана, но про который известно, что на практике он дает достаточно хорошее решение. Для решения был выбран генетический алгоритм [4-5], одно из преимуществ которого – простота при его распараллеливании.

Чтобы алгоритм был приспособлен для решения более общей задачи, допустим, что для выполнения одной заявки нужно совершить не просто загрузку, доставку и разгрузку груза, а целый список операций. То есть транспортное средство должно прибыть в пункт по адресу совершения первой операции, совершить первую операцию, далее прибыть в пункт по адресу совершения второй операции, совершить вторую операцию, и т.д. Кроме того, в

промежутке между совершениями операций одной заявки, могут выполняться любые другие операции. Заявка считается невыполненной до тех пор, пока не будет совершена последняя операция в ее списке.

Для реализации алгоритма необходимо определить структуру организма. Выбрана гибридная структура, состоящая из двух частей:

1) общий список операций;
2) частные списки операций для каждого транспортного средства.

Про данные списки известно, что если одна операция должна быть совершена раньше другой, то и в списке она находится раньше.

При построении расписания на основе общего списка необходимо учитывать то, что все операции одной заявки должны находиться в одном маршруте. Поэтому используется следующий алгоритм:

1. начальный построенный маршрут для транспортного средства считается пустым;
2. из списка выбирается первая не совершенная операция, и заявка, в которую она входит, считается текущей рассматриваемой;
3. в маршрут добавляются все операции текущей рассматриваемой заявки;
4. операции в маршруте сортируются в порядке появления в исходном списке $tasks$;
5. если маршрут удовлетворяет всем необходимым критериям, то он считается построенным, операции рассматриваемой заявки помечаются как совершенные, и процесс построения переходит к пункту 2. Если маршрут не удовлетворяет хотя бы одному необходимому критерию, то предыдущий построенный маршрут добавляется в расписание, и начинается процесс построения маршрута для следующего транспортного средства, начиная с пункта.

Частные списки для каждого транспортного средства строятся из списка операций соответствующего ему маршрута.

Если организм строится на основе частных списков, то общий список строится как их конкатенация. Далее происходит построение маршрутов ранее описанным алгоритмом, с учетом того, что для каждого транспортного средства используется свой список операций.

Любую перестановку элементов списка можно осуществить последовательностью перестановок соседних элементов. Это свойство используется в операторах мутаций, для того, чтобы не нарушать последовательность операций одной заявки в списке. Две соседних операции меняются тогда и только тогда, когда они принадлежат разным заявкам.

В алгоритме используются следующие мутации над общим списком операций: переворот случайного подсписка, обмен местами двух случайных операций, перемещение случайной операции в случайное место списка, циклический сдвиг случайного подсписка на случайное количество элементов. Над частными списками операций применяются все ранее описанные мутации. Кроме них добавляется еще перемещение заявки между списками.

Кроме мутаций, для улучшения решения применяются следующие оптимизации:

- на каждом подсписке длинны *len* каждого частного списка перебираются все перестановки операций, и из найденных решений выбирается наилучшее. В результате численных экспериментов выбрано наиболее приемлемое значение *len*, равное 6;
- перебираются все пары операций в каждом частном списке и они обмениваются местами, из найденных решений выбирается наилучшее.

В рамках данной работы реализован многопопуляционный генетический алгоритм [6], главной характеристикой которого является использование небольшого числа относительно больших подпопуляций и миграция особей между ними.

Частные списки операций для каждого транспортного средства, а так же дополнительные мутации с их использованием введены для того, чтобы находить более

точные решения к уже найденным приближениям, так называемый локальный поиск [7]. Однако, над такими организмами возможны и мутации с общим списком операций, что может кардинально поменять всю структуру организма – глобальный поиск [7]. Для достижения компромисса между локальным и глобальным поиском, используется два типа популяций. В первом – мутации с общим списком операций, во втором – мутации над частными списками, причем миграция возможна только из первого типа во второй. Кроме этого существуют оптимизации, которые следует применять только к наилучшим организмам. Поэтому используется еще одна популяция, в которую организмы переходят только из второго типа.

Для уменьшения вероятности нахождения локального минимума, через некоторое временя без улучшения решения происходит удаление всех организмов их всех популяций, кроме оптимизирующей. После этого необходимо снова генерировать начальное поколение и распространять его по всем популяциям. Для этого введена популяция, которая только генерирует новые организмы и передает их в первый куб.

Итоговый граф миграций, используемый в алгоритме, представлен на рисунке 1.

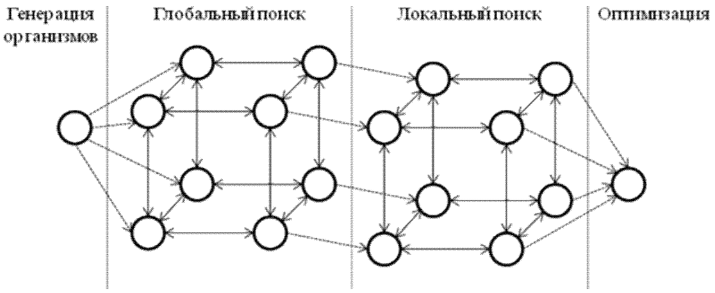


Рисунок 1 – Граф миграций. Вершины – популяции, ребра – направление миграций

Тестирование выполнялось на реальных данных компании для выполнения 30 заявок с использованием 2 транспортных средств. Алгоритм показывает достаточные для практического использования результаты за время работы в 30 секунд.

Тестирование на 183 заявках (данные за 7 дней) и 5 транспортных средствах показывает следующие результаты:

	Время работы программы (сек.)									
	10	20	30	40	50	60	70	80	90	100
Количество невыполненных заявок	96	91	88	88	88	86	85	85	85	84
Время просрочки (мин.)	60	54	47	38	40	37	38	38	38	33
Время работы транспортных средств (мин.)	3147	3145	3143	3132	3137	3139	3164	3164	3160	3149

Для тестирования был написан метод полного перебора всех решений. Уже на выполнение 9 заявок с использованием 2 транспортных средств его работа занимала около одного дня, в то время как разработанный алгоритм за 30 секунд построил оптимальное решение 19 раз при 20 испытаниях.

Список литературы

1. Michael R.G. Computers and Intractability: A Guide to the Theory of NP-Completeness. / R. G. Michael, S.J. David- NewYork: W.H. Freeman and Company 1979. - 339 p.
2. Задача коммивояжера [Электронный ресурс]. – Режим доступа: http://www.math.nsc.ru/LBRT/k5/OR-MMF/TSPr.pdf

3. Michalewicz Z. How to solve it. Modern heuristics. / Michalewicz Z., Fogel D.B Berlin. - Springer, 2004. – 155 p.
4. Pereira Francisco B. Bio-inspired algorithms for the vehicle routing problem / Francisco B. Pereira, Jorge Tavares. - Coimbra: Springer, 2008 - 221p
5. Генетический алгоритм [Электронный ресурс]. - Режим доступа: http://www.machinelearning.ru/wiki/index.php?title= Генетический_алгоритм.
6. A survey of parallel genetic algorithms [Электронный ресурс]. - Режим доступа: http://tracer.uc3m.es/tws/cEA/documents/cant98.pdf
7. A Hybrid Genetic Algorithm for Multi-Depot and Periodic Vehicle Routing Problems [Электронный ресурс]. - Режим доступа: https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2011-05.pdf

ФРЕЙМВЕРК ДЛЯ РАСПРЕДЕЛЕНИЯ ЗАДАЧ С ИСПОЛЬЗОВАНИЕМ СЕТЕЙ ФУНКЦИОНАЛЬНЫХ ВЫЧИСЛЕНИЙ

Кристалев Д.А.– студент, Старолетов С.М. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Актуальность

Мир меняется и требования к программам меняются вместе с ним. Еще несколько лет назад действительно большие приложения имели десятки серверов, секундные отклики, часы офлайнового обслуживания и гигабайты данных. Сегодня, приложения разворачиваются на чем угодно, начиная от мобильных платформ, заканчивая облачными кластерами, работающими на основе тысяч многоядерных процессоров. Пользователи ожидают миллисекундное время отклика и бесперебойное время работы. Данные меряться в петабайтах. Сегодняшним требованиям уже просто не могут удовлетворять вчерашние архитектуры приложений. На наш взгляд, на сегодняшний день наиболее актуальные требования к системам следующие:

- Отзывчивость. Система реагирует своевременно, если это возможно.
- Отказоустойчивость. Система продолжает отвечать, даже при сбоях.
- Масштабируемость. Система должна легко переноситься (распространятся) на другое железо, не теряя при этом отзывчивость.

Сегодня стал набирать сторонников функциональный подход к программированию (ФП). Это связанно с некоторыми уникальными особенностями такого подхода. К таким особенностям можно отнести: отсутствие побочных эффектов, краткость программ и их выразительность по сравнения с императивными программами, «ленивые» вычисления, возможность «горячей» замены кода и т.д. Благодаря отсутствию побочных эффектов, присущих ФП, отсутствует множество проблем, связанных с распараллеливанием программ, например, нет проблемы «гонки данных». Это и есть главный фактор, популяризующий сегодня функциональный подход. Но у такого подхода есть и недостатки, к которым можно отнести: обязательный высокоэффективный сборщик мусора, больший, по сравнению с императивными программами, необходимый объем памяти, относительно долгая скорость компиляции и т.д. Однако, эти проблемы, на сегодняшний день, в большей степени удалось решить и поэтому использование функциональных языков стало привлекательным. К функциональными языками, которые сегодня представляют наибольший интерес, относят такие языки как Scala и Erlang.

Постановка задачи

Была поставлена задача реализовать фреймворк, который позволит использовать, как и Erlang, с рядом его уникальных решений, так и Scala с JVM и богатой инфраструктурой, для создания распределенных систем, использующих акторную систему. Именно благодаря высокому уровню абстракции акторов, должна быть возможна передача сообщений между системой, написанной на Erlang, и системой, написанной на Scala, и потенциально другими системами. Фреймверк должен позволять создавать связи между двумя подсистемами, а так же предоставлять возможность создания API для обращения к такой системе и получения от нее ответов.

Подход к решению проблемы

Сегодня есть несколько решений, которые могут соответствовать потребностям отказоустойчивости, отзывчивости и масштубируемости: STM, Dataflow Concurrency и Actors. Рассмотрим подход с акторной моделью (Actors). Приложения, написанные согласно акторной модели, удовлетворяют перечисленным требованиям. Помимо этого, приложения, основанные на акторах, полагаются на асинхронную передачу сообщений и это есть главная особенность этого подхода

Актор (в агентно-ориентированном программировании и в модели акторов) – программная сущность заданной структуры и механизмов взаимодействия. В актороной модели – механизм взаимодействия – это передача сообщений.

В 1987 году первым языком, который был создан, для того, чтобы реализовать данную модель, был функциональный язык Erlang[1]. С тех пор этот язык используют для создания отказоустойчивых, распределенных систем. На самом деле, Erlang подходит для решения многих современных задач. К плюсам языка можно отнести: легковесные процессы, планировщик процессов, OTP, функциональная парадигма и т.д. Недостатками являются: необходимость непопулярной виртуальной машины BEAM, малочисленное и медленно развивающееся сообщество, из-за чего у языка почти нет открытых фреймверков, кроме стандатного OTP, некоторые недостатки синтаксиса языка.

Сегодня стали появляться альтернативы Erlang с его акторной моделью. Например, язык Go [2], разрабатываемый Google, реализует такую модель, правда со своими особенностями, такими как каналы. Однако есть система, которая очень напоминает Erlang, но при этом лишена недостатков этого языка. Это решение Scala + Akka.

Scala – это современный мультипарадигменный язык программирования. Scala развивается, у него активное сообщество, а главное этот язык использует JVM с её JIT компилятором, умным GC и богатой инфраструктурой. На этом языке была разработана библиотека Akka [3], которая реализует акторную модель во многом схожую с той, что используется в Erlang.

Для примера посмотрим приложения, написанные на языке Erlang и Scala, реализующих игру двух акторов в пинг-понг.

Пинг-понг:

- Один актор шлет другому Ping и ждет ответа.
- Другой получает Ping, и шлет Pong в ответ.
- Акторы запускаются параллельно, причем, они должны знать друг о друге, для того чтобы слать сообщения.

Реализация пинг-понг на языке Erlang[4]:

```
    -module('pingpong').
%% API
-export([main/0]).

%%%функция, инициализирующая пинг-понг. точка входа в программу.
main()->spawn(fun() ->ping(spawn(fun()->pong() end))end).

%%% процесс(актор) ping, получает pid процесса pong для отправки ему
сообщения
ping(PidPong)->
  io:format("I'm ping, pids are:  ~w ~w ~n ", [PidPong,self()]),
  PidPong ! {ping, self()}, %%%отправка сообщения ping
  receive
    pong -> io:format("PONG received~n") %%%получение ответа и вывод
сообщения об этом
  end.

%%% процесс(актор) pong
pong() ->
  io:format("I'm pong, pids are:  ~w ~n ", [self()]),
  receive
    {ping, PidPing}-> io:format("PING received, sending pong~n "),
%%%получение сообщения ping и вывод сообщения об этом
      PidPing ! pong %%%отправка ответа
  end.
```

Результат работы программы:

I'm ping, pids are: <0.30.0> <0.29.0>
I'm ping, pids are: <0.30.0>
PING received, sending pong
PONG received

Реализация пинг-понг на языке Scala:

```
    class PingServer extends Actor{
  override def receive = {
    case "Ping" => {  //получение сообщения ping и вывод сообщения об этом
      println(Thread.currentThread().getName() + ": Ping!")
      sender ! "Pong" //отправка ответа
    }
    case _ => println("Unknow message!")
  }
}
    //актор, которому при создании указывается на какой сервер слать
сообщения
class PongClient(server: ActorRef) extends Actor{
  override def receive = {
    case "Start" => server ! "Ping"   //отправка сообщения на сервер
    case "Pong" => {
      println(Thread.currentThread().getName + ": Pong!") //получение
ответа и вывод сообщения об этом
      //остановка акторной системы
      context.stop(server)
      context.stop(self)
      context.system.terminate();
    }
    case _ => println("Unknow message!")
  }
}
    object Main extends App {
  val system = ActorSystem("PingPong");   //новая акторная система
  val server = system.actorOf(Props[PingServer])    //создание и запуск
```

```
актора PingServer
  val client = system.actorOf(Props(new PongClient(server))) //создание и
запуск актора PongClient
  client ! "Start"  //отправка инициализирующего сообщения
}
```

Результат работы программы:

```
PingPong-akka.actor.default-dispatcher-3: Ping!
PingPong-akka.actor.default-dispatcher-4: Pong!
```

Как видно в обоих примерах, акторы действительно работают в разных потоках, а процесс получения и отправки сообщений во многом похож.

Проект фреймверка

В настоящий момент реализуется программный фреймворк, который объединяет представленные решения и позволит

- Регистрировать Erlang и Scala системы
- Обеспечивать общение Erlang процессов и акторов Scala между собой
- Ставить задачи гибридной системе.

Схема решения представлена на рисунке 1. Для взаимодействия с JVM со стороны Erlang будет использоваться JInterface, поставляемый как часть библиотеки Erlang.

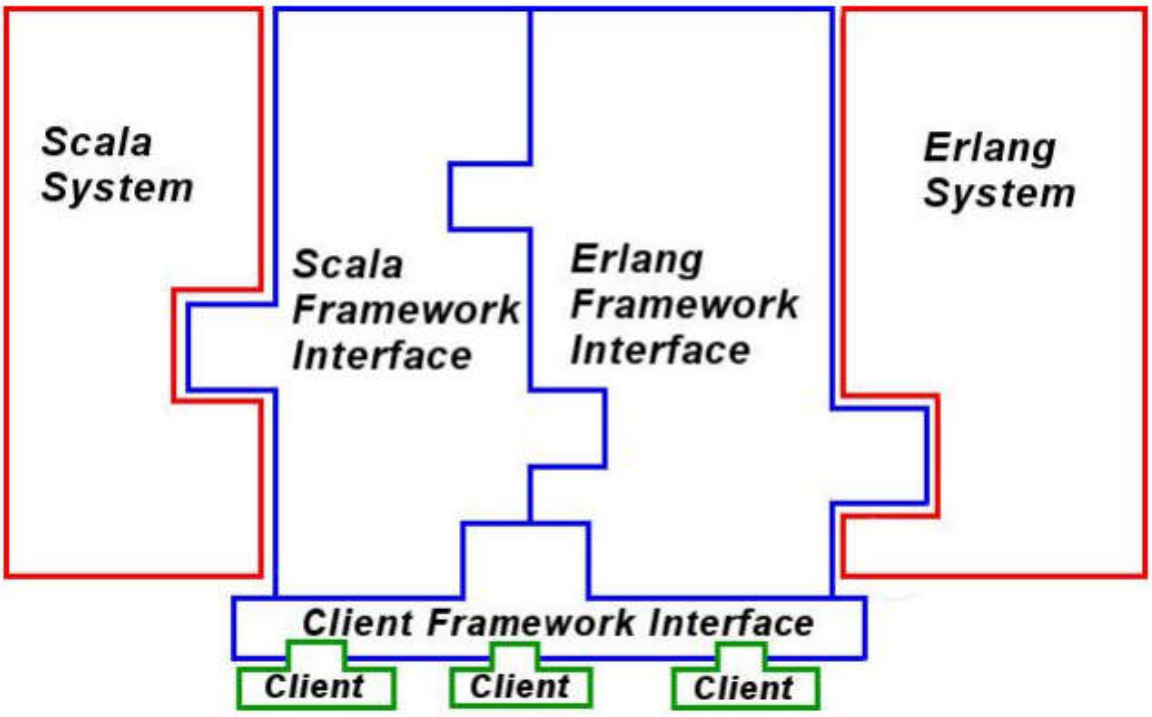


Рисунок 1 – предполагаемая схема решения

Список литературы

1. Erlang Programming Language [электронный ресурс]: Режим доступа: https://www.erlang.org/
2. The Go Programming Language. [электронный ресурс]: Режим доступа: https://golang.org/
3. Getting Started Tutorial (Scala): First Chapter [электронный ресурс]: Режим доступа: http://doc.akka.io/docs/akka/2.0/intro/getting-started-first-scala.html
4. Старолетов С.М. Функциональные языки распределённых систем: Учебно-методическое пособие /С. М. Старолетов.- Барнаул : АлтГТУ, 2015. – 81 с.

ОПТИМИЗАЦИЯ СИСТЕМЫ УДАЛЕННОГО УПРАВЛЕНИЯ НА КОМПЬЮТЕРАХ С ОПЕРАЦИОННОЙ СИСТЕМОЙ СЕМЕЙСТВА MS WINDOWS

Фаст А.С. – студент
Алтайский государственный технический университет (г. Барнаул)

В работе [1] рассматривается реализация системы, позволяющей удаленно работать на другом компьютере. Целью данной работы является комплексная оптимизация этой системы. Управляемый клиент на своей стороне реализует компрессию и архивирование изображений перед передачей, что позволяет рассмотреть сразу несколько направлений оптимизации: во-первых, по умолчанию применяется довольно сильное сжатие изображений, соответственно теряется качество, во-вторых, если между клиентами канал передачи данных достаточно широкий, то производить компрессию нет смысла, ведь при этом теряется качество изображения, в-третьих, на компьютерах, обладающих относительно большой вычислительной мощностью, можно использовать более эффективные алгоритмы компрессии и архивирования.

Проблему оптимизации можно кратко сформулировать следующим образом: необходимо максимально эффективно использовать имеющиеся в распоряжении технические ресурсы, такие как скорость интернет-соединения и производительность компьютера, не доставляя при этом неудобств пользователям.

Среди алгоритмов сжатия изображений выделяют две большие группы: сжатие без потерь и сжатие с потерями. Первая группа алгоритмов уменьшает размер данных, необходимых для хранения изображения, не ухудшая его качество. Стоит отметить, что уменьшение объема данных в данном случае будет довольно незначительным. С помощью второй группы алгоритмов можно значительно уменьшить объем памяти, необходимый для хранения изображения, но при этом качество изображения может ухудшиться.

В качестве алгоритмов сжатия без потерь были исследованы следующие алгоритмы: LZW (реализован в формате GIF) и алгоритм Хаффмана (реализован в формате PNG). Среди алгоритмов сжатия с потерями были выбраны JPEG, как самый популярный формат в этом классе, и JPEG2000.

В процессе исследования нас интересовали следующие характеристики комбинаций алгоритмов сжатия и формата файла: скорость работы, объем выходного файла и условное качество изображения. Качество изображения будем оценивать по условной пятибалльной шкале, от 1 «текст не читаемый», до 5 «чёткое изображение». Все эксперименты будем проводить с изображением, имеющим разрешение 1920x1080 пикселей и с текстом высотой около 12 пикселей. Очевидно, что некоторые итоговые параметры могут зависеть от характера изображения, к тому же в режиме удаленной работы могут появляться изображения разнопланового содержимого и различной цветовой гаммы, поэтому каждый из экспериментов был проведен с изображениями следующих типов: преобладает светлый текст на темном фоне, преобладает темный текст на светлом фоне, открыто много различных окон с мелкими элементами. Все полученные результаты для разных характеров изображений были усреднены.

Для примера рассмотрим формат JPEG. При сохранении в этот формат средствами .NET возможно установить значение качества от 0, до 100. Рассмотрим зависимость размера и качества выходного изображения от этого значения (Рис. 1).

Размер исходного изображения в этом эксперименте составлял около 200 килобайт, но уже при 83 килобайтах после архивирования, при коэффициенте качества 40, можно получить изображение качества, достаточного для получения оценки «5». Также важно то, что до коэффициента качества 40 зависимость размера от качества близка к линейной. При это фактическое время работы алгоритма практически не изменяется при изменении качества.

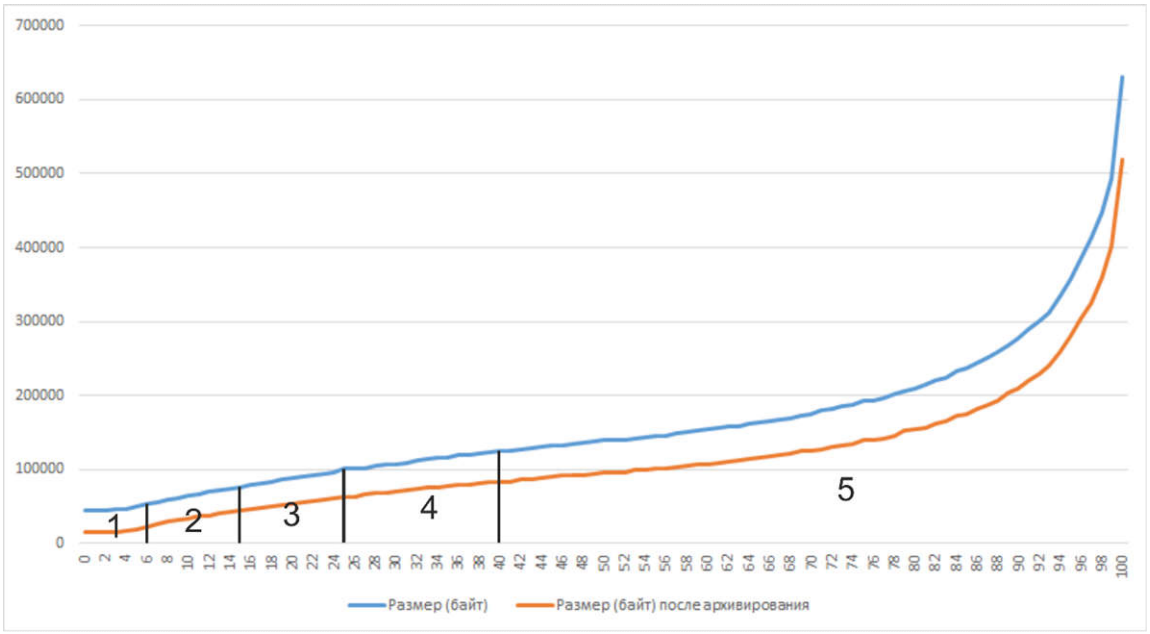


Рисунок 1

Алгоритм сжатия JPEG2000 в среднем показал немного лучшие по сравнению с JPEG характеристики, но при этом конвертация в этот формат занимает существенно большее время, поэтому даже на производительных компьютерах его использование не имеет смысла.

В ходе исследования алгоритмов сжатия без потерь были найдены несколько комбинаций форматов файлов и алгоритмов сжатия таких, которые дают размер файла, сопоставимый с JPEG, получающим «5» баллов. Например, формат файла TIFF и компрессия с помощью алгоритма LZW. Но при этом все подобные схемы сжатия практически не подвержены архивированию, поэтому использовать их есть смысл при достаточной пропускной способности канала между двумя компьютерами.

Алгоритм оптимизации.

На основе проведенных исследований был разработан следующий алгоритм работы системы:

- Перед началом сеанса удаленной работы происходит отправка не архивированного изображения рабочего стола и массива байт ровно в 2 раза меньшего размера с целью измерения относительной пропускной способности и построения примерной зависимости скорости передачи от размера файла;
- Параллельно с этим программа запускает следующие алгоритмы сжатия: JPEG с качеством 0 и 40 для построения зависимости размера файла от качества, TIFF с компрессией LZW и другие схемы сжатия без потерь, в частности архивирование, и измеряет время работы всех типов сжатия;
- Далее, исходя из предположения, что для комфортной работы достаточно того, чтобы была возможность полностью обновлять изображения экрана 5-6 раз в секунду, программа на управляемой стороне осуществляет подбор параметров сжатия так, чтобы суммарное время работы сжатия и передачи удовлетворяло требованиям частоты обновления. Если подобрать параметры так, чтобы изображения получало хотя бы оценку «3» не удалось, то частота обновления опускается до 5 раз в секунду, потом до 4, до 3 и до 2. Если и при такой частоте не удалось подобрать параметры, то программа пытается использовать худшее качество изображения и вне зависимости от того, подходит оно под требования частоты или нет, далее работает именно с ним.

- После того, как программа подобрала параметры сжатия и формат файла, она отправляет данные об этом на управляющую сторону, чтобы принимающая программа могла корректно отображать полученные изображения и после этого начинается сеанс работы.

Вывод. После проведения вышеописанной комплексной оптимизации системы существенно повысилось её быстродействие на слабых компьютерах и значительно улучшилось качество передаваемого изображения, что было отмечено пользователями системы.

Список литературы

1. Фаст А.С. Организация удаленного управления на компьютерах с операционной системой семейства MS Windows [Электронный ресурс] / А.С. Фаст, // Материалы XII Всероссийской научно-технической конференции студентов, аспирантов и молодых ученых «Наука и молодежь – 2015». Секция «Информационные технологии». Подсекция «Программная инженерия»:сб. статей. – Барнаул, 2015. – сс. 30-33.
2. Миано Д. Форматы и алгоритмы сжатия изображений в действии [Текст] / Д. Миано – Москва: Триумф, 2003. – 336 с.
3. Павлидис Т. Алгоритмы машинной графики и обработка изображений / Т. Павлидис - Москва: Радио и связь, 1986. - 400 с.

МЕТОДОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ A.D.D.

Старолетов С.М. – к.ф.-м.н., доцент

Алтайский государственный технический университет (г. Барнаул)

В данной статье впервые предлагается методология разработки программного обеспечения – Article Driven Development (разработка, основанная на написании статей).

Актуальность

Сегодня в программной инженерии существуют некоторые нерешенные проблемы:

- Написание кода без предварительного проектирования (разработчики часто начинают разработку с написания кода, а не проектирования, обдумывания, анализа существующих решений, подходов, библиотек).
- Ошибки (известно, что доказать безошибочность конкретной программы невозможно).
- Часто приходится проводить ре-дизайн системы (от заказчиков часто приходят неполные спецификации и требования к системе меняются в процессе разработки).
- Отсутствие документации (частичное или полное: программы меняются, они слабо описаны, документация для пользователя и для разработчика пишется редко и она не полна).
- Программные системы часто полны внутренних противоречий (спецификации при разработке системы меняются в процессе разработки, существуют забытые функции, унаследованный код, функциональность системы часто отличается от предполагаемой).
- Мало научной составляющей (к сожалению, программная инженерия направлена на решение конкретных проблем, новые методы и методологии возникают не часто).

Конкретно, из опыта преподавания на специальности “Программная инженерия”, у студентов имеются следующие проблемы:

- Студенты хоть как-то еще могут реализовать поставленную задачу, с использованием “костылей”, нехороших программных подходов, методов, приемов программирования, которые приводят к частым ошибкам, к неподдерживаемому коду, к неэффективности и не расширяемости в дальнейшем. Но, тем не менее, такие системы в целом рабочие.
- Однако привести хорошее описание, специфицировать поведение реализованных программных систем студенты не могут, не успевают, не хотят, не мотивированы, в итоге даже хорошие программные системы очень плохо описаны, специфицированы, промоделированы.

Следствия всего этого:

- Малое число публикаций как студентов, так магистрантов и аспирантов.
- Темы работ скорее технические, чем исследовательские.
- Низкий уровень цитирования наших работ в системах цитирования.
- Слабые позиции университета в рейтинге.

Даже лучшие студенты, призеры олимпиад по программированию и математике, хорошо пишут лишь программы. Максимум, что может получиться у них – это статья в сборниках тезисов конференции. Публикация в научном журнале для программных инженеров сильно осложнена. В дальнейшем они будут заниматься разработкой ПО в компаниях, а не исследованиями.

Следовательно, необходима методология разработки программ, которая позволит программным инженерам разрабатывать программные системы, которые будут при этом лишены перечисленных проблем, при этом так, чтобы разработанная программная система нашла свое отражение в научной (или в научно-технической статье).

Предлагаемое решение задачи

В работе предлагается реализовывать программное обеспечение совместно с написанием научной статьи. Написание статьи позволит применить подход обдумывания каждой детали в решении. Поскольку практически невозможно побудить будущего исследователя написать статью просто так, поэтому необходимо, чтобы статья писалась прямо во время разработки программного обеспечения. В этом случае при завершении разработки софта будем иметь и программное обеспечение и статью.

Процесс разработки предлагается назвать ADD (Article driven development), разработка, управляемая написанием статьи. Название “ADD” соответствует современным тенденциям в программной инженерии по наименованию методологий.

Наименование

Ранее в программной инженерии применялось семейство названий *ОП (способ программирования), например:

- ООП(OOP) – объектно-ориентированное программирование.
- АОП(AOP) – аспектно-ориентированное программирование [1] / агентно-ориентированное программирование[2].
- ЯОП(LOP) – программирование, ориентированное на создание языка [3].
- Сейчас в инженерии применяется семейство *DD (определяющее способ разработки):
- TDD – test driven development (разработка через тестирование) [4].
- BDD – behavior driven development (разработка через поведенческие спецификации)[5].

- MDD – model driven development (разработка через создание моделей)[6].

Поскольку темой данного исследования является разработка, то целесообразно применять последнюю нотацию.

Имеющиеся работы по теме исследования

На презентации “How to write a great reseach paper” [7], представленной в МГУ, Саймон Пейтон Джонс представил свой подход, в котором он призывает писать статьи по любому поводу и даже перед собственно проведением исследования. Это перекликается с идеями, представляемыми в настоящей работе, однако не оформлено в методологию разработки.

В методологии TDD Кент Бек представил подход, при котором разработка ПО начинается с разработки тестов. Непройденный тест побуждает разработчика написать новый функционал или исправить существующий. При этом используются свойства человеческого мозга, такие как инкрементность мышления (разработка идет с нуля постепенно, пишутся маленькие тесты и функционал, чтобы они проходили), ленивость (на каждом шаге реализуется лишь минимально возможный функционал, чтобы проходили написанные тесты и ничего больше), ориентированность на события (непройденный тест вынуждает дополнять или изменять код). В исследовании предлагается использовать похожий подход, при котором исследовательская статья будет побуждать писать код.

Предлагаемая методология

ADD начинается с создания статьи. Статья начинается с темы, которую нужно придумать. После этого пишется краткая аннотация статьи, которая включает в себя информацию, зачем предназначено данное ПО, какие проблемы оно будет решать и как в целом. Этого уже достаточно, чтобы начинать писать код программного обеспечения, при этом обязательно возникнут проблемы, которые можно решать по предлагаемой методологии. Основной процесс промоделирован на рисунке 1.

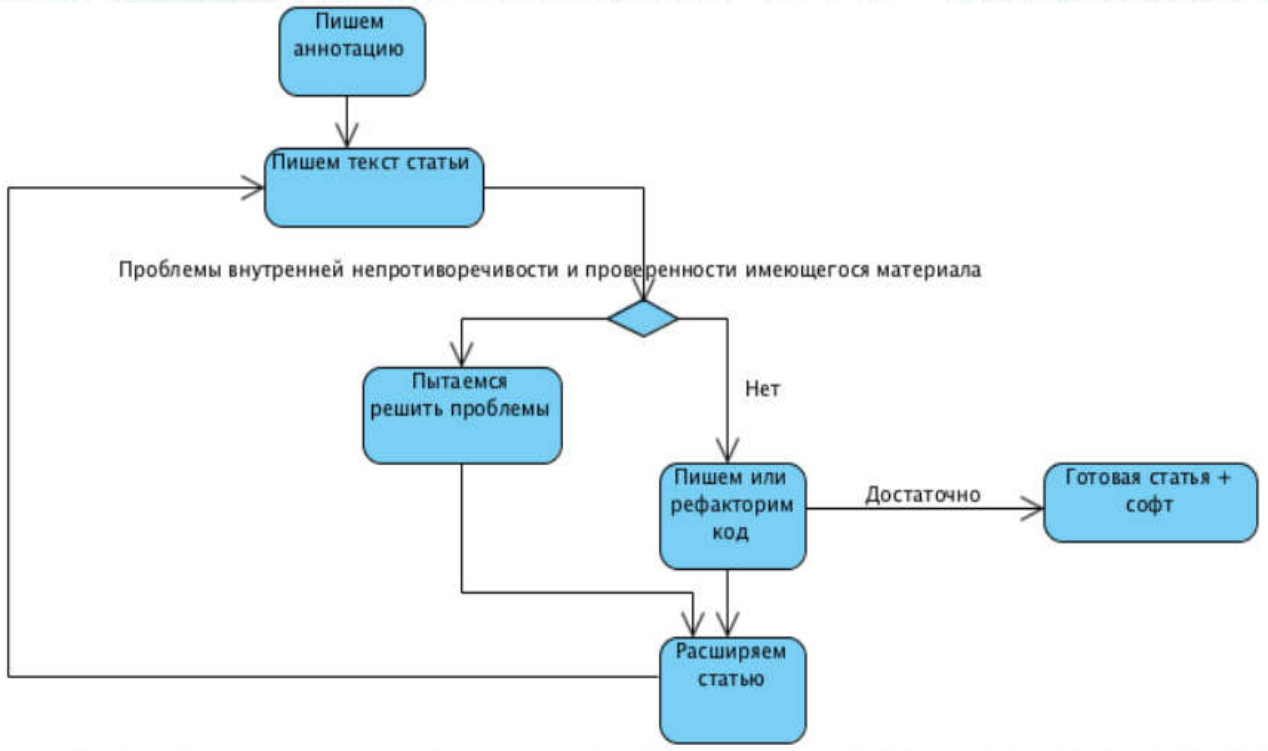


Рисунок 1 – основной процесс ADD

При разработке программы разработчик может столкнуться с следующими проблемами:

- Необходимо реализовать какой-то сложный алгоритм или найти готовую библиотеку, решающую его.
- Необходимо дополнительно продумать архитектуру приложения, спроектировать что-то (диаграмму классов, БД, архитектурные диаграммы и т.д.).
- Возникают ошибки, для завершения разработки не хватает навыков, необходимо задать вопрос или найти ответ по возникшей проблеме (например, на stackoverflow.com).
- Нужно спросить совета научного руководителя или коллег, т.к. нет своих идей, что делать дальше.
- Оказалось, что то, что хотелось сделать, уже было реализовано.

Эти проблемы должны побуждать разработчика временно отложить разработку, провести некоторые действия и написать либо дополнить некоторый раздел в статье.

Проблема с алгоритмами или библиотеками может быть решена следующим образом:

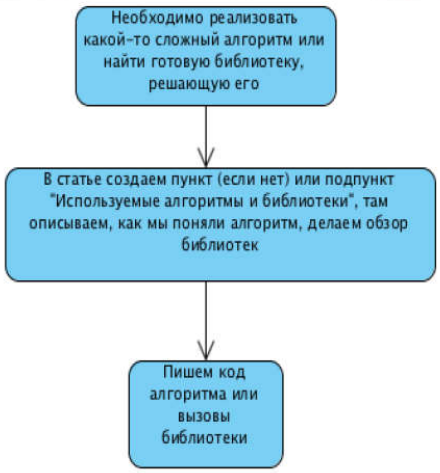


Рисунок 2 – Разрешение проблем с алгоритмами

Необходимость в проектировании может быть разрешена следующим образом:

Рисунок 3 – Разрешение проблем с проектированием

Необходимость поиска может быть разрешена так:

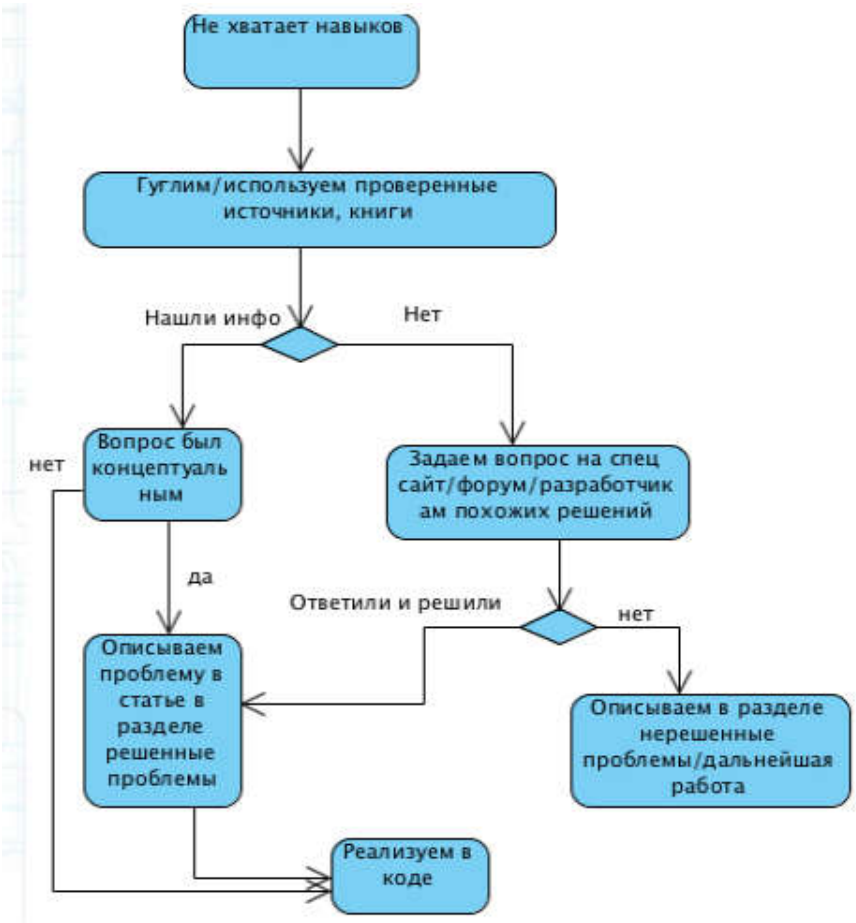


Рисунок 4 – Разрешение проблем с необходимостью поиска информации

При разработке часто необходимо разделить свои идеи с коллегами или научными руководителями. Преимуществом ADD является то, что на каждом шаге имеется описание в некотором виде, которое может быть использовано как документ для прочтения и обсуждения.

Рисунок 5 – Обсуждение разработки

Одной из часто встречающихся проблем при разработке является проблема информационного поиска и анализа готовых решений. Часто то, что разрабатывается, в каком-то виде уже реализовано. Предлагаемая методология решает и эту проблему, ведь даже при полном соответствии чужой программы нашим идеям мы имеем обзорную статью на выходе.

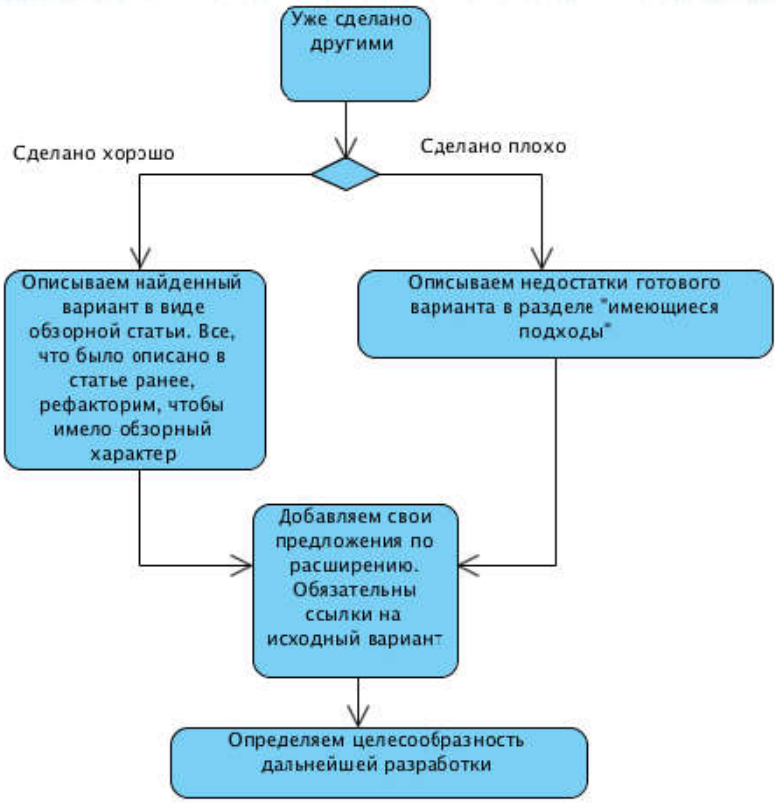


Рисунок 6 – Разрешение проблем с уже имеющимся аналогичным ПО

Результат

В результате разработки при помощи ADD возможно получить и готовое ПО и готовую статью, либо готовую статью, если разработка ПО оказалась нецелесообразной. При этом, данный процесс может быть объединен с имеющемся процессом разработки ПО. Например, если ПО реализуется по TDD, то при добавлении ADD перед написанием новых тестов можно их описывать при помощи ADD в статье (проблемы “необходимость в проектировании”), далее реализовывать тесты и писать код, для разработки по MDD в этом-же разделе описывать или корректировать модель. В результате процесса будем иметь более обдуманное и описанное программное обеспечение, которое может быть вынесено на публичное обсуждение далее.

Что касается времени разработки, то по оценкам, оно будет замедленно около в 5 раз (по оценкам, TDD уменьшает время разработки в 2 раза), следовательно, в промышленных масштабах быстрой разработки ПО данный подход не может быть применен. Его следует использовать для разработки научного ПО, либо критически важного ПО, разработка, поддержка и жизненный цикл которого может вестись годами, оно применимо лучше всего к разработке алгоритмов, фундаментальных подходов, которые могут быть применены в большом количестве областей и т.д.

В настоящее время на данной конференции уже были представлены работы, разработка которых начиналась именно с написания статьи, что доказывает практическую применимость данной методологии.

Список литературы

1. Joseph D. Gradecki, Nicholas Lesiecki, Mastering AspectJ: Aspect-Oriented Programming in Java. – Wiley, 2003. ISBN-13: 978-0471431046
2. Matthew M. Huntbach, Graem A. Ringwood. Agent-Oriented Programming: From Prolog to Guarded Definite Clauses (Lecture Notes in Computer Science / Lecture Notes in Artificial Intelligence) – Springer, 1999. ISBN-13: 978-3540666837
3. Ward M.P. Language Oriented Programming [Электронный ресурс] – Режим доступа: http://www.cse.dmu.ac.uk/~mward/martin/papers/middle-out-t.pdf
4. Кент Бек. Экстремальное программирование. Разработка через тестирование. - СПб.: Питер, 2003, 224с.: ил. ISBN: 5-8046-0051-6
5. John Ferguson Smart. BDD in Action. Behavior-Driven Development for the whole software lifecycle. Manning, 2014. – 384p. ISBN: 9781617291654.
6. Staroletov Sergey. Model Driven Developing & Model Based Checking: Applying Together [Электронный ресурс] – Режим доступа: http://www.slideshare.net/IosifItkin/model-driven-developingmodelbasedchecking
7. Simon Peyton Jones. Research skills [Электронный ресурс] – Режим доступа: http://research.microsoft.com/en-us/um/people/simonpj/papers/giving-a-talk/giving-a-talk.htm

ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ СИСТЕМЫ ВИДЕОНАБЛЮДЕНИЯ ЗА НЕРЕГУЛИРУЕМЫМИ ПЕШЕХОДНЫМИ ПЕРЕХОДАМИ

Колосовский М.А. – к.т.н., н.с., Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В этой статье представлены результаты экспериментального исследования интеллектуальной системы видеонаблюдения за нерегулируемыми пешеходными переходами [1] с точки зрения качества работы, производительности и устойчивости к неблагоприятным условиям съемки. Задача системы видеонаблюдения состоит в обнаружении нарушений правил дорожного движения на нерегулируемых пешеходных переходах, главным образом, непредоставления преимущества в движении ожидающим на тротуаре пешеходам. Основной трудностью внедрения таких систем являются недостаточно высокие показатели их работы такие, как производительность и точность и полнота обнаружения нарушений.

В связи с тем, что обработка информации в представленной системе видеонаблюдения состоит из нескольких звеньев: детектор активности, детектор объектов, трекер объектов, модель анализа траекторий, рассмотрим **качество работы** после каждого из звеньев обработки, чтобы оценить вклад каждого из звеньев.

Качество работы после различных этапов обработки данных в системе измерялось по следующим метрикам:

- *точность* — отношение числа правильно обнаруженных объектов к общему числу обнаружений, выданных системой;
- *полнота* — отношение числа правильно обнаруженных объектов к истинному количеству объектов;
- *F-мера* — среднее гармоническое точности и полноты;
- *смещение* в пикселях положений объектов, выданных системой, относительно истинных положений объектов.

Итоговые показатели качества работы (табл. 1) свидетельствуют, что из обнаруженных системой нарушений 63.4% действительно являются нарушениями и что система обнаруживает 47.2% совершенных нарушений.

Таблица 1: Качество работы после каждого этапа обработки данных интеллектуальной системы видеонаблюдения

Этап обработки		Точность	Полнота	F–мера	Смещение
Детектор активности	автомобили	0.145	0.600	0.233	34.70
	пешеходы	0.042	0.649	0.079	59.70
Детектор объектов	автомобили	0.751	0.491	0.594	14.79
	пешеходы	0.984	0.105	0.190	7.51
Трекер объектов	автомобили	0.606	0.598	0.602	21.77
	пешеходы	0.812	0.488	0.610	11.76
Анализ траекторий («красные»		0.634	0.472	0.542	—

Анализ производительности выявил наиболее времязатратные операции, выполняемые при функционировании системы видеонаблюдения. В таблице 2 представлено время работы отдельных операций на отрезке видеопоследовательности длиною в 5000 кадров, выполняемых при функционировании представляемой системы видеонаблюдения, в секундах и в процентах от общего времени работы системы. Эксперименты проводились на процессоре Inter Core i5 2.67 GHz в однопоточном режиме. Итоговая скорость работы спроектированной системы составляет около 11 кадров в секунду.

Исследование устойчивости системы к аддитивным шумам с амплитудой, подчиненной распределению Гаусса с нулевым средним и различными значениями дисперсии *var*, и шумам типа «соль и перец» при различных значениях доли искаженных пикселей *density* (табл. 3, рис. 1) показало, что такие шумы приводят к значительному падению точности и полноты определения положений объектов, что связано с искажением карты градиентов кадра, которая используется при работе детектора и трекера объектов. Измерялась точность (P), полнота (R) и F-мера. Применение соответствующих фильтров способно восстановить работу, но с меньшими показателями качества, так как исходная карта краев все-таки искажается.

Таблица 2: Время выполнения отдельных операций, выполняемых системой видеонаблюдения, в миллисекундах на один кадр и в процентах от общего времени работы при обработке видеопоследовательности

Операция	Время, мс/кадр	Доля, %
Чтение кадров	2.36	2.6
Масштабирование кадров	8.7	9.63
Прочая предобработка кадров (*rgb2gray, im2double*)	8.55	9.41
Трекинг ранее обнаруженных объектов	3.99	4.42
Обновление модели фона	2.63	2.91
Построение маски активных пикселей	1.97	2.18
Подавление активности под распознанными объектами	0.21	0.24
Морфологические операции над маской активности	12.38	13.69
Выделение прямоугольников вокруг активных областей	18.99	21.01

Объединение перекрывающихся прямоугольников	0.39	0.43
Обнаружение новых объектов в прямоугольниках	29.83	33.00
Объединение списка целей (от детектора и трекера)	0.014	0.002
Удаление целей над неактивными областями кадра	0.23	0.25
Анализ траекторий объектов	0.13	0.15
Всего	*90.39*	*100*

Таблица 3: Влияние на работу системы шума типа «соль и перец» при разных значениях доли искаженных пикселей (*density*) и результат применения медианного фильтра

Данные	**Фильтр**	**Без шума**			**Шум** *density*=0.01			**Шум** *density*=0.05		
		P	R	F	P	R	F	P	R	F
Видео 1	с фильтром	0.98	0.95	0.96	0.97	0.95	0.96	0.94	0.95	0.94
	без фильтра	0.91	0.96	0.93	0.84	0.44	0.57	не работает		
Видео 2	с фильтром	0.92	0.70	0.80	0.93	0.73	0.82	0.91	0.71	0.80
	без фильтра	0.93	0.74	0.82	0.95	0.24	0.38	не работает		

а) б) в)

Рисунок 1 – Применение к кадру видео (а) шума типа «соль и перец» с долей искаженных пикселей *density*=0.01 (б) и *density*=0.05 (в).

Внесение изменений изображения, имитирующих туман (рис. 2) или дождь, вызывает повышение точности, но чрезмерно уменьшает полноту (табл. 5), так как остаются только наиболее надежные ответы классификатора. Неравномерное освещение (рис. 3) разнородно искажает карту краев: где-то усиливаются истинные края объектов, а где-то — шумовые края, поэтому такие изменения бессистемно влияют на точность (табл. 5). Изменение модели объекта при перемещении между областями сцены с разной освещенностью увеличивает число потерь объектов трекером, что уменьшает полноту.

Таблица 4: Сравнение качества работы системы при различных условиях, имитирующих туман

Данные	**Без тумана**			**Туман 1**			**Туман 2**			**Туман 3**		
	P	R	F	P	R	F	P	R	F	P	R	F
Видео 1	0.91	0.96	0.93	0.94	0.91	0.93	1.00	0.45	0.62	не работает		
Видео 2	0.93	0.74	0.82	0.92	0.71	0.80	0.98	0.40	0.57	не работает		

Туман 1 Туман 2 Туман 3

Рисунок 2 – Демонстрация тестовых погодных условий в виде тумана.


Освещение 1


Освещение 2


Освещение 3

Рисунок 3 – Демонстрация тестовых погодных условий в виде неравномерного освещения

Таблица 5: Сравнение качества работы системы при различных условиях, имитирующих неравномерное освещение

Данные	Равномерное			Освещение 1			Освещение 2			Освещение 3		
	P	R	F	P	R	F	P	R	F	P	R	F
Видео 1	0.91	0.96	0.93	0.98	0.78	0.87	0.95	0.59	0.72	0.96	0.35	0.51
Видео 2	0.93	0.74	0.82	0.90	0.59	0.72	0.91	0.48	0.63	0.87	0.11	0.20

В данной статье были представлены результаты экспериментального исследования интеллектуальной системы видеонаблюдения за нерегулируемыми пешеходными переходами. Было оценено качество работы, производительность и устойчивости к неблагоприятным условиям съемки, способным осложнить работы системы. Точность обнаружения нарушений составляет около 63%, а полнота — 47 %. Скорость работы на процессоре с тактовой частой 2.67 GHz при использовании одного потока медленнее скорости работы в режиме реального времени в два раза, однако экспериментальное исследование показало существенный потенциал для ускорения работы представленной системы видеонаблюдения. Слабая устойчивость к неблагоприятным погодным условиям показала необходимость дополнения системы специальными алгоритмами для восстановления работы системы.

Экспериментальное исследование характеристик системы видеонаблюдения показало приемлемое качество и скорость работы для ее внедрения в реальную инфраструктуру дорожного наблюдения.

Список литературы

1. Колосовский М.А. Модельно–алгоритмическое обеспечение интеллектуальной системы видеонаблюдения за нерегулируемыми пешеходными переходами: дис. … канд. техн. наук / АлтГТУ. Барнаул, 2015 – 122 с.

АЛГОРИТМ АНАЛИЗА ТРАЕКТОРИЙ УЧАСТНИКОВ ДВИЖЕНИЯ ДЛЯ СИСТЕМЫ ВИДЕОНАБЛЮДЕНИЯ ЗА НЕРЕГУЛИРУЕМЫМИ ПЕШЕХОДНЫМИ ПЕРЕХОДАМИ

Колосовский М.А. – к.т.н., н.с., Крючкова Е.Н. – к.ф.-м.н., профессор
Алтайский государственный технический университет (г. Барнаул)

В этой статье дается описание подсистемы анализа траекторий участников дорожного движения в рамках интеллектуальной системы видеонаблюдения за нерегулируемыми пешеходными переходами [1]. Задача системы видеонаблюдения заключается в выявлении нарушений на нерегулируемом переходе, главным образом, фиксирование трудно формализуемого нарушения непредоставления преимущества в движении пешеходу на нерегулируемом переходе. Россия стабильно занимает одно из первых мест по числу

погибших и пострадавших в дорожно-транспортных происшествиях, что делает развитие технологий контроля безопасности дорожного движения особенно актуальным. Основной сложностью разработки системы видеонаблюдения за нерегулируемыми переходами является формализация задачи анализа траекторий, которая решается представленной в этой статье подсистемой.

При видеонаблюдении за нерегулируемыми пешеходными переходами с целью обеспечения безопасности дорожного движения и выявления нарушителей представляют интерес ряд ситуаций, среди которых:

1. Пересечение пешеходом проезжей части в неположенном месте.
2. Парковка перед переходом.
3. Длительная остановка на пешеходном переходе (например, во время затора).
4. Движение задним ходом на переходе.
5. Наезд на пешехода.
6. Непредоставление преимущества в движении пешеходу.

Пункты кроме первого связаны с правонарушением со стороны водителей *транспортных средств* (ТС) и подразумевают фиксирование номера ТС с целью привлечения к ответственности. При наличии данных о положении участников движения обнаружение первых пяти ситуаций не представляет сложности. Обнаружение же факта непредоставления преимущества пешеходу гораздо труднее формализовать.

Отметим, что относительно правил дорожного движения на нерегулируемых пешеходных переходах существует ряд споров и неоднозначностей, обсуждение которых оставляется за рамками данной статьи. Однако заметим, что модель обнаружения нарушений спроектирована нами так, что при помощи параметров модели имеется возможность устанавливать ту или иную трактовку правил.

Случаи непредоставления преимущества пешеходу можно разделить на два класса ситуаций, которые будут соответствовать двум состояниям модели, в которых фиксируется нарушение ПДД:

- длительное ожидание пешеходов на тротуаре;
- пересечение водителем перехода, вынуждающее пешехода, пересекающего проезжую часть, изменить скорость или направление движения.

Ввиду того, что факт нарушения водителем зависит не только от наличия пешеходов на переходе, но и от того, прошел ли пешеход(ы) полосу движения, где расположено ТС, состояние модели может быть различным для разных полос движения (рис. 1). Поэтому будем считать, что имеется отдельная модель для каждой полосы движения. Для формализации модели нарушения непредоставления преимущества пешеходу введем следующие обозначения:

- d_k — координата k-го пешехода относительно середины полосы движения: до пересечения $d_k < 0$, после пересечения $d_k > 0$ (рис. 1). Далее для простоты можем опускать индекс k, предполагая, что выражение справедливо для всех k, как, например, на рис. 3;
- S — координата «начала» области перехода R относительно середины полосы движения (рис. 1);
- S1 — координата начала «первого» тротуара относительно середины полосы движения (рис. 1);
- E — координата «конца» области перехода R относительно середины полосы движения (рис. 1);
- $D > 0$ — минимальное расстояние, которое должен пройти пешеход после пересечения середины полосы, чтобы продолжение движения ТС по этой полосе было для этого пешехода безопасным (рис. 1);

- G > 0 — минимальное расстояние перед серединой полосы движения, после преодоления которого пешеходом движение по этой полосе считается небезопасным (предполагается, что G > D, см. рис. 1);
- R — отмеченная на кадре область перехода на проезжей части (рис. 2);
- P1, P2 — отмеченные на кадре области тротуара, прилегающего к области перехода R (рис. 2);
- R = |{di | S ≤ di ≤ min(E, D)}| — количество пешеходов, находящихся в области перехода R, которые находятся на данной полосе движения или будут ее пересекать (рис. 2);
- P = |{di | S1 ≤ di < S}| — суммарное количество пешеходов, находящихся в областях P1 и P2, которые двигаются в сторону данной полосы движения (рис. 2);
- T1 — время ожидания пешеходов на тротуаре;
- K1 — максимальное время ожидания пешеходов на тротуаре, по истечении которого проезжающим ТС будет засчитываться нарушение ПДД;
- T2 — время ожидания водителей перед переходом;
- K2 — максимальное время ожидания водителей перед переходом, по истечении которого водитель может продолжить движение, если движение безопасно для пешеходов (т. е. ∀k dk ∈/ [−G, D]), даже если есть ожидающие или пересекающие переход пешеходы (т. е. P + R > 0)

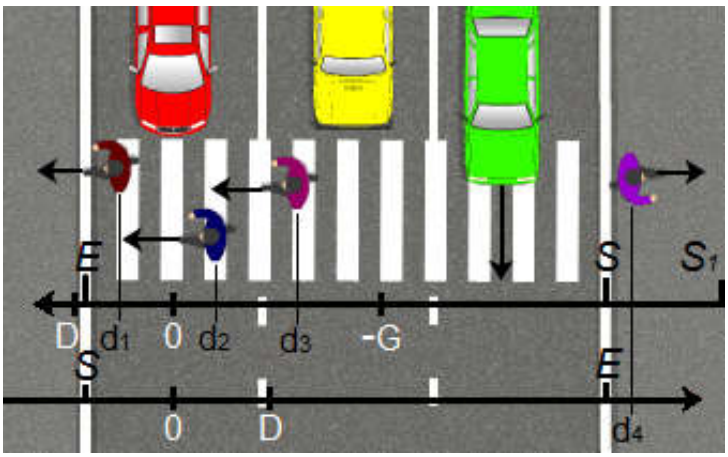


Рисунок 1 – Факт нарушения зависит от того, прошли ли пешеходы соответствующую полосу движения: автомобилю на правой полосе уже можно начать движение, а двум другим еще нет. Также дана иллюстрация определений переменных D, G, d_k, S, S_1, E для левой полосы движения.

Модель нарушения непредоставления преимущества в движении пешеходу на нерегулируемом переходе представлена на рис. 3 в виде диаграммы состояний UML.

Как уже отмечалось, вопрос об оценке правомерности поведения водителей и пешеходов в некоторых ситуациях может быть спорным. Представленная модель позволяет при помощи настройки параметров задавать ту или иную интерпретацию подобных ситуаций. Например, отметим две ситуации, особо часто вызывающие споры, описание которых дано на рис. 4.

Для первой ситуации при помощи параметра G можно задать, как далеко должны находиться пешеходы для безопасного проезда ТС, либо задать G настолько большим, что пешеход, ступивший на область перехода, запрещает любое движение ТС.

Для второй ситуации параметр T_2 задает, как долго ТС должны пропускать пешеходов. Заданием очень большого T_2 можно запретить движение ТС, пока весь поток пешеходов не пересечет проезжую часть.

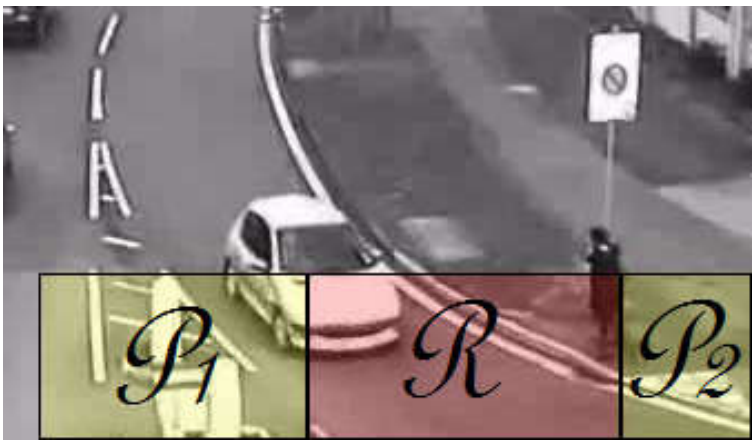


Рисунок 2 – Пример размеченных зон проезжей части R и прилегающих тротуаров P_1 и P_2.

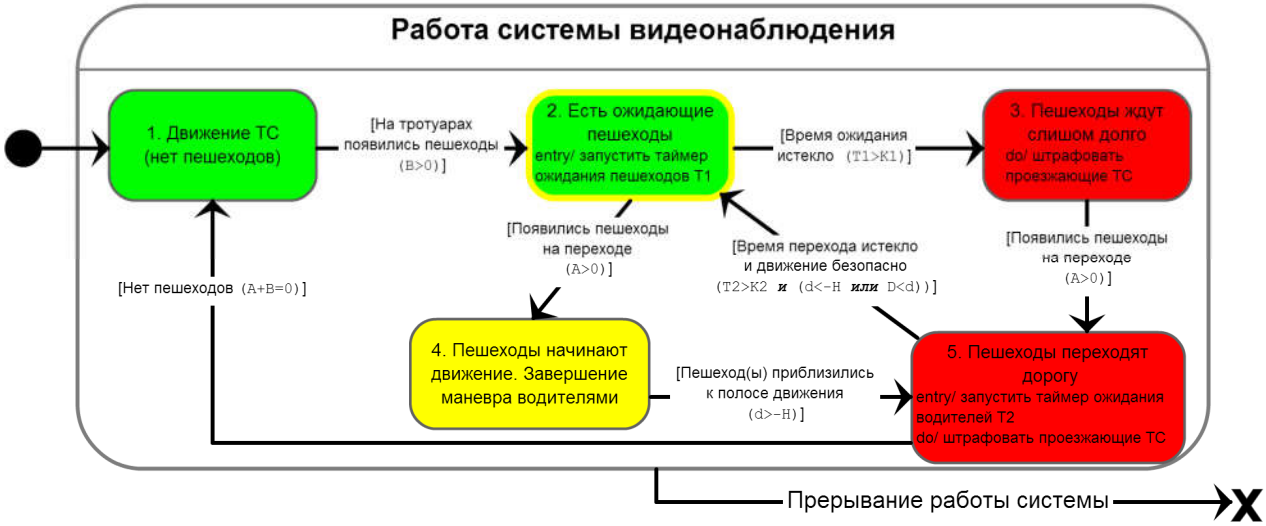
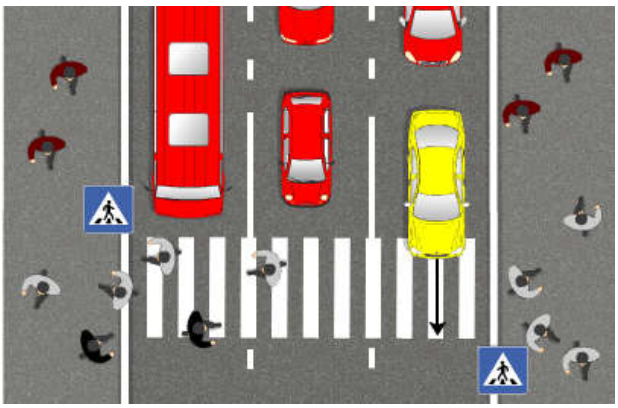
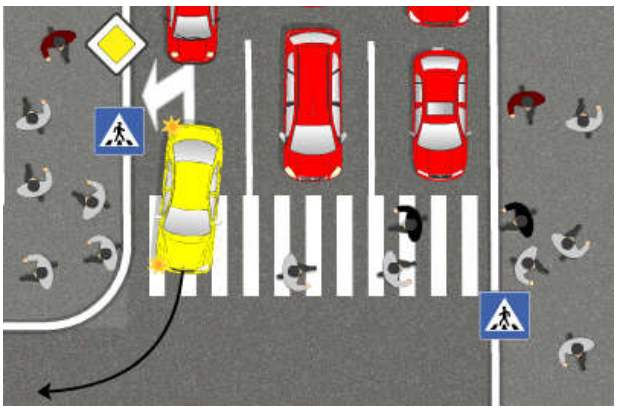


Рисунок 3 – Модель анализа траекторий для выявления нарушения «непредоставление преимущества в движении пешеходу на нерегулируемом переходе».



а) движение ТС одновременно с началом пересечения перехода пешеходами, но не представляющее опасности для пешеходов;



б) начало движения ТС после длительного ожидания при наличии идущих по переходу людей, но не представляющее для них опасность.

Рисунок 4 – Особые ситуации (а) и (б) в анализе правомерности движения ТС на нерегулируемом переходе

Список литературы

1. Колосовский М.А. Модельно–алгоритмическое обеспечение интеллектуальной системы видеонаблюдения за нерегулируемыми пешеходными переходами: дис. … канд. техн. наук / АлтГТУ. Барнаул, 2015. – 122 с.

СИСТЕМА УПРАВЛЕНИЯ УМНЫМ ДОМОМ НА ОСНОВЕ ВЕРБАЛЬНОГО И НЕВЕРБАЛЬНОГО ОБЩЕНИЯ

Колесников Н.С. – магистрант, Старолетов С.М. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Актуальность

Согласно статистике развитие «Умного дома» набирает обороты. По оценкам аналитиков, рынок умного дома активно развивается. К 2020 году общий объём мирового рынка достигнет $51.77 млрд. В период с 2013 по 2020 года среднегодовые темпы роста рынка будут на уровне 17.74%. Объёмы российского рынка значительно скромнее. В 2012 году объём рынка у нас в стране превысил 56 млн евро или 2,3 млрд рублей. В 2013 году по предварительным оценкам рынок вырос на 30% - до 65 млн евро или почти 3 млрд рублей. К 2017 году его общий объём может достигнуть 176 млн евро или 7,9 млрд рублей. Таким образом, в скором будущем умный дом будет таким же бытовым элементом, как пульт от телевизора.

Постановка задачи

Под понятием «Умный дом» считают жилое помещение, в котором встроена система, обеспечивающее безопасность и комфорт[1]. Эта система должна понимать и реагировать на бытовые ситуации, происходящие в доме. Необходимо проанализировать и разработать систему «умный дом». Основной особенностью этой системы будет простота настройки для обычного пользователя. Система должна работать без доступа к интернету. Система будет включать удобный интерфейс управления устройствами. Будет распознавать вербальные команды и жесты. Главной особенностью системы будет являться цена умного дома. Необходимо проанализировать и выбрать бюджетный вариант контроллера системы, а также всей системы в целом.

Основные параметры системы «Умный дом»

Энергосберегающая система управления освещением в многоэтажных домах (подъезды, автостоянки, придомовые территории, подвалы, чердаки) позволит снизить количество потребляемой электроэнергии в 10-15 раз. Эффективная мера энергосбережения - централизация управления освещением с использованием специально разработанных графиков включения и выключения света [2].

Управление освещением - одна из самых важных задач в доме. Благодаря интеллектуальному программированию можно сэкономить электроэнергию и срок эксплуатации ламп. Отпадает необходимость искать выключатели света в темноте, а так же выключать свет при выходе из комнаты.

Система управления климатом в помещении дает возможность корректировать уровень температуры, влажности, величину притока свежего воздуха индивидуально для каждого помещения, управлять работой системы фильтрации воздуха, создавать индивидуальную климатическую систему для каждого члена семьи [3].

Прорыв труб водоснабжения является очень неприятным событием в связи с порчей не только своего, но и соседского имущества. Обнаружить и предотвратить протечку воды так же поможет умный дом [5]. Контролируемыми зонами являются санузлы и кухня, т.е. те помещения, где проходят трубы водоснабжения.

Контроллеры

Arduino просто идеально подходит для электроники проектов и прототипирования. Вы можете легко подключить к плате несколько светодиодов, датчики, двигатели. Для программирования Arduino нужно их программное обеспечение, которые можно скачать бесплатно. В принципе с этим программным обеспечением можно загрузить исходный код непосредственно в Arduino через USB [6].

Raspberry Pi является мини-компьютером. Для работы плата нуждается в операционной системе. Все хранения обеспечивается с SD-карты. Имеет графику и имеет выход HDMI. Возможно подключить клавиатуру и монитор, загрузить Linux, и работать как на компьютере [7].

BeagleBoard - плата похожа на Raspberry Pi, но более мощная. Процессор от компании Texas Instruments серии Sitara AM3359AZCZ100 с тактовой частотой в 1ГГц содержит в своем составе вычислительное ядро Cortex-A8 и графический ускоритель SGX530 . На плате установлено 512 Мб оперативной памяти DDR3L , Flash память eMMC объемом в 2 ГБ , на которой установлена операционная система Angstrom Linux[8].

Общие характеристики всех плат представлены на рисунке 1.

Name	Arduino Uno	Raspberry Pi	BeagleBone
Model Tested	R3	Model B	Rev A5
Price	$29.95	$35	$89
Size	2.95"x2.10"	3.37"x2.125"	3.4"x2.1"
Processor	ATMega 328	ARM11	ARM Cortex-A8
Clock Speed	16MHz	700MHz	700MHz
RAM	2KB	256MB	256MB
Flash	32KB	(SD Card)	4GB(microSD)
EEPROM	1KB		
Input Voltage	7-12v	5v	5v
Min Power	42mA (.3W)	700mA (3.5W)	170mA (.85W)
Digital GPIO	14	8	66
Analog Input	6 10-bit	N/A	7 12-bit
PWM	6		8
TWI/I2C	2	1	2
SPI	1	1	1
UART	1	1	5
Dev IDE	Arduino Tool	IDLE, Scratch, Squeak/Linux	Python, Scratch, Squeak, Cloud9/Linux
Ethernet	N/A	10/100	10/100
USB Master	N/A	2 USB 2.0	1 USB 2.0
Video Out	N/A	HDMI, Composite	N/A
Audio Output	N/A	HDMI, Analog	Analog

Рисунок 1 – Характеристики плат

Системы распознавания голоса

CMU Sphinx – также для краткости просто называемая Sphinx, главным образом была написана группой разработчиков систем распознавания речи университета Карнеги Меллон[9]. Она включает в себя серию распознавателей речи (Sphinx 2-4) и тренировщика акустической модели (Sphinx train).

Julius – это высокопроизводительный распознаватель непрерывной речи с большим словарем (large vocabulary continuous speech recognition), декодер программного обеспечения для исследования в области связанной речи и разработки.

RWTH ASR включает в себя инструментарий для разработки акустических моделей и декодеры а также компоненты для адаптации речи спикера, адаптивные системы обучения речи спикера, неконтролируемые системы обучения, дифференциальные системы обучения и решетчетые словообразные формы обработки [4].

VoxForge – это свободный речевой корпус и языковая акустическая модель, представленная на Open Source хранилище данных[10]. VoxForge был собран как хранилище транскрипций речи, имеющий свободный GPL корпус для использования с речевыми движками, которые имеют открытый исходный код.

Моделирование

Для построения модели был выбран контроллер Arduino UNO. Он наиболее приемлем по цене и не уступает по возможностям другим рассмотренным контроллерам.

По рассмотренным системам распознавания речи была выбрана CMU Sphinx. Так как в ней наиболее продумана общая архитектура с распознаванием русого языка.

В ходе исследования работы контроллера было написано несколько программ:

Мигание встроенной лампочки с интервалом в 1 секунду (1 секунду горит, 1 не горит).

Рисунок 2 – Горит, не горит

Включение-выключение лампочки по нажатию кнопки.

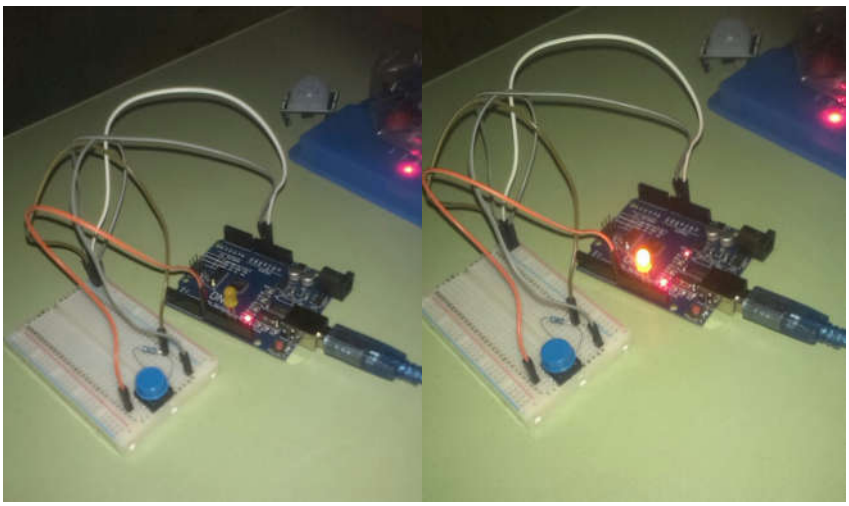


Рисунок 3 – Кнопка выключатель

Ночник. Работа диодной лампочки со светочувствительным детектором. (Лампочка загорается только если в комнате становится темно, чем темнее в комнате, тем ярче загорается лампочка)

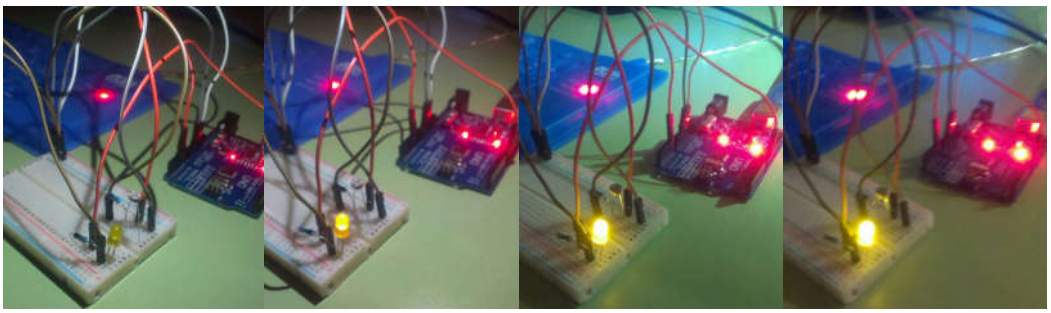


Рисунок 4 – Ночник

Создание локального сервера который может включать-выключать лампочки из браузера по WiFi.

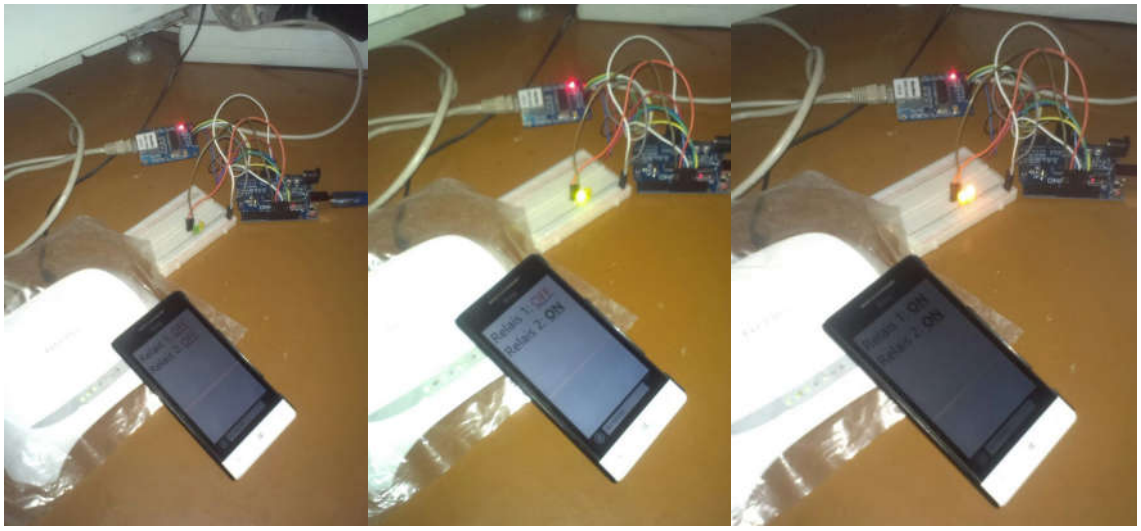


Рисунок 5 – Доступ по локальному серверу

Использование детектора движения и реле, для лампы с напряжением в 220 вольт.

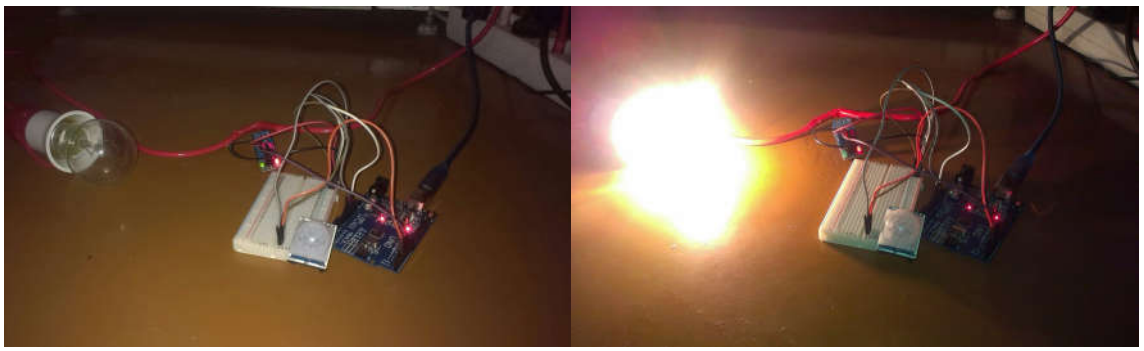


Рисунок 6 – Датчик движения и реле

Была реализована программа распознавания речи на русском языке, которая распознавала цифры.

Пример грамматики:

<greeting> = (ок|окей)(компьютер|калькулятор);

<n1>=(ноль|один|два|три|четыре|пять|шесть|семь|восемь|девять);

<n2>=(десять|одинадцать|двенадцать|тринадцать|четырнадцать|пятнадцать|шестнадцать|семнадцать|восемнадцать|девятнадцать)|(двадцать|тридцать|сорок|пятьдесят|шестьдесят|семьдесят|восемьдесят|девяносто)[<n1>];
<n3> = (сто|двести|триста|четыреста|пятьсот|шестьсот|девятьсот)[<n2>|<n1>];
<n4> = [<n1>|<n2>|<n3>](тысяча|тысяч|тысячи)[<n3>|<n2>|<n1>];
<number> = <n4>|<n3>|<n2>|<n1>;
<expression> = <number>;
public <query> = <greeting><expression>;

Заключение

Для построения умного дома были рассмотрены несколько контроллеров и выбран Arduino. Так как он удовлетворяет по многим критериям. Были реализованы несколько программ, которые управляли устройствами. Была реализована система распознавания чисел на русском языке с использованием CMU Sphinx.

В дальнейшем необходимо проанализировать систему голосового ответа. Необходимо проанализировать системы управления жестами. Необходимо разработать систему голосового обучения. Она должна использовать систему CMU Sphinx. Необходимо разработать интерфейс, который позволял пользователю просто записывать или даже диктовать команды, а система автоматически строила бы Марковскую модель. Необходимо реализовать систему генерирования устройств умного дома. Система должна также иметь дружелюбный интерфейс для обычного пользователя. В ней можно будет конструировать свой умный дом. Она способна будет генерировать рабочую среду, с возможностью протестировать ее. Эта система будет автоматически генерировать код для контроллера Arduino, вплоть до построения веб-сервиса со всеми устройствами и возможностью управления этими устройствами по Wi-Fi.

Список литературы

1. Сопер М.Э. Практические советы и решения по созданию « Умного дома » / Сопер М. Э. – М.: НТ Пресс, 2007. – 432 с.
2. Тесля Е.А. «Умный дом» своими руками. Строим интеллектуальную цифровую систему в своей квартире / Тесля Е.А. – Санкт Петербург, 2008. – 224 с.
3. Харке В.Н. «Умный дом. Объединение в сеть бытовой техники и систем коммуникаций в жилищном строительстве» / Харке В.Н. – М.: Техносфера, 2006. – 292 с.
4. Элсенпитер Т.Р., Велт Дж. «Умный Дом строим сами» / Элсенпитер Т. Р., Велт Дж / КУДИЦ-ОБРАЗ. 2005. – 384 с.
5. Гололобов В.Н. «Умный дом» своими руками. / Гололобов В.Н. – М.: НТ Пресс, 2007. – 416 с.
6. Arduino WiFi Power Switch & Energy Monitoring Device [Электронный ресурс]. Режим доступа: https://www.openhomeautomation.net/arduino-wifi-switch/, свободный.
7. Arduino [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Arduino, свободный.
8. Делаем управление «Умным домом» через интернет за пару минут [Электронный ресурс]. Режим доступа: https://geektimes.ru/post/258504/, свободный.
9. Сверхбыстрое распознавание речи без серверов на реальном [Электронный ресурс]. Режим доступа: https://habrahabr.ru/post/237589/, свободный.
10. Sourceforge [Электронный ресурс]. Режим доступа: https://sourceforge.net/, свободный.

СИСТЕМА РАСПОЗНАВАНИЯ МАРКИ АВТОМОБИЛЯ ПО ФОТОГРАФИЯМ С КАМЕР ФОТОФИКСАЦИИ

Лаптев М.А. – магистрант, Старолетов С.М. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Актуальность

Дорожно-транспортные происшествия являются одной из важнейших мировых угроз здоровью и жизни людей. Проблема усугубляется и тем, что пострадавшие в авариях — как правило, молодые и здоровые (до аварии) люди. По данным Всемирной организации здравоохранения, в мире ежегодно в дорожных авариях погибают 1,2 млн человек и около 50 млн получают травмы. Ежегодно только на российских дорогах погибает более 27000 человек. Основной причиной дорожно-транспортных происшествий является несоблюдение требований ПДД. Поэтому поддержание правопорядка на дорогах является первоочередной задачей для современного общества.

Одним из инструментов ГИБДД является использование камер фотовидеофиксации для выявления нарушителей скоростного режима. Принцип работы этих камер имеет слабое звено.

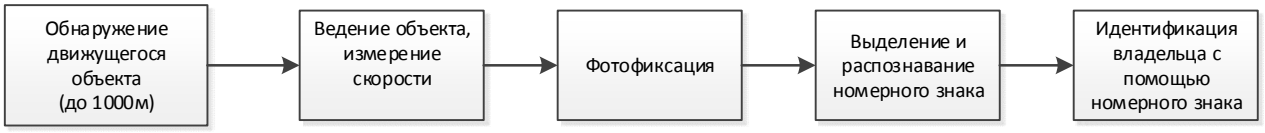


Рисунок 1 – Алгоритм работы комплексов фотовидеофиксации

Идентификация нарушителя производится только по номерному знаку транспортного средства. Вся остальная информация на изображении отбрасывается. Эту особенность используют некоторые недобросовестные водители, которые передвигаются на автомобилях с заведомо подложными номерами. Таким образом, злоумышленники избегают наказания за нарушения ПДД, а также скрываются от правоохранительных органов, находясь в розыске.

Постановка задачи

Необходимо разработать систему распознавания марки автомобилей по фотографиям с камер фотофиксации с использованием интегрального алгоритма идентификации.

Цели системы

1. Выявление автомобилей с подложными номерами.
2. Выявления ошибок в базе данных регистрации транспортных средств
3. Повышение точности идентификации транспортных средств (сокращение процента ошибок распознавания за счёт дополнительной проверки марки автомобиля).

Текущее положение

На данный момент составлен протокол о намерениях с региональным отделом информационного обеспечения ГИБДД ГУ МВД России по Алтайскому краю, произведён анализ существующих комплексов фотовидеофиксации, осуществлена подборка тестовых фотографий и ведётся разработка интегрального алгоритма идентификации марки автомобилей.

Одной из составных частей интегрального алгоритма был выбран каскад Хаара по причине своего быстродействия. В процессе работы было установлено, что каскады Хаара очень чувствительны к обучающей выборке и размеру искомой области. Так первая версия программы, которая искала автомобили марки «Лада Приора», несмотря на большое количество обучающих примеров, работала крайне нестабильно и определяла искомый автомобиль иногда даже в колесе автомобиля другой марки.

Рисунок 2 – Результаты работы первой версии программы

Причина этого в том, что для обучения использовались изображения целого автомобиля, содержащие много лишней информации. Кроме того, в качестве искомой области использовался квадрат, и, таким образом, большую часть искомой области занимал капот и лобовое стекло автомобиля. А эти части не содержат ключевых особенностей, которые помогли бы в идентификации.

Затем была предпринята модификация обучающей выборки: в качестве позитивных примеров подавались изображения исключительно лицевой часть автомобилей.

Рисунок 3 – Обучающие примеры для каскадов Хаара

А в качестве искомой области был определён прямоугольник с соотношением сторон 1:2. Для обучения была использована небольшая выборка: 150 позитивных примеров и 500 отрицательных, что примерно в 3 раза меньше чем в первом случае. Тем не менее, работа алгоритма стала на порядок более стабильной.

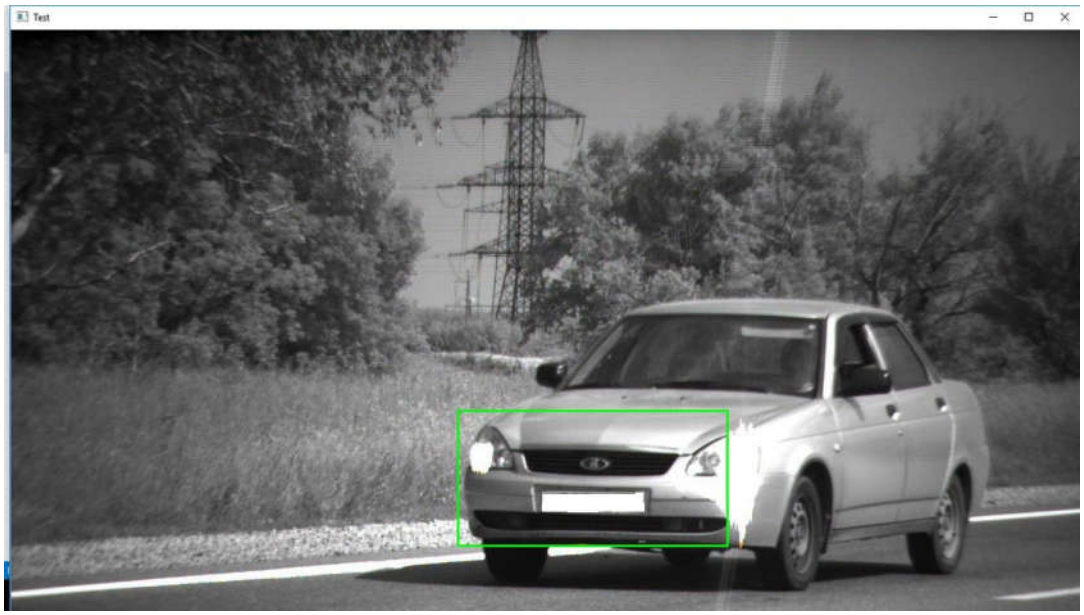


Рисунок 4 – Пример успешно распознанного автомобиля Lada Priora

Для количественной оценки работы алгоритма было проведено исследование с попытками распознать «Ладу Приору» на автомобилях разных марок.

Результаты исследования точности распознавания приведены в таблице 1.

Таблица 1

Марка авто	**Количество примеров**	**Количество срабатываний**	**Процент срабатываний**
Лада Приора	500	334	66,8
Хонда Фит	500	161	32,2
Форд Фокус	500	155	31
Лада Гранта	500	139	27,8
Рено Логан	500	124	24,8
Фольксваген Пассат	500	96	19,2
Хёндай Солярис	500	83	16,6
Ниссан Санни	500	66	13,2
Лада 2110	500	58	11,6
Мазда 3	500	54	10,8
УАЗ	500	34	6,8
Ваз 2107	500	16	3,2

Планы

Планируется дальнейшая работа по модификации обучающей выборки и параметров каскада Хаара для повышения точности распознавания. Кроме того, будет проведено исследование по применимости каскадов Хаара для распознавания типа транспортного средства, а также типа кузова. Также планируется использование других алгоритмов и написание своего детектора и классификатора.

Список литературы

1. Viola and Jones, «Rapid object detection using a boosted cascade of simple features», Computer Vision and Pattern Recognition, 2001

2. Lienhart R. and Maydt J. «An extended set of Haar-like features for rapid object detection», ICIP02, pp. I: 900—903, 2002
3. Дэвид Форсайт, Жан Понс. Компьютерное зрение. Современный подход. — М.: «Вильямс», 2004. — 928 с.
4. Bradsky G., Kaehler A. Learning OpenCV — O'Reilly, 2008.

УЧЕТ НАЛОГОВЫХ И НЕНАЛОГОВЫХ ПОСТУПЛЕНИЙ В БЮДЖЕТ АЛТАЙСКОГО КРАЯ ПРИ ПОМОЩИ OLAP ТЕХНОЛОГИИ АНАЛИЗА ДАННЫХ

Ложкина Н.С. – студент, Старолетов С.М. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

Актуальность

Вся наша жизнь состоит из информации и на протяжении всего времени перед человечеством стоит глобальная задача – хранение данных. Сегодня хранение данных помогает справиться с огромным потоком информации, с которой раньше человеку приходилось справляться вручную. Теперь благодаря современным базам данных информация может обновляться десятками тысяч раз за один день. В современном мире хранить какую-либо информацию в базе данных это обычное дело, поэтому теперь стоит другая задача – как качественно и быстро можно обработать эту информацию.

Постановка задачи

В комитете Администрации Алтайского края по финансам возникла необходимость решения задачи точного учета налоговых и не налоговых поступлений в бюджет. Вся документация – это большие наборы данных (около 10000 записей в день), которые приходят в виде xls файлов следующей структуры:

- дата регистрации;
- сумма;
- ИНН получателя;
- КПП получателя (Код причины постановки на учет);
- наименование получателя;
- КБК (код бюджетной классификации – специальный цифровой код, используемый для группировки статей государственного бюджета)
- ОКАТО (Общероссийский классификатор объектов административно-территориального деления);
- назначение (налог);
- отдел в Администрации.

В Администрации такая задача реализована на Visual Basic. На вход подаются три вида xls файлов: возврат, приход, уточнения. Для каждого вида таких файлов имеется свой макрос загрузки данных в базу. Далее они обрабатываются, и затем все редактированные данные переходят в итоговый xls файл путем обработки макросом. Такой способ загрузки и выгрузки не обеспечивает скорости обработки, расширяемости и надежности. Поэтому необходимость модернизации данного алгоритма очевидна. Наиболее перспективным решением для данной задачи является разработка проекта, написанного средствами .net приложения, что позволит повысить скорость импорта и экспорта данных большого объема в базу (рисунок 1). Язык высокого уровня позволит повысить надежность, отказоустойчивость, избежать ошибок на свежих версиях операционных системах. Что не маловажно, такая

реализация будет иметь удобный для пользователя интерфейс и широкие возможности для поддержки и развития проекта.

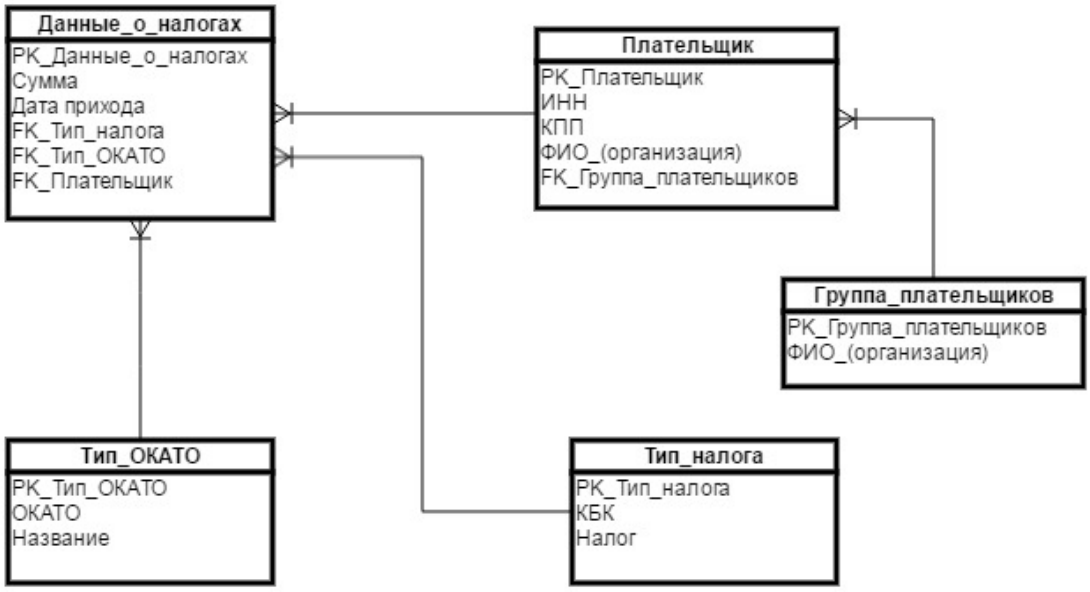


Рисунок 1 – Схема реляционной модели базы данных

Говоря про загрузку данных, следует учесть, что информация в выписках зачастую содержит неточности в части наименования плательщика, которую нужно исправлять на этапе импорта (путем сверки с ЕГРЮЛ и ЕГРИП). Все поступающие сведения анализируются в различных разрезах: период, вид дохода, муниципальное образование, налогоплательщик с целью агрегирования суммы платежей или возвратов.

Структура существующего итогового файла Администрации становится громоздкой из-за постоянных обновлений (поступающих заявок). Для загрузки данных необходимо подключение к базе, затем вся информация о налогоплательщике (ИНН, наименование плательщика) собирается в одну колонку таблицы, а данные о суммах и типах налога, формируемые за каждый отчетный период, отображаются в сводной таблице (рисунок 2). Данный файл позволяет просматривать всю принятую информацию по видам дохода и по заданному периоду, который измеряется или месяцем, или годом, также имеется итоговая сумма всех налогов.

Названия строк	01 Январь	02 Февраль	03 Март
2221054872			
ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТ			
2222034935			
ВИГОРТ ООО			
2224081627			**459,00**
ООО "Алтайский вернисаж"			459,00

Рисунок 2 – Часть итогового файла

Такой анализ данных очень неудобен, так как он не только визуально недружелюбный для понимания, но и технически сложен, потому что нет быстрого доступа к данным, то есть помимо нужной информации есть и лишняя. И, исследуя данную проблему, можно сделать вывод, что для такой задачи применить обычные SQL-выборки нецелесообразно, так как объем данных большой и необходима группировка.

При выборе вариантов решения задачи были рассмотрены различные способы анализа данных в базе и выбрана технология OLAP (**On**Line **A**nalytical **P**rocessing, то есть оперативный анализ данных), потому что один из плюсов такой технологии – это скорость.

Данные, приходящие извне и собираемые в базу, складываются в реляционное хранилище, а оттуда уже идут на обработку. OLAP-структура, созданная из рабочих данных, называется OLAP-куб. Суть такого куба состоит в объединении таблиц базы данных (рисунок 3). При больших вычислениях изменения происходят только для некоторых измерений, для остальных же когда потребуется. Такую технологию можно определить как совокупность средств многомерного анализа данных, накопленных в БД.

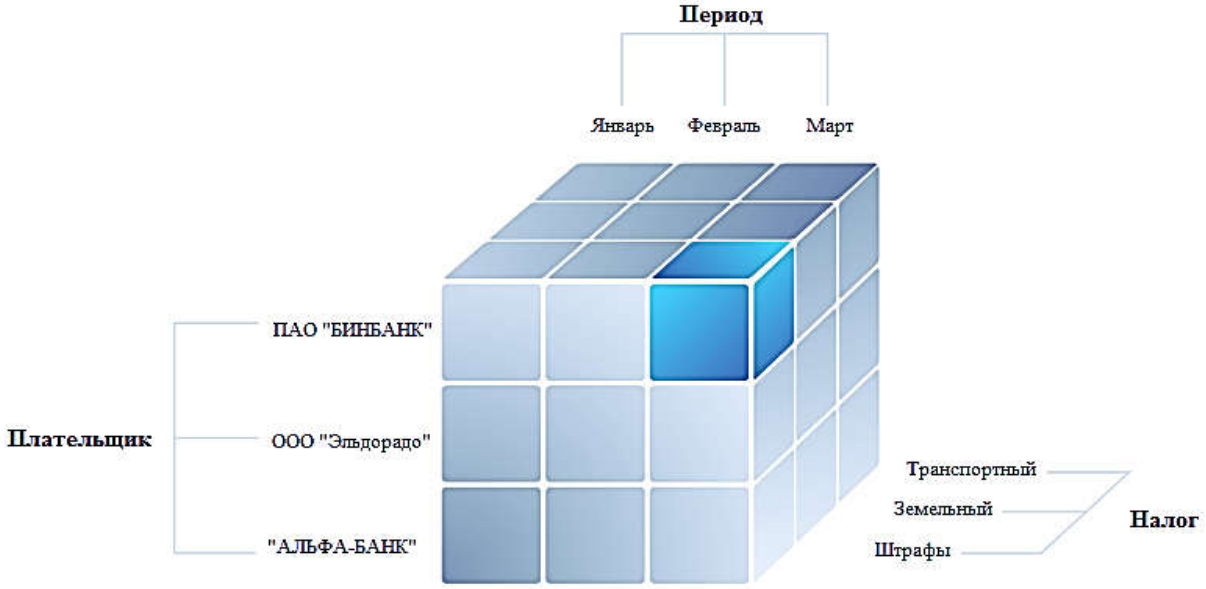


Рисунок 3 – Куб с измерениями: Плательщик, Период, Налог

Подходя к вопросу о разработке данного программного обеспечения, отметим, что импорт данных будет осуществляться также sql-запросами в базу данных MS SQL Server 2012, которая будет обрабатываться службой Analysis Services (SSAS). После прохождения аналитической обработки (OLAP) получим вычисленные данные (рисунок 4). Далее остается только вопрос о визуализации и экспорте итоговых отчетов в файлы.

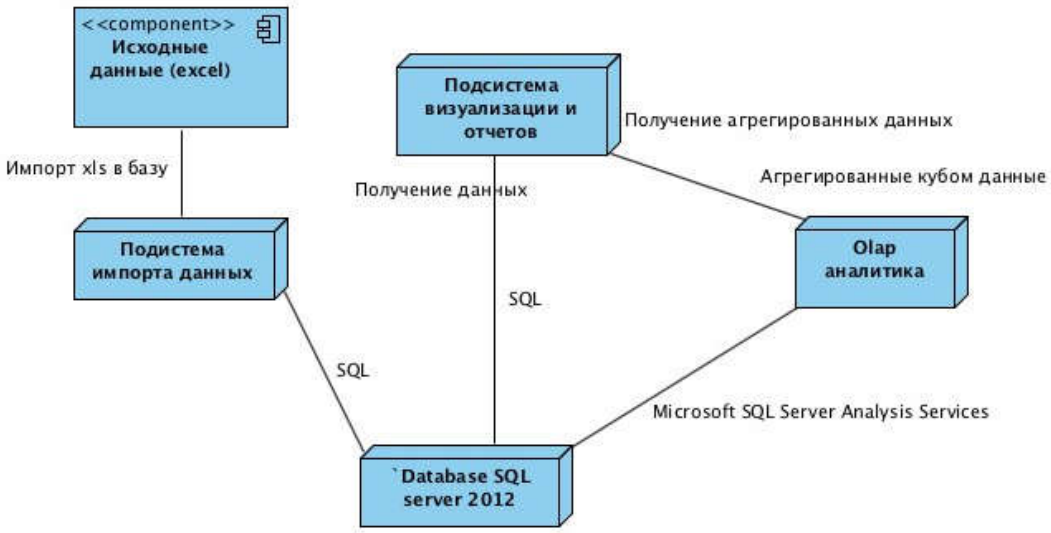


Рисунок 4 – Архитектура разрабатываемой системы

В настоящее время производится реализация ПО, которое разрабатывается по заказу комитета Администрации Алтайского края по финансам. Для разработки используется язык C#, база данных SQL Server 2012.

Список литературы

1. Крёнке Д. Теория и практика построения баз данных. - 8-е изд. - учеб. пособие. - СПб.: Питер, 2003. – 800с.: ил. - (Серия Классика computer science)

АВТОМАТИЧЕСКАЯ ГЕНЕРАЦИЯ ИЛЛЮСТРАТИВНОГО МАТЕРИАЛА ИЗ ТЕКСТА НА ЕСТЕСТВЕННОМ ЯЗЫКЕ ДЛЯ ИСПОЛЬЗОВАНИЯ В ТЕХНОЛОГИЯХ ЭЛЕКТРОННОГО ОБУЧЕНИЯ

Дука С.В. – инженер ЛЭОР, Крайванова В.А. – к.ф.-м.н., доцент
Алтайский государственный технический университет (г. Барнаул)

На сегодняшний день электронное обучение в России активно развивается и занимает достойное место в системе образования страны. В соответствии с ФЗ № 273-ФЗ «Об образовании в Российской Федерации» введена отдельная статья законопроекта (ст. 16), посвященная электронному обучению (ЭО) и дистанционным образовательным технологиям (ДОТ) [1]. Активное использование методов электронного обучения в очном образовании способствовало развитию смешанной модели обучения, то есть, совмещения аудиторного и электронного обучения.

Однако, как в традиционном, так и в электронном обучении главной задачей преподавателя является умение правильно и эффективно донести материал до студента. При этом значение качества материалов возрастает при внедрении образовательных технологий, где взаимодействие между студентом и преподавателем опосредовано компьютером. Как известно, длинные тексты воспринимаются достаточно сложно, поэтому возникает необходимость в визуализации информации. Создание качественных визуализаций – сложный творческий процесс, требующий специфических навыков, которым владеют далеко не все разработчики учебных материалов. В любом случае разработка действительно эффективных обучающих изображений требует больших затрат времени. В связи с этим актуальной является задача автоматическогоперевода из текстового материала в иллюстративный (графическое аннотирование, заготовки презентаций, построение схем, таблиц, диаграмм, создание библиографии, глоссариев).

Для достижения поставленной задачи нами была разработана двухэтапная модель обработки текста:

- парсинг текста на естественном языке и перевод его на формальный язык (в частности, на OWL);
- креолизация формального представления и перевод в иконическую форму.

Креолизованные тексты — это тексты, фактура которых состоит из двух негомогенных частей: вербальной (языковой/речевой) и невербальной (принадлежащей к другим знаковым системам, нежели естественный язык, в частности, иконическим) [2]. Иконический компонент текста может быть представлен иллюстрациями (фотографиями, рисунками), схемами, таблицами, символическими изображениями, формулами и т.д.

Текст на естественном языке представляет собой систему со сложной, неравномерной внутренней структурой, с разной степенью формализованности. Поэтому перед применением алгоритмов креолизации необходимо выделить из текста фрагменты, которые наиболее пригодны для переведения в иконическую форму. Такие фрагменты можно разделить на два вида:

- Фрагменты, имеющие высокий уровень формализованности в исходном тексте. Это перечисления и списки, а также однородные члены предложения, т.е. слова, относящиеся к одной части речи, разделенные между собой пунктуационными

знаками «,» и «;», а также таблицы, тезаурусы и другие подобные элементы. Перевод таких фрагментов в иконическую форму представляет собой сравнительно простую задачу, соответствующую контекстно-свободным грамматикам;

- Слабоструктурированные фрагменты, которые, тем не менее, хорошо соотносятся с известными типами схем и диаграмм. Примерами элементов такого вида являются предложения с глаголами движения, а также глаголами, обозначающими причинно-следственные связи, иконически могут быть представлены в виде схем. Предложения, содержащие глаголы в императивной форме, можно преобразовать в алгоритмы.

Креолизация требует преобразования и сокращения фраз, взятых из текста без существенной потери смысла. При этом часть смысла переходит в другие знаковые системы.

При конвертации вербальной информации в иконическую важно учитывать некоторые особенности:

- Явление омонимии в языке. Омонимия - звуковое совпадение различных языковых единиц, значения которых не связаны друг с другом [3]. Особый интерес представляют собой омоформы (слова, совпадающие в определенной грамматической категории): три (число) – три (форма императива от глагола «тереть»).
- Явление полисемии в языке. Полисемия (многозначность) предполагает наличие у слова не одного, а несколько значений. Например, ядро: 1) внутренняя часть плода в твердой оболочке; 3) центральная часть чего-либо; 4) старинный орудийный снаряд.
- Явление синонимии в языке. Синонимы – слова, тождественные или близкие по своему значению. Например, думать – мыслить – размышлять – раздумывать – подумывать.

Омонимия и полисемия, как правило, могут быть разрешены из контекста. Синонимы требуется учитывать при описании формальной модели иллюстрации. В частности, использовать отдельные глаголы в шаблонах иллюстраций невыгодно, описывать параметры шаблонов лучше гнездами синонимов.

Важно также учитывать возможные опечатки в тексте, в особенности – неверное расположение знаков пунктуации и возможные переносы слов на другую строку, специфику источников текста (восстановление текста, распознанного из изображений, обработка форматов tex и html), наличие слов, словосочетаний и/или предложений на иностранных или формальных языках.

Электронные учебные материалы – это не просто набор разрозненных иллюстраций. Поэтому кроме фрагментов, пригодных для создания креолизованных текстов, необходимо извлекать их формальную структуру: заголовки глав и разделов. Кроме того, необходимо генерировать заголовки для каждого креолизованного элемента.

Проведенный анализ проблем перевода вербальной информации в иконическую, а также предложенные пути организации текста на естественном языке в невербальной форме позволят реализовать программный комплекс для автоматической генерации заготовок иллюстраций различных типов.

Автоматическая генерация иллюстраций повысит экономическую эффективность и снизит трудозатраты на разработку электронных учебных материалов.

Список литературы

1. Федеральный закон Российской Федерации от 29 декабря 2012 г. N 273-ФЗ "Об образовании в Российской Федерации" – Режим доступа: http://rg.ru/2012/12/30/obrazovanie-dok.html (дата обращения: 15.04.16)
2. Сорокин Ю. А., Тарасов Е. Ф. Креолизованные тексты и их коммуникативная функция / Ю. А. Сорокин, Е. Ф. Тарасов // Оптимизация речевого воздействия. — М.: Высшая школа, 1990. – сс. 180–186.

3. Большой энциклопедический словарь. Языкознание. Гл. ред. ... Ярцева В.Н. М.: Большая Российская энциклопедия, 1998. – 344 с.