Министерство образования и науки
Донецкой Народной Республики

ГОУВПО «Донецкий национальный технический университет»

Факультет «Компьютерные науки и технологии»

Кафедра «Программная инженерия»

Программная инженерия: методы и технологии разработки информационно-вычислительных систем

(ПИИВС-2018)

Сборник материалов II Международной научно-практической конференции

(студенческая секция)

г. Донецк
14-15 ноября 2018 года

Донецк – 2018

УДК 004.4
ББК 32.81
П78

Программная инженерия: методы и технологии разработки информационно-вычислительных систем (ПИИВС-2018): сборник научных трудов II научно-практической конференции (студенческая секция), Том 2, 14-15 ноября 2018 г. – Донецк, ГОУВПО «Донецкий национальный технический университет», 2018. – 264с.

Студенческая секция, проводимая в рамках научной конференции ПИИВС-2018, является одним из этапов совместной исследовательской деятельности преподавателей и студентов. Основная задача студенческой секции состояла в укрепление научного и педагогического сотрудничества среди студентов и научных руководителей, развитие у студентов навыков самостоятельной работы, способностей к анализу и обобщению изучаемого материала, умению формировать собственные выводы и заключения, излагать их письменно и в форме публичных выступлений.

Основные направления работы студенческой секции конференции:

- программная инженерия;
- системы автоматизированного проектирования;
- информационные технологии в медиаиндустрии и дизайне
- прикладная математика;
- информационные системы и технологии в технике и бизнесе;
- системный анализ и интеллектуальные информационные системы.

На 4 подсекциях были представлены 53 доклада, в которых рассмотрен широкий спектр научных направлений работ студентов.

Конференция организована Донецким национальным техническим университетом Министерства образования и науки ДНР. В организации конференции приняли участие: Донецкий национальный университет, Министерство связи ДНР, Ульяновский государственный технический университет (г.Ульяновск), Образовательно-научный центр «Кибернетика» ФГБОУ ВО «РЭУ им. Г.В. Плеханова» (г.Москва), Полоцкий государственный университет (Республика Беларусь, г.Полоцк).

Во втором томе сборника научных трудов представлены доклады студентов, которые учатся в бакалавриате и магистратуре высших учебных заведений из Ижевска, Кемерово, Таганрога и Донецка.

УДК 004.4
ББК 32.81

ОГЛАВЛЕНИЕ

СЕКЦИЯ «ПРОГРАММНАЯ ИНЖЕНЕРИЯ»

УДК 004.457

Применение метода Хаффмана для сжатия информации

Алексеев А.А.
Научный руководитель: Н.В. Латыпова
Удмуртский государственный университет
alekanton95@mail.ru

Алексеев А.А. Применение метода Хаффмана для сжатия информации. *В статье рассматривается алгоритм метода Хаффмана для сжатия информации, используется статическое и динамическое кодирование. С помощью написанной программы проведен сравнительный анализ файлов, содержащих различные данные и имеющих разные расширения. Сравнение проводилось как коэффициентом сжатия, так и по времени кодирования и раскодирования.*

Ключевые слова: *сжатие данных, алгоритм Хаффмана, дерево Хаффмана.*

Введение

Развитие телекоммуникационных систем, рост пропускной способности и общего количества линий связи, развитие глобальных компьютерных сетей и расширение спектра предоставляемых ими услуг с одновременным ростом числа пользователей компьютеров и сетей приводит к проблемам хранения и передачи различной информации (текстовой, звуковой, графической и т.п.). Применение сжатия происходит постоянно в любой области информационных технологий и не только. Например, радио и телевизор тоже используют сжатие, но только не данных, а сигнала.

Цель работы – реализация метода Хаффмана для сжатия данных и исследования его возможностей на файлах разных типов.

Нам потребуются следующие понятия определения:

1. Гнездо — это любой символ, т.е. гнездо является уникальным (неповторяющимся) для символов с различными кодами.
2. Частота появления символа — это сколько раз символ с данным кодом встречается в входном потоке.
3. Содержимое гнезда — это частота появления символа.
4. Пустое гнездо — когда содержимое гнезда равно нулю.
5. Статус гнезда — это состояние гнезда на текущий момент, т.е. гнездо может быть доступно (может участвовать при построении дерева) или недоступно (уже участвовало при построении дерева).
6. Двоичное дерево — это набор гнезд, соединенных некоторым способом между собой ветвями (связями). Каждое гнездо может иметь от одного до трёх связей, причем только одна ветвь может связывать с данное гнездо с гнездом более высшего уровня, но каждое гнездо может быть связано двумя ветвями с гнездами более низшего уровня (или не быть связанным вообще).

Классический алгоритм Хаффмана

Идея алгоритма состоит в следующем: зная вероятности символов в сообщении, можно описать процедуру построения кодов переменной длины, состоящих из целого количества битов. Символам с большей вероятностью ставятся в соответствие более короткие коды. Коды Хаффмана обладают свойством префиксности (то есть ни одно кодовое слово не является префиксом другого), что позволяет однозначно их декодировать.

Классический алгоритм Хаффмана на входе получает таблицу частот встречаемости символов в сообщении. Далее на основании этой таблицы строится дерево кодирования Хаффмана (Н-дерево).

1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.
2. Выбираются два свободных узла дерева с наименьшими весами.
3. Создается их родитель с весом, равным их суммарному весу.
4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.
5. Одной дуге, выходящей из родителя, ставится в соответствие бит 1, другой — бит 0.

6. Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.

Перед тем как начать сжатие потока данных, компрессор (кодер) должен построить коды. Это делается с помощью вероятностей (или частот появления) символов. Вероятности и частоты следует записать в сжатый файл для того, чтобы декомпрессор (декодер) Хаффмана мог сделать декомпрессию данных. Это легко сделать, так как частоты являются целыми числами, а вероятности также представимы числами. Обычно это приводит к добавлению нескольких сотен байтов в сжатый файл. Можно, конечно, записать в сжатый файл сами коды, однако это вносит дополнительные трудности, поскольку коды имеют разные длины. Еще можно записывать в файл само дерево Хаффмана, но это потребует большего объема, чем простая запись частот.

В любом случае, декодер должен прочесть начало файла и построить дерево Хаффмана для алфавита. Только после этого он может читать и декодировать весь файл. Алгоритм декодирования очень прост. Следует начать с корня и прочитать первый бит сжатого файла. Если это нуль, следует двигаться по нижней ветке дерева; если это единица, то двигаться надо по верхней ветке дерева. Далее читается второй бит и происходит движение по следующей ветке по направлению к листьям. Когда декодер достигнет листа дерева, он узнает код первого несжатого символа (обычно это символ ASCII). Процедура повторяется для следующего бита, начиная опять из корня дерева [1].

Статическое кодирование

Необходимые определения

1. Гнездо — это любой символ, т.е. гнездо является уникальным (неповторяющимся) для символов с различными кодами.

2. Частота появления символа — это сколько раз символ с данным кодом встречается в входном потоке.

3. Содержимое гнезда — это частота появления символа.

4. Пустое гнездо — когда содержимое гнезда равно нулю.

5. Статус гнезда — это состояние гнезда на текущий момент, т.е. гнездо может быть доступно (может участвовать при построении дерева) или недоступно (уже участвовало при построении дерева).

6. Двоичное дерево — это набор гнезд, соединенных некоторым способом между собой ветвями (связями). Каждое гнездо может иметь от одного до трёх связей, причем только одна ветвь может связывать с данное гнездо с гнездом более высшего уровня, но каждое гнездо может быть связано двумя ветвями с гнездами более низшего уровня (или не быть связанным вообще).

Описание алгоритма.

1. Если число доступных, непустых гнезд менее чем 2, то перейти к пункту 5.

2. Среди доступных гнезд ищем два гнезда с минимальным (не равным нулю) содержимым гнезда.

3. Создается новое гнездо с содержимым равным сумме содержимых двух найденных гнезд. Новому гнезду присваивается статус доступно, а двум найденным гнездам присваивается статус недоступны и дополнительный статус: первому найденному гнезду - 0, второму - 1.

4. Перейти к пункту 1.

5. Это одно оставшееся доступное ненулевое гнездо — мы назовем корень дерева и дополнительного статуса не имеет.

Теперь каждый символ из входного потока будет заменяться последовательностью бит, получаемую шагая от корня дерева по связям к гнезду, соответствующему кодируемому символу, и при этом запоминается дополнительный статус каждого из гнезд в этой цепочке. Сформированная цепочка бит и будет кодом текущего символа. Она имеет переменную длину (от 1 до 256 бит теоретически, практически же почти всегда до 12).

После того как мы закодировали текст, мы должны позаботиться, чтобы он мог быть раскодирован. Для этого распаковщику надо знать дерево, поэтому возникает необходимость сохранения дерева в файле, который содержит упакованный текст, и естественно при наименьшем размере.

Варианты записи дерева в файл:

I. Если взять и записать дерево не сжимая, то понадобится 256 последовательностей бит:

1) 4 бита — длина цепочки битов, являющейся кодом символа

2) Переменная длина (0..15) — сама последовательность бит

В самом худшем случае мы получим дерево, записанное в 256+128=384 байта, что является довольно большой потерей процента упаковки.

II. Теперь попробуем усовершенствовать предыдущий способ. Так как мы знаем, что длина кода практически никогда не превышает 12 бит, то длины последовательностей равные 13,14,15 мы можем использовать в качестве управляющих. Определим:

1) управляющий код 14 при первом появлении — указывает что следующие 8 бит являются кодом первого используемого символа, имеющего наименьший код ASCII (это означает, что если к примеру, использован текстовый файл для упаковки, то наименьший используемый код в тексте имеет возврат каретки (код=13), символы же с кодами меньше 13 не используются в данном тексте и поэтому нет смысла запоминать их в при

упаковке дерева. Иными словами, это означает, что символы с кодами меньше 13 при постройке дерева не участвовали).

2) управляющий код 15 — префикс повторения, означает что следующие 4 бита являются числом 4-х битовых нулевых последовательностей. Позволяет группировать от 3 до 18 таких последовательностей.

3) управляющий код 13 — префикс повторения, означает, что следующие 4 бита, являются числом 4-х битовых нулевых последовательностей. Позволяет группировать от 19 до 34 таких последовательностей.

4) управляющий код 14 при втором появлении означает конец дерева, то есть что символы с кодами больше, чем последнее записанное значение не участвовали при построении дерева.

III. Третий вариант кодирования дерева:

1 байт — количество вершин в дереве;

2 байт — символы являющиеся вершинами дерева (кол-во указано в первом байте)

3 байт — последовательности битов, означающие:

1) Четыре бита - количество бит на символ (0000=1 .. 1111=16)

2) Четыре бита - сколько символов с указанной выше битовой длиной (0000=1 .. 1110=15), если 1111 - то считать еще 5 бит. Пять бит - количество символов (00000=16 .. 11110=46), если 11111 - то считать еще 5 бит и т.д. (количество таких сочетаний указано в первом байте).

4 бит последовательности бит, означающие битовые последовательности, означающие Хаффмановский код символа в том порядке, как они приведены выше [2].

Динамическое кодирование

Динамическое кодирование имеет преимущество по сравнению со статическим кодированием Хаффмана: оно хорошо приспосабливается к быстро меняющимся данным и нет необходимости хранить дерево вместе с запакованным тестом.

Первоначально дерево выглядит так: все символы ASCII имеют частоту появления, равную единице, и строится дерево. Читается символ из входного потока. Частота соответствующего символа в дереве увеличивается на 1 и дерево перестраивается. По мере того, как некоторое количество символов считается из входного потока, и по мере того, как часто перестраивается дерево, символы, встречающиеся чаще, будут иметь меньшую длину кода, чем остальные.

Рассмотрим описание метода на основе примера:

"маша_мыла_раму,_мама_мыла_тоже"

Подсчитаем частоты появления символов:

'а' - 7, 'м' - 6, '_' - 5, 'ы' - 2, 'л' - 2, 'р' - 1, 'у' - 1, ',' - 1, 'т' - 1, 'о' - 1, 'ж' - 1, 'е' - 1.

Дерево запакуется: (по байтам) 12 (количество символов - 1), 'а','м','_','ы','л','ш','р', 'у',',','т','о','ж','е', (сами символы) $11 (шестнадцатеричное. Длина кода = 7-1=6), $20, $31, $47, (далее идут Хаффмановские коды символов)

|00 01 100 1|010 1011 1|1000 1100|1 11010 11|011 11100| 11101 111|10 11111

Знак "|" делит битовые последовательности на байты.

Дерево пакуется в 199 бит (24 байта и 7 бит)

Далее идет запакованный текст:

01 00 11000 00 100 01 1010 1011 00 100 11001 00 01 11010 11011 100

(м а ш а _ м ы л а _ р а м у , _)

01 00 01 00 100 01 1010 1011 00 100 11100 11101 11110 11111

(м а м а _ м ы л а _ т о ж е)

Всего: 12 байт и 1 бит = 97 бит.

Общая длина с деревом: 38 байт.[3]

Анализ возможностей программы

По данному алгоритму была написана программа на языке C# в среде разработки Visual Studio 2013, и с помощью неё проведен сравнительный анализ по сжатию (и восстановлению) файлов с различными данными (и разными расширениями) с помощью написанной программы.

Таблица 1 – Коэффициент сжатия

Тип файла	Формат данных	Исходные данные (в КБ)	Сжатые данные (в КБ)	Коэффициент сжатия
docx	текст	31	32	1,1
doc	текст	6161	3111	0,5
max	3D-изображение	200	95	0,475
mkv	видео	666540	664589	0.99
pptx	презентация	317	308	0.97

ppt	презентация	1109	1000	0.9
jpg	картинка	619	620	1
bmp	картинка	6751	6499	0.96
gif	картинка	984	985	1
png	картинка	3188	3188	1
txt	текст	420	226	0,54
png	Черно-белая картинка	1387	1388	1,01
jpg	Черно-белая картинка	778	778	1
txt	Числа	1427	647	0,45
rtf	текст	11888	5488	0,46

Таблица 2 – Время сжатия и раскодирования

Тип файла	**Формат данных**	**Время сжатия (в сек)**	**Время раскодирования (в сек)**
docx	текст	1	0,1
doc	текст	1	0,2
max	3D-изображение	1.5	0,5
mkv	видео	941	120
pptx	презентация	1.5	0,1
ppt	презентация	2	0,1
jpg	картинка	1	0,1
bmp	картинка	10	2
gif	картинка	2	1
png	картинка	6	0,1
txt	текст	1	0,1
png	Черно-белая картинка	2	0,1
jpg	Черно-белая картинка	1	0,1
txt	Числа	1	0,1
rtf	текст	10	0,1

Как видно из таблицы 1, в расширениях .docx, .mkv, .pptx, .ppt, .jpg, .bmp, .gif, .png коэффициент сжатия близок или больше 1. Связано это, скорее всего, с тем, что в файлы с такими расширениями сжатие уже заложено изначально.

Из таблицы 1 видно, что хороший коэффициент сжатия (около 0,5) получается для файлов с расширениями .doc,. max, .txt, .rtf. Для таких расширений можно использовать сжатие по методу Хаффмана.

В таблице 2 представлено сравнение между временем сжатия и раскодирования. Восстановление происходит без потери данных, лишь необходимо знать тип файла, т.к. программа не запоминает тип сжатого файла. По времени восстановление происходит гораздо быстрее, чем сжатие.

Выводы

В работе представлена реализация метода Хаффмана на языке C# и исследованы возможности полученного программного продукта. Сравнительный анализ проводится на файлах, содержащих данные различной природы и имеющих разные расширения. Лучший коэффициент сжатия (≈0,5) получается для файлов с расширениями .doc,. max, .txt, .rtf. Восстановление происходит без потерь и быстрее, чем сам процесс сжатия.

Литература

1. Ватолин Д.С. Алгоритмы сжатия изображений / Метод.пособие. М., 1999. 76 с.
2. Кормен Т. Алгоритмы: построение и анализ // Т. Кормен, Ч. Лейзерсон, Р.Ривест, К.Штайн – Москва: Вильямс, 2012. – 1296 с.
3. Дьяков А. Метод сжатия Хаффмана 2010 – Режим доступа: http://www.codenet.ru/progr/alg/huffman/
4. Верещагин Н.К. Колмогоровская сложность и алгоритмическая случайность // Н.К. Верещагин, В.А. Успенский, А. Шень – Москва: МЦНМО, 2013. – 575 с.
5. Сэломон Д. Сжатие данных, изображений и звука Москва: Техносфера, 2004. 368 с.

Алексеев А.А. Применение метода Хаффмана для сжатия информации. *В статье рассматривается алгоритм метода Хаффмана для сжатия информации, используется*

статическое и динамическое кодирование. С помощью написанной программы проведен сравнительный анализ файлов, содержащих различные данные и имеющих разные расширения. Сравнение проводилось как коэффициентом сжатия, так и по времени кодирования и раскодирования.

Ключевые слова: *сжатие данных, алгоритм Хаффмана, дерево Хаффмана.*

Alekseev Anton. The application of the Huffman method to compress the information. *The article discusses the Huffman algorithm to compress the information used for static and dynamic encoding. With the help of the written program, a comparative analysis of files containing different data and having different extensions is carried out. The comparison was carried out both by compression ratio and by encoding and decoding time.*

Keywords: *data compression, Huffman algorithm, Huffman tree.*

УДК 004.942

Повышение информационно-образовательного уровня сайта кафедры на основе использования интеллектуальных систем

Артеменко О.Г., Федяев О.И.
Донецкий национальный технический университет
olga_artyomenko@bk.ru, fedyaev@donntu.org

Артеменко О.Г., Федяев О.И. Повышение информационно-образовательного уровня сайта кафедры на основе использования интеллектуальных систем. *Статья посвящена вопросам расширения функциональных возможностей информационно-образовательного сайта мультимедийными иллюстрациями работы интеллектуальных систем, созданных на кафедре. Рассмотрен пример контента об интеллектуальной мультиагентной системе моделирования процесса трудоустройства студентов.*

Ключевые слова: *информационно-образовательный сайт, интеллектуальные системы, обучение, агентно-ориентированные системы*

Введение

В настоящее время компьютерная сеть Интернет для многих людей стала средством оперативного получения самой различной информации. В российском сегменте Интернета уже накопилось значительное число порталов и сайтов, которые по чисто формальным признакам можно рассматривать как образовательные.

В литературе, посвященной образовательным сайтам, постоянно поднимается вопрос о критериях и методике, которые следует использовать при их анализе и оценке. Сформулировать по этому поводу чёткие и обоснованные рекомендации достаточно проблематично. Сложность ситуации вызвана, прежде всего, тем, что для использования хотя бы простейших количественных статистических оценок необходимо стандартизовать на сайтах формы представления важнейших ресурсов и разделов, однако соответствующие рекомендации по этому поводу пока отсутствуют. Сказанное в первую очередь касается информационной составляющей сайтов.

Важнейшим показателем качества образовательного ресурса является его содержание, то есть ценность его как учебного материала. Возможность оценить эту позицию непосредственно количественной мерой отсутствует, но здесь может быть использован опосредованный критерий по числу обращений к ресурсу со стороны пользователей, то есть посещаемость ресурса. Посещаемость сайта можно повысить путём включения в информационный материал различных мультимедийных фрагментов, визуально иллюстрирующих структуру и функции сложных систем (процессов). В этом случае образовательный ресурс непременно будет постоянно востребован и для получения такой оценки на сайтах должен присутствовать соответствующий статистический программный модуль.

В ранее разработанном прототипе информационно-образовательного сайта представлена статическая информация о названиях тем, связанных с интеллектуальными разработками кафедры программной инженерии ДонНТУ [8]. За одними названиями тем не видно сути представленных на сайте работ, т.е. в нём не представлены результаты в виде постеров и иллюстраций динамики интеллектуальных процессов.

Поэтому целью данной статьи является разработка рекомендаций по улучшению существующего сайта на основе включения в его контент мультимедийных компонент, иллюстрирующих возможности разработанных интеллектуальных систем. Для этого в работе рассматривается пример разработки многоагентной системы моделирования процессов трудоустройства выпускников вуза, как интеллектуальной программной системы, с целью выделения главных особенностей системы для составления по этой разработке постера для включения в сайт кафедры.

Особенности структуры информационно-образовательного сайта кафедры

Несмотря на то, что значительная часть информации практически по всем областям знаний представлена в Интернете, проблема эффективного обеспечения научного сообщества информацией по интересующим его тематикам ещё далека до своего решения [2]. Информационно-обучающий интернет-ресурс, предназначенный для информационной поддержки научной и учебной деятельности выпускающей кафедры «Программная инженерия» (ПИ) Донецкого национального технического университета в области программной инженерии интеллектуальных программных разработок, является одним из шагов к решению таких задач, как:

– сведение информационных ресурсов, относящихся к такой важной области знаний, как разработка интеллектуальных программных систем на кафедре ПИ, в единое (компактное) информационное пространство;

– поддержка логической целостности системы семантических описаний информационных ресурсов, сведенных в единое информационное пространство, по актуальной для программной инженерии тематике;

– обеспечение разработчикам сайта возможности открытого содержательного доступа (изменение, обновление) к структурированным информационным ресурсам сайта;

– оперативное информирование специалистов по программированию интеллектуальных систем о результатах деятельности кафедры ПИ в области искусственного интеллекта и формирование на основе взаимного интереса вокруг этой тематики круга специалистов-единомышленников [2].

Процесс разработки кафедрального сайта регламентировался жизненным циклом данного типа изделия. При создании информационно-образовательного портала большое внимание уделено проектированию непосредственно структуры веб-ресурса.

Тематика информационно-образовательного портала включает: новостную ленту и рассылки; электронные учебники и библиотеки; расширенные многоуровневые средства навигации и поиска; каталоги; компьютерные демонстрации; универсальные обучающие среды (разработанные на кафедре интеллектуальные системы); справочный отдел для поиска информации. Структура кафедрального интернет-ресурса изображена на рисунке 1.

Разделы сайта	Текущие мероприятия кафедры, связанные с ИИ	Новости по ИИ на кафедре
		Текущие мероприятия по ИИ
	Методы ИИ в учебном процессе	Учебные дисциплины по ИИ на кафедре
		Тематика магистерских работ
		Список магистров
	Методы ИИ в научно-технических проектах	Научные направления по ИИ на кафедре
		Тематика аспирантуры
		Участие в научных конференциях
	Этапы развития области знаний по ИИ на кафедре	Первые шаги
		Связь с другими организациями
		Интеллектуальные системы, разработанные на кафедре

Рисунок 1 - Разделы информационного сайта кафедры

Структура презентации разработанной интеллектуальной системе для сайта кафедры

Структура презентации разработанной интеллектуальной системы, которая должна улучшить иллюстративную составляющую сайта кафедры, должна включать следующие части: актуальность разработки, решаемые задачи, методы их решения, архитектуру программной системы и имитационную модель, иллюстрирующую динамику интеллектуального процесса решения задач. Рассмотрим это на примере моделирования системы трудоустройства выпускников вуза. Вопрос трудоустройства как никогда актуален для будущего выпускника высшего учебного заведения. Это связано с тем фактом, что выбор рабочего места может оказать значительное влияние на успешность построения дальнейшей карьеры.

Именно заинтересованность человека в своем будущем мотивирует к развитию такого направления как прогнозирование.

Прогнозирование трудоустройства выпускника кафедры преследует следующие цели:

- прогнозирование успеха трудоустройства выпускника на предложенных фирмах;
- методика определения привлекательности предприятия для студентов;
- выявление возможные пробелы в знаниях студентов;
- указание обучающей кафедре на возможные недостатки обучающей программы [3];
- получение эффекта «тренировки» при прохождение виртуального собеседования;
- прогнозирование качества профессионального обучения студентов в зависимости от их личностных характеристик и других факторов.

Задачи такого типа являются трудно формализуемыми и поэтому не могут быть решены традиционными математическими методами [4]. Кроме того, участники рассматриваемого процесса территориально удалены друг от друга, неоднородны по структуре и их деятельность интеллектуальна по своей природе. Эти особенности обуславливают целесообразность применения теории интеллектуальных агентов к разработке имитационной модели для анализа и управления процессами подготовки кадров и их трудоустройства.

Основная задача, способствующая повышению информационно-образовательного сайта кафедры ПИ, заключается в разработке агентно-ориентированной модели социально-экономического процесса на примере процесса трудоустройства молодых специалистов.

Агентно-ориентированные модели являются приоритетными для моделирования социально-экономических процессов, т.к. позволяют обеспечить большую адекватность системе моделирования.

Агентно-ориентированный анализ процесса подготовки и трудоустройства молодых специалистов показал, что его субъекты взаимосвязаны, образуют распределённую, неоднородную и интеллектуальную систему. Поэтому разработка имитационной модели такой системы может быть успешно выполнена на основе методов агентно-ориентированного моделирования, представленная на рисунке 2.

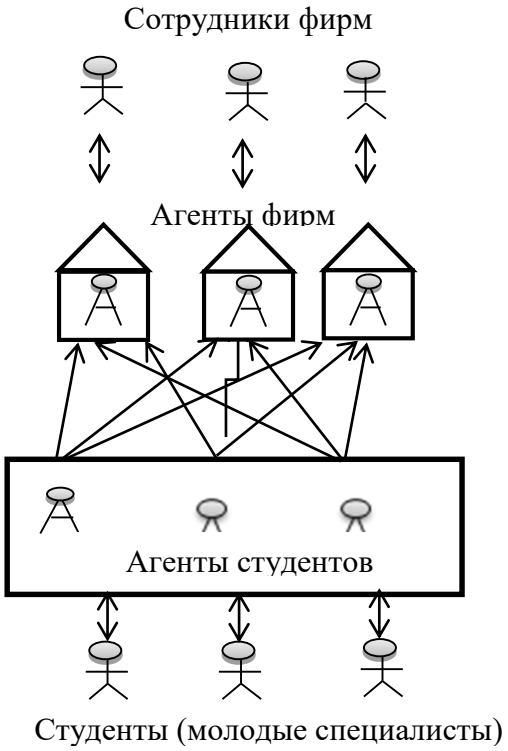


Рисунок 2 - Схема мультиагентной системы моделирования трудоустройства специалистов

Многоагентная технология считается одной из самых важных и многообещающих областей развития информационных технологий. Разработанная программная система имитации прохождения студентами собеседований будет учитывать имеющиеся у реального студента знания (полученные посредством передачи их своему агенту), а также принимать во внимание тестовые задания работодателя с возможностью их оперативной замены на другие. Более того, система будет обучаема, и прогноз трудоустройства с каждым разом будет иметь большую точность.

Пример построения контента о презентации многоагентной системы трудоустройства студентов

Молодой специалист при поиске работы проходит собеседование на фирме и(или) выполняет тестовые задания, по результатам которых соискателя принимают или не принимают на имеющуюся вакансию. В то же время выпускник может отказаться от предлагаемой должности, если его не устраивают предложенные фирмой условия труда, и продолжить поиски.

Фирма при собеседовании даёт выпускнику анкету, в которую он заносит данные о себе и выполняет тестовые задания, после чего фирма оценивает анкету и ответы соискателя. Выпускник в то же время получает информацию о требованиях к соискателям, условиях работы и формирует для себя субъективную оценку о привлекательности фирмы. После завершения этих процессов фирма и выпускник должны принять общее соглашение о заключении или не заключении трудового договора.

Исходя из схемы диалога во время собеседования выпускника с представителем фирмы, была построена структура агентно-ориентированной модели, представленная на рисунке 3.

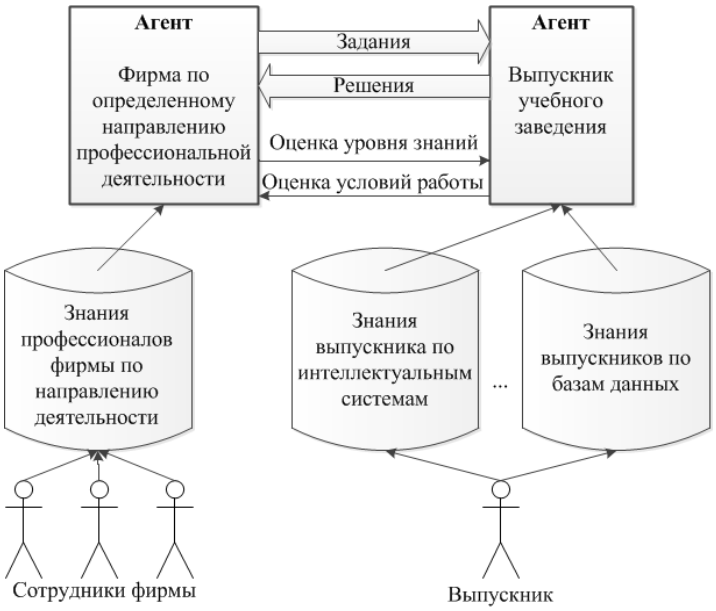


Рисунок 3 — Структура агентно-ориентированной системы трудоустройства

Эта модель состоит из двух типов искусственных (программных) агентов: агента фирмы и агента выпускника. Взаимодействие агентов начинается с того, что агент фирмы рассылает условия заданий всем агентам выпускников. Агенты выпускников решают полученные задания и возвращают свои решения (ответ) фирме, которая прислала им задание. Далее, если агент фирмы принял решение о том, что некоторый агент выпускник удовлетворяет требованиям фирмы, то отправляет ему сообщение с предложением принять участие в конкурсе на имеющуюся вакансию на предлагаемых условиях труда. Если этот агент выпускника побеждает в конкурсе с другими отобранными агентами и его устраивают условия труда на фирме, то заключается трудовой договор. В противном случае трудовой договор между ними не заключается, и агент продолжает поиски работы на других фирмах.

При построении искусственных агентов была выбрана нейросетевая архитектура. Поскольку речь идёт об имитации поведения множества людей, то применяемые модели должны иметь возможность обучаться на данных реально проводимых персональных опросов нескольких десятков выпускников и представителей фирм.

Интеллектуальными задачами, которые решаются нейросетевым способом, являются: оценивание выпускником условий труда на фирме, решение студентом типовых заданий по профилю фирмы, оценивание фирмой ответов выпускника на тестовые задания. С этими задачами успешно справляется трёхслойный персептрон с нелинейной функцией активации, изображенный на рис 4:

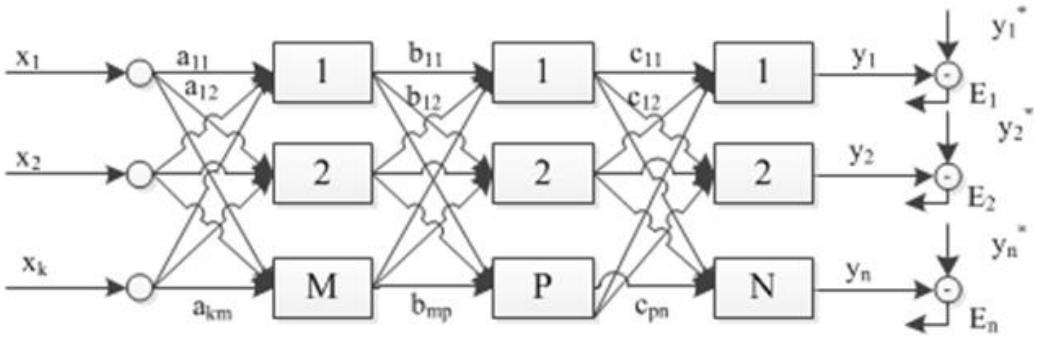


Рисунок 4 - Структура трёхслойной нейронной сети

Для передачи профессиональных навыков от молодого специалиста (источника знаний) к нейросетевому программному агенту использовались коммуникативные методы извлечения знаний из реальных студентов-выпускников и алгоритм настройки нейросети по стратегии «обучение с учителем». Для извлечения знаний из выпускников были составлены опросные анкеты по каждому разделу программной инженерии (например, «Системы искусственного интеллекта», «Базы данных и т. д.), из которых формировались обучающие множества для нейросетей. В анкету входил набор типовых заданий по каждому из намеченных разделов учебного плана

специальности и правильные ответы к ним в виде номеров необходимых знаний и умений из предлагаемого списка для их решения.

Рисунок 5 - Схема извлечения знаний для обучения нейронной сети агента соискателя

Рассмотрим один из экспериментов, посвящённый моделированию процесса трудоустройства выпускников с низким уровнем профессиональной подготовки. В эксперименте запланировано участие трёх фирм, у которых имеется определённое количество вакансий: первый агент фирмы (company-1) имеет 2 вакансии, второй (company-2) – 4 и третий (company-3) – 1. В качестве кандидатов на работу в этих фирмах запланировано участие 15 выпускников ВУЗа с низким уровнем знаний по профилю данных фирм. Условия труда, которые предлагают фирмы в данном эксперименте, оценивались выпускниками по 8 показателям: заработная плата, предоставление жилья, форма собственности и т. д [7].

Динамика процесса визуализируется с помощью специальных окон (рис.6), в которых на каждом шаге моделирования отображается состояние трудоустройства, т.е. сколько и кого уже отобрала каждая фирма, какие выпускники ещё проходят собеседование и т.д.

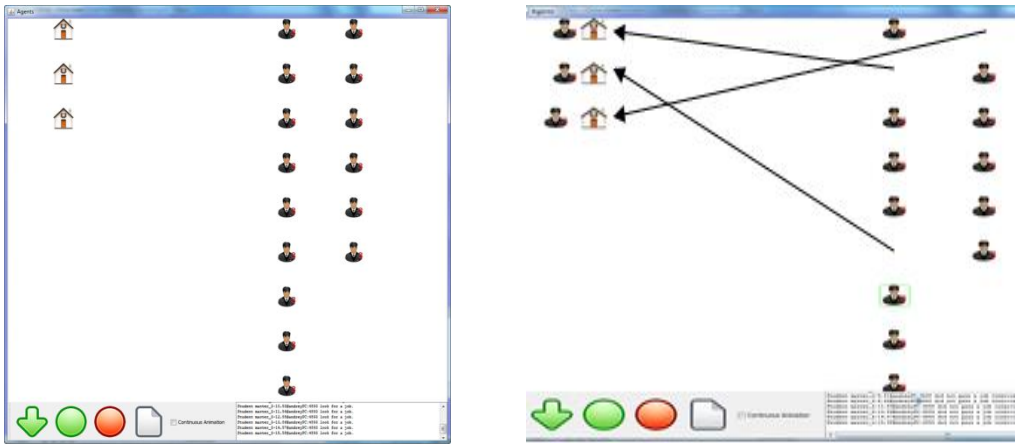


Рисунок 6 — Начальное и конечное состояние моделирования

С помощью данной модели были проведены и другие эксперименты, соответствующие реальным проблемным ситуациям при трудоустройстве выпускников ВУЗа.

Заключение

Выполнен анализ возможных способов повышения информационно-образовательного уровня сайта кафедры. Предложены рекомендации по улучшению существующего кафедрального сайта путём включения в его контент мультимедийных компонент, иллюстрирующих возможности разработанных интеллектуальных

систем.

В качестве примера интеллектуальной системы была рассмотрена мультиагентная система прогнозирования трудоустройства будущих выпускников кафедры, реалистично отражающая поведение людей и фирм при поиске работы в зависимости от социальных, профессиональных и экономических факторов, делегированных членам искусственных обществ модели системы трудоустройства.

Литература

1. Горбунова Л. И., Субботина Е. А. Использование информационных технологий в процессе обучения // Молодой ученый. – 2013 г.
2. Артеменко О.Г., Федяев О.И., Сысолятина С.А. Информационно-образовательный сайт интеллектуальных программных разработок кафедры программной инженерии / Материалы 5-й Международной научно-технической конференции « Современные информационные технологии в образовании и научных исследованиях» (СИТОНИ-2017). Донецк: ДонНТУ, 2018. – С.244-248.
3. Архитектура сайта [Электронный ресурс] – Режим доступа: http://www.hmx.ru/arhitektura_saita.html
4. Правильная архитектура сайта [Электронный ресурс]– Режим доступа:http://fb.ru/article/289653/pravilnaya-arhitektura-sayta-kak-izbejat-oshibok
5. Комаревцев Е.М. Образовательные порталы как средство систематизации и структурирования информации: Учебное пособие. — Ставрополь: Изд-во СГУ, 2005.
6. Д. Лещев. Создание интерактивного Web-сайта. Учебный курс: Учебное пособие.-СПб: Изд-во Питер, 2003 г.
7. Тарасов В.Б. От многоагентных систем к интеллектуальным организациям: философия, психология, информатика. – М.: Эдиториал УРСС, 2002. - 352 с.

Артеменко О.Г., Федяев О.И. Повышение информационно-образовательного уровня сайта кафедры на основе использования интеллектуальных систем*. Статья посвящена вопросам расширения функциональных возможностей информационно-образовательного сайта мультимедийными иллюстрациями работы интеллектуальных систем, созданных на кафедре. Рассмотрен пример контента об интеллектуальной мульти-агентной системе моделирования процесса трудоустройства студентов.*

Ключевые слова: *информационно-образовательный сайт, интеллектуальные системы, обучение, агентно-ориентированные системы*

Artemenko O. G., Fedyaev O. I. Improving the information and educational level of the Department's website based on the use of intelligent systems. *The article is devoted to the expansion of the functionality of information and educational site multimedia illustrations of intelligent systems created at the Department. An example of the content of the intellectual multi-agent system simulation of the process of employment of students.*

Keywords: *information and educational website, intelligent systems, training, agent-oriented systems.*

УДК 004

Лексический и синтаксический разбор как первый этап статического анализа SQL-кода

Варяник А.С., Грищенко В.И.
Донецкий национальный технический университет, г. Донецк
кафедра программной инженерии
varyanik2@gmail.com
victor.grischenko@gmail.com

А.С.Варяник, В.И. Грищенко Лексический и синтаксический разбор как первый этап статического анализа SQL-кода. *Выявлена сущность и предмет статического анализа SQL-кода для оценки качества структуры базы данных. Выделены этапы подготовки исходного SQL к процессу статического анализа. Исследованы существующие средства реализации лексического и синтаксического разбора и их применения в контексте задачи.*

Ключевые слова: *лексическией разбор, синтаксический разбор, статический анализ, токены, синтаксическое дерево, система управления базами данных, спецификация, диалект*.

Введение

В статье [1] были рассмотрены критерии качества структуры реляционной базы данных. Описанные критерии могут быть подвержены оценке путем проведения процедуры статического анализа исходного SQL-кода, определяющего анализируемую базу.

Статический анализ кода играет важную роль в современном мире разработки прикладного программного обеспечения. Правильное и своевременное встраивание этапа статического анализа в процесс написания кода может значительно сэкономить как временные, так и финансовые ресурсы. Учитывая, насколько широко, на данный момент, используются СУБД с поддержкой SQL, инструмент, позволяющий комплексно оценить качество и связность написанного SQL-кода, может быть крайне востребован.

Стоит отметить, что сам процесс анализа кода является завершающим этапом цепочки последовательных операций. В его основе лежат процедуры лексического и синтаксического разбора, реализация которых напрямую зависит от специфики текста, в данном случае программного кода.

В данной статье будет сформулировано понятие статического анализа в целом и в контексте «structured query language», а также предметно описан процесс лексического и синтаксического разбора, включая средства его реализации.

Статический анализ программного кода

Статический анализ кода – это процесс, заключающийся в выявлении логических и стилистических ошибок в исходном коде. Статический анализ в какой-то степени является аналогом процесса обзора кода (code review), но в автоматизированном виде. Обычно на практике используются оба подхода, так как каждый имеет ряд своих достоинств [7].

Статический анализ выполняется на этапе кодирования, что позволяет исправлять некачественный код еще до его попадания в реальную среду выполнения или даже в общее хранилище (репозиторий). Данный процесс позволяет выявлять как критические ошибки в программном коде, приводящие к исключительным ситуациям при выполнении программы, так и минорные нарушения программистом стилистических спецификаций и рекомендаций. Если нарушения кодом логики языка от программы к программе будут соответствовать общим правилам, заложенным в его инфраструктуре, то правила, касающиеся стиля кодирования, могут зависеть от конкретной компании или команды разработчиков. Поэтому, зачастую, системы, выполняющие статический анализ, позволяют гибко конфигурировать и задавать собственные правила, исходя из которых, будет проверяться программный код.

Важно отметить, что пропорционально росту программной системы растет и количество ошибок, которые могут быть допущены при написании исходного кода, поэтому в сфере разработки баз данных, где нередко можно встретить крупные программные комплексы, статический анализ особенно актуален.

Касаемо SQL-кода статический анализ в целом работает так же, как и для других языков программирования. Существует общий стандарт языка, который имеет собственную лексическую и синтаксическую формализацию. Каждая СУБД, в свою очередь, дополняет или переопределяет стандартную спецификацию, образуя собственные диалекты: PostgreSQL. Oracle. MySQL и т.д. Данный факт значительно усложняет задачу разработки универсального инструмента для автоматизированного статического анализа, так как при анализе необходимо учитывать особенности конкретного диалекта.

Конкретизируя предмет статического анализа SQL-кода, стоит обратиться к критериям и метрикам качества, описанным в [1]. Например, описанный критерий ссылочной целостности сложно определить не имея контекста и предметной области: средства статического анализа не позволяют определить логические связи между сущностями базы данных. Используя лексический и синтаксический разбор можно разложить любые инструкции языка на составляющие и определить, следует ли программист стилистическим нормам описания связи и в таком случае установил ли он ее на самом деле. Либо наоборот, отталкиваясь от формального наличия внешнего ключа проверить, соблюдены ли правила его именования.

Подобным образом можно проанализировать любое формализованное правило. Отталкивая от спецификации SQL и сформированных обществом разработчиков рекомендаций можно сформировать необходимый перечень этих правил, для эффективной работы автоматизированной системы статического анализа. Подробное описание данного перечня будет рассмотрено в последующих работах.

Основные понятия и идея лексического и синтаксического разбора

Лексический анализ или разбор представляет собой процесс анализа и распознавания определенных элементов или их групп (лексем) во входном наборе или последовательности символов, для их последующей идентификации и выделении токенов (токенизация). В общем случае в процессе лексического анализа входными данными является поток символов, а выходными – набор классифицированных токенов (классификация может иметь различные уровни сложности) [5].

Программная система, выполняющая лексический анализ, называется лексическим анализатором или лексером. Любая подобная система работает в два этапа. Первый этап заключается в определении лексем, которые могут встречаться во входном потоке и их сопоставлении с регулярными выражениями, с помощью которых эти лексемы можно найти. На втором этапе происходит непосредственно выявление в потоке символов значений определенных ранее лексем в виде идентифицированных токенов. Полученный в результате набор токенов в последствие может быть передан на вход синтаксическому анализатору или парсеру, чья задача будет состоять в проведении синтаксического анализа.

Синтаксическим анализом называется процесс обработки набора токенов (в общем случае лексем) определенного формального языка с его грамматикой. Данный процесс выполняется синтаксическим анализатором (парсером), который принимает на вход результаты работы предшествующего лексического анализа. Выходными данными синтаксического анализа является синтаксическое дерево или дерево разбора, представляющее из себя всю синтаксическую структуру анализируемой последовательности символов [6].

Связка из лексического и синтаксического анализатора (рис.1) используется во многих областях программирования и не только, но ручное их написание является достаточно трудоемким процессом, поэтому во многих программных продуктах, компиляторах, интерпретаторах и т.д. используются генераторы анализаторов.

Рисунок 1 – Схема работы связки лексера и парсера

Генераторы лексических и синтаксических анализаторов принимают на вход лексическую и синтаксическую структуру языка в некотором формальном представлении, на выходе же мы получаем исходный код лексера и парсера на одном из языков программирования (язык зависит от выбранного генератора).

Инструменты реализации задачи генерации лексического и синтаксического анализатора для SQL

Задача генерации лексических и синтаксических анализаторов не нова и на данный момент существует большое количество программных продуктов, позволяющих ее решить.

Стоит отметить популярную связку из программ Flex и Bison, которые являются аналогами и поддерживают обратную совместимость с продуктами Lex и Yacc, но используются в системах на базе пакетов GNU [8].

Здесь Flex (Lex) – является генератором лексических анализаторов и позволяет генерировать лексеры в виде функции на языке C, в свою очередь Bison (Yacc) является генератором синтаксических анализаторов и генерирует исходный код не только на языке C, но и на популярных языках C++ и Java.

Данное сочетание широко используется во множестве программных систем в свою очередь из-за того, что оба программных продукта являются открытыми и имеют качественную документацию.

Но при исследовании предметной области и анализе существующих средств реализации для более глубокого ознакомления и изучения был выбран иной программный продукт – ANTLR.

ANTLR (Another Tool for Language Recognition) – представляет собой открытое программное обеспечение, предназначенное для генерации нисходящих анализаторов для формальных языков.

Основным преимуществом ANTLR перед другим программным обеспечением является то, что данный продукт объединяет в себе функции генерации лексического и синтаксического анализаторов, позволяя при этом использовать единую нотацию. Также генератор предоставляет возможность получать информацию об ошибках в процессе анализа и возможность восстановления после ошибок.

ANTLR нашел применение во многих крупных проектах: использовался для реализации Python и Groovy, а так же в различных IDE для анализа различных языков программирования [2].

Говоря подробнее об использовании ANTLR, стоит отметить удобство и простоту его нотации. В одном файле описываются правила разбиения текста на токены (лексемы) для лексера и правила обработки этих токенов для парсера.

В контексте языка SQL описание лексического анализатора будет представлять собой описание ключевых слов и всевозможных литералов, допускающихся к использованию, а описанием синтаксического анализатора будет перечень правил, по которым могут формироваться SQL-инструкции из набора описанных ранее токенов. На рисунке 2 показан простой пример описания правила для оператора «JOIN» и токенов, которые могут быть его составляющими.

```
K_LEFT : L E F T;
K_NATURAL : N A T U R A L;
K_OUTER : O U T E R;
K_INNER : I N N E R;
K_CROSS : C R O S S;
K_JOIN : J O I N;

join_operator
  : ','
  | K_NATURAL? ( K_LEFT K_OUTER? | K_INNER | K_CROSS )? K_JOIN
  ;
```

Рисунок 2 – Пример описания лексера и парсера для SQL в ANTLR

На платформе «github» существуют проекты с описанием лексического и синтаксического анализаторов SQL для генератора ANTLR. Так в [3] реализована работа с SQLLite, а в [4], являющемся его ответвлением, присутствует реализация классической спецификации SQL и отдельная реализация диалекта Oralce. Оба проекта далеко не идеальны и требуют значительных доработок в формализации анализаторов и их расширению для различных SQL диалектов, но они являются хорошей базой, от которой можно отталкиваться в собственной реализации.

Подводя итог под темой лексического и синтаксического анализа, стоит сказать, что только после проведения необходимого разбора исходного кода и получения синтаксического дерева можно приступать непосредственно к процессу статического анализа. От методов и алгоритмов обработки этого дерева будет зависеть скорость и эффективность процесса анализа SQL-кода и структуры анализируемой базы данных. На данном этапе ANTLR также имеет значительные преимущества перед своими конкурентами, так как предоставляет широкие возможности для работы с выходным синтаксическим деревом. На рисунке 3 представлена полная последовательность действий, необходимая для автоматизированного анализа SQL-кода.

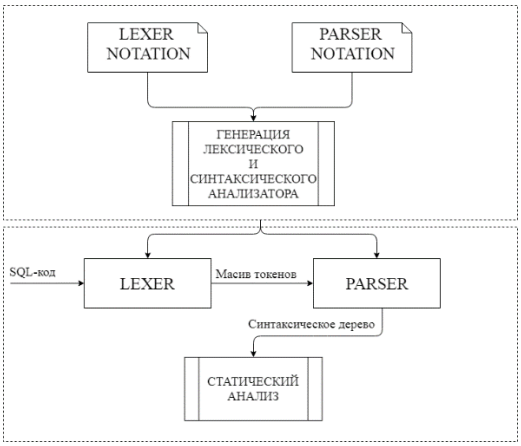


Рисунок 3 – Полный процесс статического анализа SQL-кода

Выводы

В данной статье был проведен анализ понятия статического анализа SQL-кода. Определен подход к этому процессу в использовании его для оценки качества структуры базы данных. Также были выделены этапы обработки исходного кода, предшествующие статическому анализу, исследованы существующие средства их реализации и готовые решения с их применением. В дальнейшем планируется закончить процесс реализации этапа подготовки SQL и приступить к детализации процесса обработки подготовленных структур и данных.

Литература

1. Варяник А.С. Грищенко В.И / Выявление критериев и компонентов качества структур реляционных данных для системы автоматизированного анализа / Донецк: ДонНТУ, 2018г. – 68-71с.
2. ANTLR (ANother Tool for Language Recognition) [Электронный ресурс] – Режим доступа http://www.antlr.org/
3. Github: bkiers/sqlite-parser [Электронный ресурс] – Режим доступа https://github.com/bkiers/sqlite-parser
4. Github: lchaboud-restlet/antlr-sqlparser [Электронный ресурс] – Режим доступа https://github.com/lchaboud-restlet/antlr-sqlparser
5. Хантер Р. / Основные концепции компиляторов / «Вильямс» М., 2002г. – 256с.
6. А. Ахо, Дж. Ульман./ 3. Теория синтаксического анализа, перевода и компиляции / 1978г. – 614с.
7. Синтаксический анализ кода [Элекстронный ресурс] – Режим доступа https://habr.com/company/pvs-studio/blog/254855/
8. Р. Левин / Flex & Bison / О'Рейли, 2009г. – 294с.

А.С.Варяник, В.И. Грищенко Лексический и синтаксический разбор как первый этап статического анализа SQL-кода. *Выявлена сущность и предмет статического анализа SQL-кода для оценки качества структуры базы данных. Выделены этапы подготовки исходного SQL к процессу статического анализа. Исследованы существующие средства реализации лексического и синтаксического разбора и их применения в контексте задачи.*

Ключевые слова: *лексическией разбор, синтаксический разбор, статический анализ, токены, синтаксическое дерево, система управления базами данных, спецификация, диалект.*

A.Varyanik, V.Greeschenko Lexical and syntactic analysis as the first stage of static analysis of SQL-code. *The essence and subject of static analysis of SQL-code for assessing the quality of the database structure are revealed. The stages of preparing the original SQL to the process of static analysis are highlighted. Investigated the existing means of implementing lexical and syntactic parsing and their usage in the context of the task.*

Keywords: *lexical parsing, syntactic parsing, static analysis, tokens, syntax tree, database management system, specification, dialect.*

УДК 004.4’272

Обзор актуальных методов процедурной генерации комнат в видеоиграх

Вивденко В.С., Копытова О.М.
Донецкий национальный технический университет
hectovlad@yandex.ru

В.С. Вивденко, О.М. Копытова Обзор актуальных методов процедурной генерации комнат в видеоиграх. *В статье дан обзор нескольких самых актуальных методов процедурной генерации контента, которые могут быть использованы для генерации комнат и уровней в видеоиграх, обозначены их особенности и рекомендуемые области применения. Рассмотрены их недостатки и определены пути доработки.*

Ключевые слова: *разработка игр, процедурная генерация контента, уровни, BSP-деревья, клеточные автоматы*

Введение

Процедурная генерация контента – одно из актуальных и быстроразвивающихся направлений в сфере разработки компьютерных игр. Разработки в этой области могут позволить значительно сократить расходы на производство игр, а также повысить качество продукта путем создания более привлекательного игрового наполнения и механик. Однако этот подход требует тщательной обдуманного и проработанного применения, так как, будучи выполнен некачественно, способен значительно испортить ценность игры. Успешное применение процедурной генерации контента требует широкую осведомленность в наборе доступных инструментов и их тонкостей. Эта статья посвящена обзору методов процедурной генерации, которые позволяют решить одну из самых популярных задач в этой области – создание игровых комнат и уровней. В ней рассматриваются одни из самых оптимальных алгоритмов, которые подходят для выполнения этой задачи, а также их тонкости и области применения.

Процедурная генерация комнат

Процедурная генерация подразумевает автоматическое создание наполнения игрового мира, к которому могут относиться всевозможные игровые объекты, такие как уровни, комнаты, предметы, противники, инструменты игрока. С помощью алгоритмов генерации могут задаваться как внешний вид и свойства этих объектов, так и их расположение и даже поведение. Кроме того, процедурно-сгенерированными могут быть текстуры, музыкальное сопровождение и сюжетные линии.

Процедурная генерация контента позволяет значительно снизить затраты на разработку компьютерных игр, позволяя с помощью одной процедуры создавать неограниченное количество различных объектов. Кроме того, генерация может происходить динамически во время игрового процесса, создавая для каждого игрока уникальные ситуации и неповторимый опыт при каждой новой попытке, значительно повышая интерес к игре.

Однако такой подход к разработке требует точности и детально проработки алгоритмов процедурной генерации. Контент, созданный таким образом, должен быть мало отличим от созданного вручную, чтобы не вызвать негативные впечатления.

Одной из первых задач, с решением которых отлично справляются алгоритмы процедурной генерации контента, стало построение игрового уровня и расположение комнат [1]. Понятие игрового уровня и комнат широко используется в компьютерных играх самых различных жанров. Особенно актуальна задача процедурной генерации уровней стоит в играх тех жанров, в которых особенностью является то, что игроку предстоит множество раз повторно начинать игру сначала, начиная с первого уровня. Для того, чтобы этот процесс не стал монотонным, было предложено каждый раз создавать новый тип уровня, меняя в нем расположение комнат, предметов и противников. Таким образом, игрок никогда не мог быть уверенным в том, что ждет его впереди, даже после нескольких попыток.

Перед алгоритмом процедурной генерации уровня ставятся следующие задачи:

- создание уникальных наборов комнат;
- полученный уровень должен быть сбалансированным и проходимым;
- комнаты должны располагаться естественным, интуитивно-понятным способом.

Комнаты в каждой отдельной игре будут обладать своим набором свойств, которые определят ее уникальность. Среди них могут быть ее структура, внешний вид, наполнение. Однако среди всех игр у комнат можно выделить такие обобщенные черты: форма, размер, расположение на карте, связь с другими комнатами.

Каждый алгоритм процедурной генерации будет включать в себя создание набора свойств из этого списка, и далее варьироваться в зависимости от особенностей игры [2]. Разные алгоритмы позволяют создавать комнаты, которые больше или меньше подходят под заданные условия игры. Так, например, структура уровня, действие которого происходит в пещерах под землей, должна значительно отличаться от уровня, расположенного в городской среде. Таким образом, для разных целей будут уместны разные алгоритмы, которые лучше справляются с построением уровней, структура которых ближе к определенным игровым реалиям.

Метод двоичного разбиения пространства

Метод двоичного разбиения пространства основывается на структуре, называемой BSP деревом.

Двоичное разбиение пространства (binary space partitioning) – метод рекурсивного разбиения евклидова пространства в выпуклые множества и гиперплоскости. В результате объекты получают представление в виде структуры данных, называемой BSP-деревом [3].

Входными данным для алгоритма являются общий размер уровня и минимальный размер ожидаемых комнат. Вся площадь уровня принимается за один лист BSP-дерева, который случайным образом делится на две части. Эти части становятся листьями в следующем уровне дерева (см. рис. 1). Алгоритм выполняется рекурсивно для каждого нового листа до тех пор, пока его размер не окажется ниже заданного минимального порога.

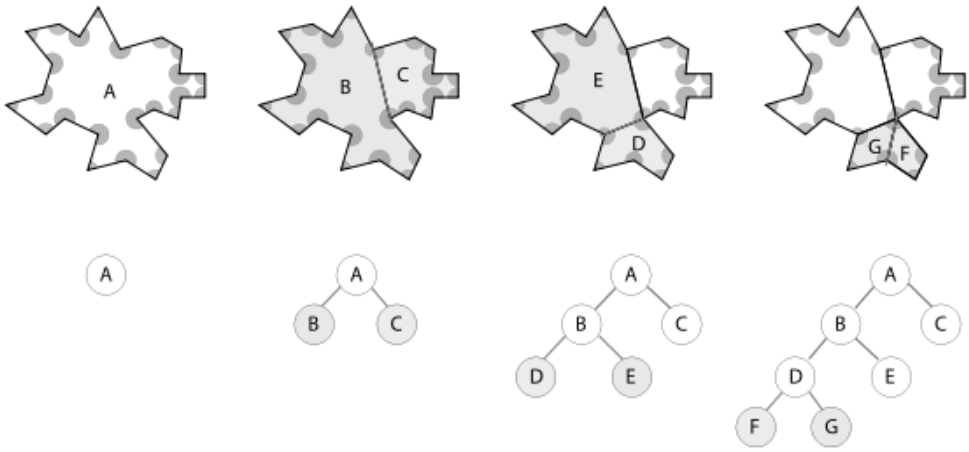


Рисунок 1 – Схема выполнения алгоритма двоичного разбиения пространства

Метод двоичного разбиения комнат является простым и удобным в реализации. Он позволяет быстро разбить заданную изначально фигуру на необходимое количество равномерно распределенных меньших фигур, не оставляя между ними пустых пространств и создавая геометрически строгие комнаты. Уровень, полученный с помощью такого алгоритма, хорошо подойдет там, где есть необходимость изобразить искусственно-созданный ландшафт. Это может быть городская среда или этаж отдельного помещения (см. рис. 2).

Базовый алгоритм не предполагает создания коридоров – переходов между комнатами. Эту задачу можно выполнить вручную, либо дополнить алгоритм генерации.

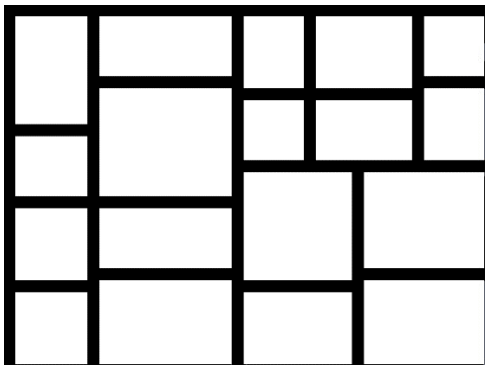


Рисунок 2 – Пример двоичного разбиения пространства

Метод «Походки пьяницы»

Метод «Походки пьяницы» (Drunkard walk) – одна из вариаций метода «Случайной походки» (Random walk). Он получил свое название за соответствующий хаотичный узор, который образуется в результате его работы.

Метод основывается на перемещении курсора, который закрашивает область комнаты, двигаясь в случайном направлении [4]. Он описывается следующими этапами:

1. Каждая точка уровня объявляется «стеной» - непроходимой областью
2. На площади уровня выбирается случайная точка – точка начала. Она отмечается как «пол» - проходимая область
3. Выбирается случайное направление движения.
4. Курсор делает шаг в выбранном направлении, его новая точка расположения отмечается как «пол»
5. Шаги 3-4 повторяются до тех пор, пока не будет закрашено заданное количество точек.

Метод «Походки пьяницы» гарантирует отсутствие изолированных пустых областей – из каждой точки комнаты можно будет добраться в любую другую точку на карте.

Для создания более естественных структур комнат, алгоритм предполагает несколько улучшений. Так, на каждое возможное направление движения выставляются весовые коэффициент, таким образом, чтобы курсор смещался к центру уровня, а не упирался в границы. Кроме того, на следующей итерации можно задать больший коэффициент для предыдущего направления – таким образом между комнатами появятся более выраженные коридоры.

Данный алгоритм хорошо передает образование естественных, природных структур. Он хорошо подходит для создания уровней в пещерах либо джунглях (см. рис. 3).

Рисунок 3 – Итог работы алгоритма «Походки пьяницы»

Клеточные автоматы

Клеточный автомат – набор клеток, образующих некоторую периодическую решетку с заданными правилами перехода, определяющими состояние клетки в следующий момент времени через состояние клеток, находящимися от нее на расстоянии не больше некоторого, в текущий момент времени. Как правило, рассматриваются автоматы, где состояние определяется самой клеткой и ближайшими соседями[5].

Алгоритм, который основывается на клеточном автомате, начинает свою работу с того, что случайным образом объявляет каждую точку уровня «стеной» либо «полом». Таким образом получается стартовое состояние клеточного автомата. Соседями каждой точки считаются 8 смежных с ней точек. После этого начинается его итерационный процесс, который определяется следующими правилами:

- точка остается стеной, если она была стеной и четыре ее соседние точки – стены;
- точка становится стеной, если она была полом, а пять ее соседей – стены.

Таким образом, после нескольких итераций, образуются комнаты.

В результате работы алгоритма могут образовываться изолированные области, их обработку необходимо проводить вручную либо отдельным алгоритмом.

Конечный результат содержит пустые области с пологими, естественно выглядящими стенами, без строгого деления на «комнаты» и «коридоры». Такого рода уровень хорошо отражает ландшафт естественной среды (см. рис. 4).

Рисунок 4 – Итог работы клеточного автомата

Вывод

Алгоритмы процедурной генерации контента позволяют создавать неограниченное количество разнообразных и качественных структур уровней и комнат. Гибкие настройки генерации, после корректировок, позволяют добиться желаемого разработчиком внешнего вида уровня. Для разных типов окружения, которое передает игровой мир, будут уместны различные алгоритмы и параметры генерации. Осведомленность и разумное применение различных методов генерации позволяют создавать высококачественное наполнение видеоигр с минимальными затратами ресурсов. Однако, обобщенных алгоритмов недостаточно для создания конкурентно-способных продуктов, и каждый алгоритм нуждается в доработках для достижения лучших результатов. Информация данной статьи может быть использована для создания более узкоспециализированных алгоритмов процедурной генерации контента.

Литература

1. Procedural Content Generation Wiki [Electronic resource]/ Интернет-ресурс. – Режим доступа : www/ URL: pcg.wikidot.com. – Загл. с экрана
2. Procedural Map Generation [Electronic resource]/ Интернет-ресурс. – Режим доступа : www/ URL: https://www.gridsagegames.com/blog/2014/06/procedural-map-generation. – Загл. с экрана.
3. How to use BSP trees to generate game maps [Electronic resource]/ Интернет-ресурс. – Режим доступа : www/ URL: https://gamedevelopment.tutsplus.com/tutorials/how-to-use-bsp-trees-to-generate-game-maps--gamedev-12268. – Загл. с экрана.
4. Drunken Master cave generation [Electronic resource]/ Интернет-ресурс. – Режим доступа : www/ URL: https://forums.roguetemple.com//index.php?topic=4128.0. – Загл. с экрана.
5. Generate random cave levels using cellular automata [Electronic resource]/ Интернет-ресурс. – Режим доступа : www/ URL: https://gamedevelopment.tutsplus.com/tutorials/generate-random-cave-levels-using-cellular-automata--gamedev-9664. – Загл. с экрана.

В.С. Вивденко, О.М. Копытова Обзор актуальных методов процедурной генерации комнат в видеоиграх. *В статье дан обзор нескольких самых актуальных методов процедурной генерации контента, которые могут быть использованы для генерации комнат и уровней в видеоиграх, обозначены их особенности и рекомендуемые области применения. Рассмотрены их недостатки и определены пути доработки.*

Ключевые слова: *разработка игр, процедурная генерация контента, уровни, BSP-деревья, клеточные автоматы*

V.S. Vivdenko, O.M. Kopytova Review of current methods for procedural generation of rooms in video games*. The article provides an overview of several of the most relevant procedural content generation methods that can be used to generate rooms and levels in video games, identifying their features and recommended uses. Considered their shortcomings and identified ways to refine.*

Keywords*: game development, procedural content generation, levels, BSP-trees, cellular automata*

УДК 004.045

Анализ специфики реализации объектного подхода в современных технологиях программирования

Воробьев Л.О., Григорьев А.В.
Донецкий национальный технический университет
Lev.Vorobjev@rambler.ru, grigorievalvl@gmail.com

Воробьев Л.О., Григорьев А.В. Анализ специфики реализации объектного подхода в современных технологиях программирования. *Статья посвящена исследование объектно-ориентированного подхода в программировании. В статье представлены примеры реализации элементов объектно-ориентированного подхода в разных языках программирования, проведен сравнительный анализ средств реализации объектно-ориентированных подхода.*

Ключевые слова: *объекты, языки программирования, паттерны проектирования.*

Введение

Исследованию вопроса объектной ориентированности разработки программного обеспечения посвящёно много научных работ [1,2,3,4,5] на протяжении нескольких десятилетий. Однако до сих пор нет точного определения объектной ориентированности программного обеспечения [6]. В данной статье сделана попытка унифицировать знания об объектном подходе с целью использования их для совершенствования методов разработки программного обеспечения.

Актуальность данной статьи заключается в необходимости разрешение противоречий между разными определениями объектно-ориентированной парадигмы программирования.

Целью работы является систематизация объектных операций, составляющих основу объектного подхода.

Новые знания, полученные в результате исследования данной статьи, могут быть использованы для повышения качества программного обеспечения путем развития технологий программирования.

Объектно-ориентированный подход

Существуют разные подходы к определению объектно-ориентированного подхода (ОО-подхода).

Согласно Лафоре, ОО-подход изолирует данные и алгоритмы из разнородных фрагментов программы.

Согласно Флоренсову [5], ОО-подход основан на динамическом определении данных.

В целом, определения ОО-подхода имеют следующие противоречия:

— неоднозначность понятия класса (классом могут быть типы, объекты, группы типов, программные модули, прототипы);

— зависимость реализации от языка программирования, например объекты Smalltalk не похожи на объекты C++. Вообще, для преобразования объекта из одного языка в другой необходимо сначала представить объект в виде набора данных.

Абстрактное ОО-проектирование в независимом от языка программирования контексте представляет собой способ организации ПО, основанный на операциях с функционально связанными объектами, которые инкапсулируют данные и алгоритмы.

Не исключено, что назначение и трактовка ОО-похода в будущем изменится непредвиденным образом благодаря прогрессу в области технологий программирования.

Технологии программирования

Технологиями реализации ОО-подхода являются парадигма объектно-ориентированного программирования и паттерны (приемы) ОО-проектирования.

Согласно Б. Страуструпу, парадигмой программирования является указание программисту, что делать, при написании программы. Например: «решите, какие процедуры нужны, используйте наилучшие алгоритмы».

Согласно Бердоносову [6] и теории ТРИЗ, парадигма программирования возникает в результате разрешения противоречий, и сохраняет некоторые признаки предшествующих парадигм.

Для реализации ОО-подхода используются парадигмы ООП и обобщенного программирования. Первая связана с определением абстрактных типов данных, а последняя — с параметризацией алгоритмов типами.

Паттерном является «решение общей задачи проектирования в конкретном контексте» [7].

В сущности, паттерны ОО-проектирования используют композицию, наследование и протокол взаимодействия объектов для решения часто возникающей задачи проектирования [7,8].

Тиражирование типов

Тиражирование типов — создание новых типов данных из образца.

В разных языках программирования использоваться разные способы тиражирование типов данных. Метаклассы используется в языках объектно-ориентированного программирования с иерархической структурой. Для прототипных объектно-ориентированных языков программирования характерно создание новых типов данных путем клонирования объектов. В функциональных языках программирования для тиражирования типов данных используется функция высшего порядка, оперирующие типами данных. Статически типизированные компилируемые языки для этих целей используют шаблоны классов и объектов.

Реализация полиморфизма типов в разных языках программирования различны. Приведем пример: создаем функцию, которая вычисляет квадрат числа. Необходимо обеспечить возможность использования созданной функции для всех видов чисел: целые, вещественные, комплексные. Реализация на C++14 (см.рис.1).

```
template <typename T, typename =
    std::enable_if_t<std::is_arithmetic<T>::value>>
inline T square(T x) { return x*x; }
```

Рисунок 1 — Реализация обобщенной функции square на C++14

На рис. 1 приведен пример обобщенного программирования с использованием **шаблонов**. Шаблоны функций и классов на современном C++ подробно рассмотрены в литературе [9] и ISO IEC 14882.

Для реализации этой функции на языке Си раньше использовались макроопределения.

Реализация на языке Haskell выглядит короче (см. рис. 2).

```
square :: Num a => a -> a
square = (^2)
```

Рисунок 2 — Запись функции square на языке Haskell

Тип функции в первой строке означает, что входом и выходом функции являются числа одного типа. На самом деле, в данной ситуации определение типа функции является необязательным, поскольку компилятор использует алгоритм Хиндли-Милнера автоматического вывода типов. Разница в размере исходного кода обусловлена необходимостью указывать типы входных и выходных значений в статически типизированных языках, а также наличием механизма вывода типов в функциональных языках.

Определение функции на рис. 2 демонстрирует тиражирование типов в функциональных языках. Параметром типа является Num a. Это любой тип данных, который имеет арифметические операции. Таким образом, эту функцию можно использовать для целых, вещественных и комплексных чисел.

Кроме того, для вычисления квадрата комплексного числа с помощью шаблонной функции на C++ необходимо явно специализировать шаблонную операцию «если арифметический тип» (см. рис. 3).

```
template <typename T>
struct is_arithmetic<std::complex<T>> : public true_type{};
```

Рисунок 3 — Расширение класса арифметических типов данных на языке C++

Однако этого можно не делать, если убрать проверку типа в шаблоне функции.

Посредством специализации шаблона на рис. 3 компилятору сообщается, что для комплексных чисел операция «если арифметический тип» должна давать положительный ответ. Аналогичная операция на языке Haskell использует ключевое слово instance, при этом необходимо определить все операторы класса типов.

В динамически типизированных языках за счет динамического определения типов функция может обрабатывать любые числа (см. рис. 4,5,6). Аналогичного результата в статически типизированных языках можно достигнуть с использованием шаблонов или вывода типов.

```
def square(x):
    return x*x;
```

Рисунок 4 — Реализация функции square на языке Python

```
def square(x) x**2 end
```

Рисунок 5 — Реализация функции square на языке Ruby

```
sub square { return $_[0] ** 2; }
```
Рисунок 6 — Реализация функции square на языке Perl

Реализация на Python (см. рис. 4) отличается от реализации на Perl наличием имен параметров и отсутствием фигурных скобок. То же самое утверждение относится к реализации на Ruby (см. рис. 5).

Динамическая типизация означает, что тип параметра x определяется в момент вызова подпрограммы. Следовательно, передавая в программу целые, вещественные, комплексные числа, функция будет работать. Однако, передавая в функцию строки или другие данные, результат работы функции будет отличаться от заложенной в нее семантики. В динамической типизации за корректность типов отвечает разработчик.

Существует также прототипная модель ООП, в которой объекты являются прототипами (см. рис. 7,8).

```
function square(x) return x^2 end
```
Рисунок 7 — Реализация функции square на языке Lua

```
Number square := method (**2)
```
Рисунок 8 — Реализация функции square на языке Io

В прототипных ОО-языках полиморфизм типов обеспечивается тем, что все числа порождаются от Number, которому был добавлен метод square. Однако, при определении нового типа, например, комплексных чисел, необходимо переопределить его операторы.

Полиморфизм типов с помощью шаблонов реализован в Java (см. рис. 9), C#, D (рис. 10,11).

```
public static <T extends Number> T square (T x) {
    return x*x;
}
```
Рисунок 9 — Реализация функции square на языке Java

```
public static T square<T> (T x) { return x*x; }
```
Рисунок 10 — Реализация функции square на языке C#

```
T square(T)(T x) { return x*x; }
```
Рисунок 11 — Реализация функции square на языке D

Компилируемые в байт-код языки (Java, C#) на самом деле используют обобщения вместо шаблонов. Это означает, что типом параметра T может быть только класс. При этом, для использования шаблонных функций с базовыми типами используются механизм упаковки переменных в обертки.

Примечательно, что в языке D для указания списка параметров шаблона используются круглые скобки. Эту особенность авторы обосновывают упрощением синтаксического анализатора [10].

Абстракции типов

Абстракция типов — это отделение интерфейса от классов от его реализации. Разные языки программирования предоставляют механизмы для этого вида абстрагирования. В качестве примера приведем общую задачу создания абстрактного интерфейса (см. рис. 12).

```
struct ISimple {
    virtual int touch() = 0;
}
class Simple : public ISimple {
    int touch() override { return 0; }
}
```
Рисунок 12 — Отделение интерфейса от реализации средствами C++

Операция абстрагирования данных является составной частью многих паттернов проектирования. Она используется для скрытия реализации определенного интерфейса вот клиентской части. В данном примере простой интерфейс содержит одну операцию. Реализация интерфейсов подразумевает определение этой операции.

На языке Haskell для выделения абстрактных интерфейсов используются классы типов (см. рис. 13).

```
class Simple a where
    touch :: Int
instance Simple Int where
    touch = 0
```
Рисунок 13 — Отделение интерфейса от реализации на Haskell

В Haskell классы типов являются одновременно средством абстрагирования и вывода типов.
На языке Perl классами являются программные модули (см. рис. 14).

```
package ISimple;
sub new { return bless { }, shift; }
package Simple;
@ISA = qw/ISimple/;
sub touch { return 1; }
1;
```
Рисунок 14 — Определение иерархии классов на Perl

В Perl методы класса и объекта определяются, как функции модуля. Поскольку в Perl нет абстрактных методов (функций без реализации нет), реализация интерфейса классом только подразумевается.
На Ruby, для реализации классов используются соответствующие выразительные средства (см. рис. 15)

```
require 'interface'
ISimple = interface{ required_methods :touch }
class Simple
    def touch 0 end
    implements ISimple
end
```
Рисунок 15 — Отделение интерфейса от реализации на Ruby

На языке D реализация абстракции типов похожа на C++, но проще (см. рис. 16).

```
interface ISimple {
    int touch();
}
class Simple : ISimple {
    void touch() { return 0; }
}
```
Рисунок 16 — Отделение интерфейса от реализации на языке D

Отличие состоит в отсутствии необходимости указания чистоты и виртуальности методов интерфейса.

Создание объектов

ОО-подход — описание программы в терминах объектов и операций над ними. Для создания объектов определенного типа в разных языках программирования используются разные инструкции (см. рис. 17,18,19,20).

```
auto object = new Object;
```
Рисунок 17 — Создание объекта C++

```
do object <- Object; object
```
Рисунок 18 —Определение объекта Haskell

```
my $object = Object->new;
```
Рисунок 19 — Создание объекта Perl

```
object = Object.new
```
Рисунок 20 — Создание объекта Ruby

В языке Haskell, в отличие от императивных языков, отсутствуют переменные. Поэтому определение

экземпляра определенного типа создает значение этого типа, которое передается другой функции. Это значение можно считать объектом класса типов, к которому принадлежит тип Object создаваемого значения (см. рис. 18).

В языке Perl (см. рис. 19) объекты создаются с помощью определенной пользователем функции new (см. рис.14). Эта функция содержит оператор bless, который связывает переменную с именем модуля. Таким образом переменная становится объектом.

В языке Ruby (см. рис. 20) операция new определена в недрах стандартной библиотеки.

В прототипных ОО-языках объекты создаются путем клонирования (см. рис. 21).

```
object := Object clone
```

Рисунок 21 — Создание объекта Io

При клонировании новый объект получает копию старого объекта со всеми методами и данными. Аналогичный механизм реализован в Lua (см. рис. 21), где Obejct — определенный пользователем тип.

```
object = Object:new()
```

Рисунок 21 — Создание объекта Lua

Кроме того, в языке Smalltalk объект класса создается путем посылки сообщения классу-объекту.

С созданием объектов связаны порождающие паттерны проектирования.

В качестве примера задачи проектирования: интерфейс, создающий объекты неизвестного класса. Существует несколько способов решения этой задачи.

Для обеспечения модульности используются паттерны ОО проектирования, которые разделяют процесс создания объекта с целью гибкой замены его частей. Например, фабричный метод отделяет определение типа объекта от клиента, так что клиент может создать объект без указания его типа. Паттерн строитель позволяет также отделить от клиента внутреннюю структуру объекта.

Порождающие паттерны проектирования разделяют процедуру создания объекта на части, которыми управляет клиент и реализация.

Взаимодействие объектов

В языке Smalltalk объекты взаимодействуют между собой путем посылки сообщений. Другие языки программирования по-разному реализуют панели посылки сообщений между объектами. В частности, путем вызова методов, как подпрограмм. Подробнее об этом описано в литературе [2].

Выводы

Языки программирования по объектной ориентированности делятся на поддерживающий подход и допускающие его. На примере тиражирования типов абстракции типов и создания объектов была продемонстрирована возможность разных языков программирования реализовывать объектный подход. В результате было получено, что объектный подход можно организовать на любом языке программирования.

Таблица сравнительной характеристики языков программирования, отражающая набор элементарных операции над объектами, не может быть объективной, поскольку отражает лишь степень владения разработчиком языка программирования. Если бы такая таблица содержала объективную информацию, то все ячейки были бы заполнены. Поскольку универсальные языки программирования предоставляют возможность любой операции над объектами.

Литература

1. Бадд Т. Объектно-ориентированное программирование в действии — СПб.: Питер, 1997.
2. Кирютенко Ю.А., Савельев В.А. Объектно-ориентированное программирование. Язык Smalltalk — М.: Вузовская книга, 2006. — 328 с.: ил.
3. Орлов, С. А. Программная инженерия. Учебник для вузов. 5-е издание обновленное и дополненное. Стандарт третьего поколения, — СПб.: Питер, 2016 — 640 с.: ил.
4. Логанов, С. В. Ещё раз о принципах применения наследования — Нижний новгород, 2017.
5. Флоренсов, А. Н. О внутренних процессах объектно-ориентированного программирования // Научные труды SWorld — Омск : ОмГТУ, 2015.
6. Бердоносов, В. Д. Применение ТРИЗ-эволюционного подхода к исследованию объектно-ориентированных языков программирования // Бердоносов, В. Д., Животова, А. А. — Комсомольск-на-Амуре : ФГБОУ ВПО «КиАГТУ», 2014.
7. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования // Гамма Э.,

Хелм Р., Джонсон Р., Влиссидес Дж. — СПб.: Питер, 2015 — 386 с.: ил.

8. Нестерук, Д. Design Patterns in Modern C++: Reusable Approaches for Object-Oriented Software Design, 2018. Apress. URL: https://github.com/Apress/design-patterns-in-modern-cpp

9. Вандевурд, Д., Джонаттис, Н. М., Грегор, Д. Шаблоны C++. Справочник разработчика, 2-е изд. : Пер. с англ. — СПб.: ООО „Альфа-книга“, 2018. — 848 с.: ил.

10. Александреску А. Язык программирования D. — Пер. с англ. — СПб.: Символ-Плюс, 2012.

Воробьев Л.О., Григорьев А.В. Анализ специфики реализации объектного подхода в современных технологиях программирования. *Статья посвящена исследование объектно-ориентированного подхода в программировании. В статье представлены примеры реализации элементов объектно-ориентированного подхода в разных языках программирования, проведен сравнительный анализ средств реализации объектно-ориентированных подхода.*

Ключевые слова: *объекты, языки программирования, паттерны проектирования.*

Vorobyev Lev, Grigoriev Alexander Analysis of the specifics of the implementation of the object approach in modern programming technologies. *The article is devoted to the study of object-oriented approach in programming. The article presents examples of the implementation of the elements of an object-oriented approach in different programming languages, and a comparative analysis of the means of implementing an object-oriented approach.*

Keywords: *objects, programming languages, design patterns.*

УДК 004.4

Голосовые мессенджеры. Обзор существующих протоколов

Вязмин В.И., Чернышова А. В.
Донецкий национальный технический университет
testerreality@gmail.com, chernyshova.alla@rambler.ru

Вязмин В.И., Чернышова А.В. Голосовые мессенджеры. Обзор существующих протоколов. *В статье рассмотрены популярные на данный момент времени голосовые мессенджеры. Произведен обзор существующих протоколов передачи аудиоданных. Подробно рассмотрен протокол XMPP и его возможность передачи аудиоданных. Определены слабые и подчеркнуты сильные стороны протокола XMPP.*

Ключевые слова: *XMPP, аудиоданные, шифрование, XML, мессенджеры.*

Обзор голосовых мессенджеров

Голосовые мессенджеры [1] приобрели свою популярность совсем недавно. Развитие данной отрасли берет своё начало в момент, когда средняя скорость интернета по всему миру стала достаточно комфортной для использования данных приложений даже на смартфонах. Мессенджер - это программа, мобильное приложение или веб-сервис для мгновенного обмена сообщениями, в данном случае, голосовыми. Данная отрасль начала развиваться достаточно бурно и эффективно, что привело к созданию по меньшей мере сотен, а то и тысячи мессенджеров. Однако, на рынке программных продуктов такого типа смогли показать своё превосходство лишь единицы. Одним из наиболее популярных голосовых мессенджеров в настоящее время является WhatsApp [2].

WhatsApp - бесплатная система мгновенного обмена сообщениями (текстовыми, голосовыми и видео) для мобильных и иных платформ. Позволяет пересылать текстовые сообщения, изображения, видео и аудио через Интернет. Для передачи данных, WhatsApp использует модифицированный протокол Extensible Messaging and Presence Protocol (XMPP, ранее известный как Jabber) [3]. Мультимедиа-сообщения отправляются путём загрузки изображения, звука или видео на HTTP-сервер [4] и передачей гиперссылки на объект вместе с закодированным в Base64 [5] уменьшенным вариантом изображения. В данный момент WhatsApp использует сквозное шифрование (end-to-end) [6]. Данное шифрование относится ко всем типам пересылаемых данных, будь то текстовое, фото, видео или голосовое сообщение. Произвести расшифровку данных сообщений может только получатель, содержимое недоступно даже серверам WhatsApp. В реализации используются алгоритмы ECDH [7] на Curve25519 [8], AES-256 [9], AES-GCM [10]. Два пользователя имеют возможность сверить ключи шифрования путём сканирования QR кода или сравнения 60-значного числа, что позволит исключить атаки класса man-in-the-middle.

Следующим, не менее известным мессенджером является Skype [11].

Skype — бесплатное проприетарное программное обеспечение с закрытым исходным кодом, которое обеспечивает текстовую, голосовую и видеосвязь через интернет между компьютерами (IP-телефония). Опционально приложение использует технологии пириновых сетей. Главной отличительной чертой среди других подобных программ являлось то, что для передачи данных Skype изначально использовал децентрализованную P2P-архитектуру [12]. Каталог пользователей Skype был распределён по компьютерам пользователей сети Skype, что давало возможность легко масштабироваться до чрезвычайно больших размеров без затрат на расширение серверной части. Протокол Skype является закрытым и недокументированным, используется только оригинальным ПО Skype. С июня 2014 года оригинальный протокол объявлен устаревшим, вместо него Skype использует протокол MSNP24 [13]; с августа 2014 года оригинальный протокол был отключен на серверах. Пользователь может разговаривать как с одним, так и с несколькими пользователями одновременно. Skype использует кодеки (алгоритмы сжатия данных) SILK, G.729 и G.711 [14], и при достаточной скорости интернет-соединения (30—60 кбит/с) ничем не уступает привычным телефонным разговорам. При установке соединения между ПК аудиоданные шифровались при помощи алгоритма AES-256, для передачи ключа которого, в свою очередь, использовался 1024-битный ключ алгоритма RSA [15]. Открытые ключи пользователей сертифицируются центральным сервером Skype при входе в систему с использованием 1536- или 2048-битных сертификатов RSA. Не смотря на все отличия данного мессенджера, в последние годы он стал терять свою популярность, и большинство пользователей предпочитают другие мессенджеры, например, такие, как Discord [16].

Discord — бесплатный мессенджер с поддержкой VoIP и видеоконференций, изначально ориентированный для пользователей компьютерных игр. Discord использует звуковой кодек Opus [17], который имеет низкую задержку и предназначен для сжатия речи. Для передачи данных, Discord использует модифицированный протокол User Datagram Protocol (UDP) [18]. Отправленные голосовые дынные должны быть закодированы с помощью Opus, используя два канала и частоту дискретизации 48 кГц. Голосовые данные отправляются с использованием заголовка RTP [19], за которым следуют зашифрованные звуковые данные Opus. Голосовое шифрование использует ключ, который является кодом операции на сервере с 24-байтным заголовком (используется как однократно используемое число, с добавлением 12 нулевых байтов), алгоритм шифрования неизвестен, предположительно AES.

Аудиоданные в протоколе XMPP

XMPP (Extensible Messaging and Presence Protocol) — основанный на XML, открытый протокол для мгновенного обмена сообщениями, который поддерживает передачу голоса, видео и файлов. XMPP является децентрализованной и расширяемой системой. XMPP напоминает другие протоколы прикладного уровня, например, SMTP [20]. В данной архитектуре каждый клиент обладает уникальным именем и обменивается информацией с другими клиентами через сервер. При этом клиенты включают реализации клиентской части протокола, в то время как сервер выполняет функции маршрутизатора. XMPP представляет собой относительно простой протокол для передачи сообщений через сокеты TCP [21]. Клиент и сервер взаимодействуют асинхронно путем передачи станс XML (XML stanzas) [22] внутри XML-потоков. XML-потоки выполняют функции контейнеров, инкапсулирующих обмен XML-данными между двумя объектами, в то время как стансы являются дискретными информационными единицами. В XMPP XML-стансы используются для передачи как пользовательских сообщений, так и информации о статусе присутствия. Пример взаимодействия двух клиентов представлен на рисунке 1. Следует отметить, что в переговорах должен принимать участие как минимум один сервер. На рисунке 1 левый клиент – это инициирующая сторона (является инициатором XMPP-обмена). Данный XML-поток использует атрибут «to» для распознавания получающего домена (а также для определения пространства имен XML). Принимающий клиент справа получает XML-поток и отвечает также XML-потоком (используя атрибут «from»). Данный этап позволяет проводить ряд различных переговоров, в частности, об аутентификации и шифровании.

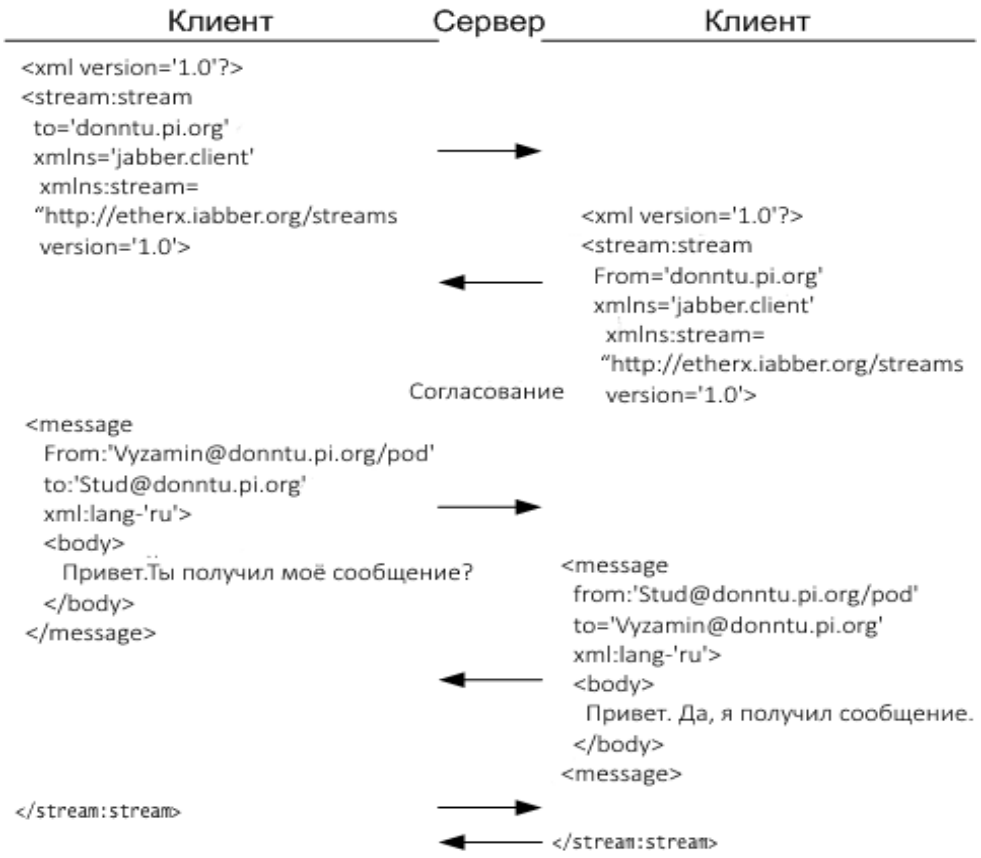


Рисунок 1 — Пример (упрощенный) обмена XMPP-сообщениями

Для передачи аудиоданных протоколу XMPP необходимо дополнение. Данное дополнение получило имя Jingle [23]. Jingle — это дополнение к протоколу XMPP, позволяющее передавать между двумя клиентами аудио- и видеоданные, поддерживает только P2P [24] соединения. Он был разработан компанией Google и XMPP Standards Foundation. Официальное название стандарта — XEP-0166. Поддержка сервера не требуется для Jingle, поскольку клиенты напрямую взаимодействуют и используют только протокол XMPP для ведения переговоров. Следующим необходимым шагом является наличие STUN [25]. STUN - протокол, позволяющий обходить ограничения, связанные с работой за NAT [26] (от англ. Network Address Translation — «преобразование сетевых адресов»). STUN нужен для работы Jingle, когда собеседники находятся за NAT и не имеют прямого IP-адреса. В ряде случаев чтобы работала передача файлов, а также общение голосом и видео нужно указывать STUN-сервер. Передача аудиоданных происходит по протоколу RTP (Real-time Transport Protocol). Схематическое представление передачи аудиоданных представлено на рисунке 2.

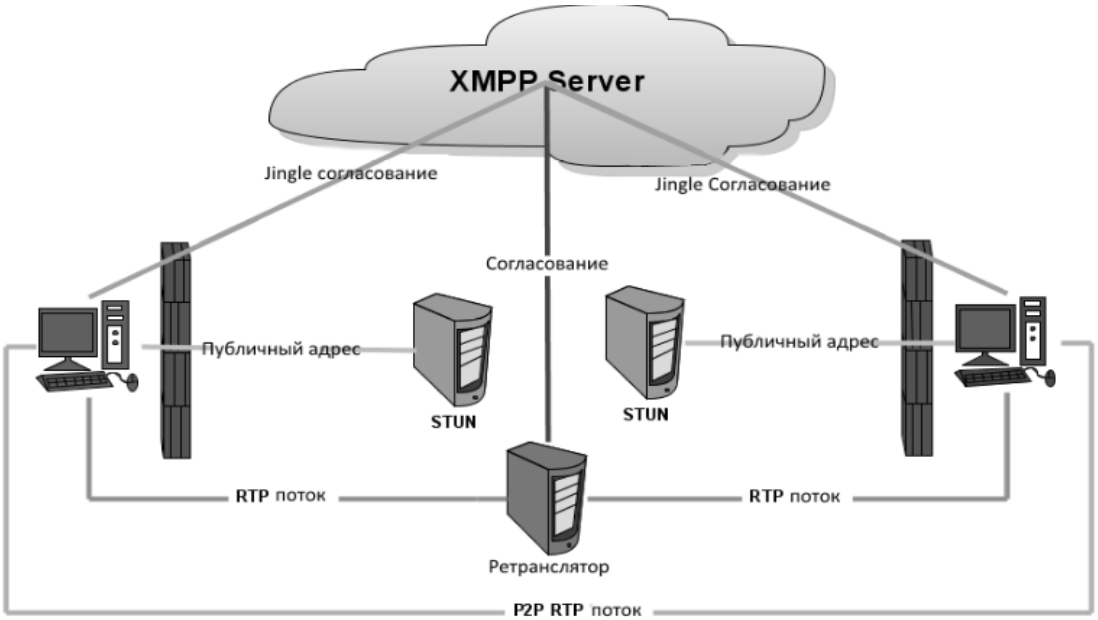


Рисунок 2 — Схематическое представление передачи аудиоданных при помощи Jingle

После того, как клиент запустил Jingle-сеанс, XMPP сервер получает список пользователей для соединения. Затем каждый клиент получает общедоступный адрес при помощи STUN-сервера и просит XMPP сервер выдать ему ip-адрес клиента, с которым необходимо установить соединение. Если данная операция невозможна, скорее всего, пользователь не в сети. Всё это время сервер посылает запрос ECHO клиентам, чтобы убедиться, что они доступны. После того, как маршрут согласован, начинается P2P связь.

Достоинства и недостатки XMPP

К достоинствам протокола XMPP можно отнести децентрализацию, поскольку архитектура сети подобна электронной почте; кто угодно может запустить свой собственный XMPP-сервер, и нет какого-либо центрального сервера. Также, достоинством является и богатая история, которая позволила приобрести множество дополнений к стандартам XMPP. Следующим достоинством является безопасность. XMPP серверы могут быть изолированы от публичных сетей XMPP (например, во внутренней сети компании) и хорошо защищены встроенными в ядро XMPP спецификациями. Одним из важнейших достоинств данного протокола является гибкость. Настраиваемая функциональность может быть надстроена поверх XMPP.

Недостатками можно считать избыточность передаваемой информации: в большинстве случаев более 70% трафика между серверами XMPP составляют сообщения о присутствии, из которых 60% являются избыточными. XMPP на данный момент создаёт избыточный трафик при доставке сообщений о присутствии (то есть «статус-сообщений») нескольким пользователям.

Шифрование XMPP

Для шифрования данных в протоколе XMPP можно использовать OTR [27]. OTR (Off-the-Record Messaging) является криптографическим протоколом для систем мгновенного обмена сообщениями. Протокол OTR разрабатывался для того, чтобы обеспечить приватность переговоров, аналогичную переговорам без использования средств телекоммуникаций. Для передачи сообщений участникам необходимо установить общий секретный ключ. Для этого используется протокол аутентифицированного распределения ключей, основанный

на протоколе Диффи — Хеллмана [28]. Для шифрования сообщений используется алгоритм AES в режиме счётчика. Использование построенного таким образом поточного шифра обеспечивает спорное шифрование [29]. Иными словами, любой, кто перехватит сообщение, сможет изменить любые биты в сообщении на своё усмотрение. Если сообщение стало известно, его можно изменить на любое другое сообщение такой же длины. Поскольку присутствует спорное шифрование, участники протокола OTR могут утверждать, что любое из переданных сообщений было изменено третьей стороной. С точки зрения безопасности, это является не самым рациональным решением, однако, такое решение позволят скрыть источники, и позволяет отправителям отрицать свою причастность к отправленному сообщению.

Выводы

В наше время голосовые мессенджеры приобретают широкую популярность. В связи с этим, становится актуальным вопрос программной реализации подобных мессенджеров. Ознакомившись с протоколами, которые позволяют передавать аудиоданные, а именно, исследовав протокол XMPP, можно сделать вывод, что именно этот протокол идеально подходит для разработки небольшого голосового мессенджера. Структура данного протокола позволяет совершать передачу аудиоданных а также их шифрование. В дальнейшем планируется разработать небольшой голосовой мессенджер с использование протокола XMPP.

Литература

1. Обзор мессенджеров. Лучшие и популярные интернет мессенджеры // VoIPOFfice [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://www.voipoffice.ru/tags/messendzhery. - Загл. с экрана.
2. WhatsApp // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://wikipedia.green/WhatsApp. - Загл. с экрана.
3. XMPP (Extensible Messaging and Presence Protocol) // Национальная библиотека им. Н. Э. Баумана [Электронный ресурс] – Электрон. дан. - 2017. – Режим доступа: https://ru.bmstu.wiki/XMPP_(Extensible_Messaging_and_Presence_Protocol). - Загл. с экрана.
4. HTTP // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/HTTP. - Загл. с экрана.
5. SGVsbG8gd29ybGQh или история base64 / AlexLeonov // habr [Электронный ресурс] – Электрон. дан. - 2010. – Режим доступа: https://habr.com/post/88077. - Загл. с экрана.
6. Сквозное шифрование // Википедия [Электронный ресурс] – Электрон. дан. - 2010. – Режим доступа: https://ru.wikipedia.org/wiki/Сквозное_шифрование. - Загл. с экрана.
7. Протокол Диффи — Хеллмана на эллиптических кривых // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Протокол_Диффи_—_Хеллмана_на_эллиптических_кривых. - Загл. с экрана.
8. Curve25519, EdDSA и Poly1305: Три обделенных вниманием криптопримитива / Scratch // habr [Электронный ресурс] – Электрон. дан. - 2015. – Режим доступа: https://habr.com/post/247873. - Загл. с экрана.
9. Advanced Encryption Standard // Википедия [Электронный ресурс] – Электрон. дан. - 2014. – Режим доступа: https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard. - Загл. с экрана.
10. Galois/Counter Mode // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Galois/Counter_Mode. - Загл. с экрана.
11. Skype // Национальная библиотека им. Н. Э. Баумана [Электронный ресурс] – Электрон. дан. - 2017. – Режим доступа: https://ru.bmstu.wiki/Skype. - Загл. с экрана.
12. Одноранговая сеть // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Одноранговая_сеть. - Загл. с экрана.
13. Microsoft Notification Protocol // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Microsoft_Notification_Protocol. - Загл. с экрана.
14. Технология интернет-телефонии в программе Skype // Боровское исследовательское учреждение [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://bourabai.kz/mmt/skype.htm. - Загл. с экрана.
15. RSA // Университет ИТМО [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://neerc.ifmo.ru/wiki/index.php?title=RSA. - Загл. с экрана.
16. Discord // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Discord. - Загл. с экрана.
17. Основы кодирования аудио с потерями. Тестирование бета-версии Opus 1.3 / Zeben // habr [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://habr.com/post/346532. - Загл. с экрана.
18. UDP // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/UDP. - Загл. с экрана.
19. Протокол передачи видео- и аудиоинформации в реальном масштабе времени // Боровское исследовательское учреждение [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://bourabai.kz/mmt/rtp.htm. - Загл. с экрана.

20. SMTP // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/SMTP. - Загл. с экрана.
21. TCP протокол / Кунегин С. В. // kunegin [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://kunegin.com/ref1/net_prot/tcpprot.htm. - Загл. с экрана
22. Станс // Хранилище знаний о Jabber [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://wiki.jrudevels.org/Stanza. - Загл. с экрана
23. Jingle (протокол) // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Jingle_(протокол). - Загл. с экрана.
24. Одноранговая сеть // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Одноранговая_сеть. - Загл. с экрана.
25. STUN // Хранилище знаний о Jabber [Электронный ресурс] – Электрон. дан. - 2011. – Режим доступа: http://jawiki.ru/STUN. - Загл. с экрана
26. NAT // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/NAT. - Загл. с экрана.
27. Off-the-Record Messaging // Википедия [Электронный ресурс] – Электрон. дан. - 2013. – Режим доступа: https://ru.wikipedia.org/wiki/Off-the-Record_Messaging. - Загл. с экрана.
28. Протокол Диффи — Хеллмана // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Протокол_Диффи_—_Хеллмана. - Загл. с экрана.
29. Отрицаемое шифрование - оружие бесправного // Компьютерные вести [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: https://www.kv.by/content/otritsaemoe-shifrovanie-oruzhie-bespravnogo. - Загл. с экрана.

Вязмин В.И., Чернышова А.В. Голосовые мессенджеры. Обзор существующих протоколов. *В статье рассмотрены популярные голосовые мессенджеры, протоколы, которые позволяют передавать аудиоданные в реальном времени. Определены слабые стороны и подчеркнуты сильные стороны протокола XMPP.*

Ключевые слова: *XMPP, аудиоданные, шифрование, XML, мессенджеры.*

Vyazmin V. Chernyshova A. Voice messengers. Review of cases of protocols. The article describes the popular voice messengers, protocols that allow you to transfer audio data in real time. Certain weaknesses and strengths of the XMPP protocol.

Keywords*: XMPP, audio, encryption, XML, messengers.*

УДК 004.934.1

Речевое управление системой оценивания ментальности студентов

Гончаров К.Д., Федяев О. И.
Донецкий национальный технический университет
kriogen0501@gmail.com, fedyaev@donntu.org

Гончаров К.Д., Федяев О.И. Речевое управление системой оценивания ментальности студентов. *Для студентов со слабыми навыками работы с периферийными устройствами управления компьютером разработан канал речевого управления программой построения менталитета студента. Ментальность студента определяется психологическим тестированием в течение длительного диалога с компьютером. Диалог на основе речевых команд повышает интеллектуальность человеко-компьютерного взаимодействия, что облегчает процесс тестирования студентов.*

Ключевые слова: *ментальность студента, психологическое тестирование, распознавание речи, речевой интерфейс*

Введение

Вопросы анализа и управления образовательными процессами в Вузах всегда являются актуальными, так как от них зависит качество подготовки кадров. Эти вопросы должны решаться регулярно и оперативно. Такая возможность позволит обеспечить качественное функционирование системы подготовки кадров, которая относится к классу сложных социальных систем. Учитывая слабую формализацию таких систем, очень трудно использовать классические методы математики. Слабо структурированные задачи, которые необходимо решать при анализе и управлении системой обучения, могут успешно решаться с помощью современных информационных технологий, использующие последние достижения в области нечеткой математики, искусственного интеллекта, психофизиологии.

Оценивание качества подготовки студентов является важной задачей для кафедры и университета в целом. Для этого применяются традиционные подходы в виде экзаменов, контрольных работ. Эти формы дают возможность оценить качество после окончания обучения студентов по отдельным курсам. Однако, в некоторых случаях представляет интерес спрогнозировать перед изучением дисциплины уровень компетенции в виде оценки, которую получат студенты на экзаменах. Кроме того, студентам интересно узнать какие разделы дисциплины могут вызвать трудность при изучении и т.д. Полученный такой прогноз, может использоваться студентами и преподавателями для правильного планирования и коррекции процесса изучения данной дисциплины.

Существующие подходы к оценке качества обучения основываются на статистических методах, которые не дают оценку динамических факторов влияния на учебный процесс.

Психологическое тестирование трудоемкий и долгий процесс. Тестируемые могут иметь разные уровни практического владения тактильно-сенсорными средствами управления компьютером (клавиатура, мышь).

Процесс обучения одинаков для всех студентов. Однако все студенты имеют разные способности к обучению и, в результате, разную успеваемость. Учитывая широкий охват испытуемых, было решено упростить управление и встроить в систему возможность голосового управления.

Целью разработки является система построения ментального портрета студента при помощи методов психологического оценивания, с целью последующего прогнозирования успеваемости и востребованности на рынке труда.

Назначение и структура ментального портрета студента

Одно из важнейших звеньев рынка труда является система профессионального обучения. Основной функцией процесса обучения студентов является передача профессиональных знаний и выработка умений у будущих специалистов решать определённые производственные задачи. В данной работе особое внимание уделяется студенту как личности и его месту в процессе обучения. Студенты как объекты обучения индивидуальны по мотивации и способностям к обучению. Возникает задача определения всех факторов, влияющих на качество усвоения студентами знаний, и степень их влияния. Поскольку каждый студент является, прежде всего, личностью, то необходимо в первую очередь анализировать его личностные характеристики [Федяев].

Для всестороннего анализа личности были выделены следующие типы факторов: мотивация студента, интеллектуальные способности студента, вычислительные способности, психологические особенности, тип темперамента, творческие способности, социальная приспособленность, физическое состояние и жилищные условия, влияющие на обучение. Выявление и анализ этих факторов позволит охарактеризовать личность обучаемого с разных сторон и выявить наиболее важные ментальные особенности, влияющие на успешность обучения.

Была разработана специальная методика, позволяющая анализировать психологические, эмоциональные, природные и физические особенности студента. Методика была основана на классических психофизиологических тестах, обладающих универсальностью, а также сравнительной легкостью для их реального применения. Эти методики в совокупности образуют систему, благодаря которой определяется ментальный портрет студента.

Тестирование личности основывалось на следующих общеизвестных тестах:

- тест Айзенка на определение уровня интеллекта (в объёме 40 заданий);
- тест Айзенка на определение типа темперамента (в объёме 57 вопросов с вариантами ответов);
- тест Гилфорда на определение социального интеллекта (4 субтеста и в каждом от 12 до 15 ситуаций);
- тест Гречикова на определение уровня мотивации (в объёме 23 вопросов с вариантами ответов);
- тест Торренса на определение уровня креативности (в объёме 6 картинок для завершения фигур);
- тест Айзенка на определение специальных (вычислительных) способностей (в объёме 50 заданий).

Такие факторы ментального портрета, как умение студента работать в команде, его жилищные условия и состояние здоровья, определяются путём анкетирования студентов.

После прохождения всех опросов и тестов будет определён ментальный (многопрофильный) портрет студента, который можно будет использовать при разработке модели передачи знаний.

Разработка многофункционального интерфейса системы

Для выявления показателей ментальности студентов была разработана программная система, реализующая методы психологического тестирования и предоставляющая возможность студентам в диалоге с компьютером проходить все тесты (см. рис. 1). Ввиду накладываемых ограничений на время выполнения отдельных тестов, широкого охвата испытуемых студентов и необходимости снижения влияния внешних факторов на результаты, таких как усталость от рутинных действий, было решено предоставить испытуемым более естественный для них речевой интерфейс для управления процессом тестирования.

В системе предусмотрено несколько основных модулей:

- модуль распознавания речевых команд;
- модуль тестирования;
- модуль обработки результатов тестирования;
- модуль визуализации тестирования.

На объектно-ориентированном языке программирования Java была описана функциональность прототипа системы. Средством хранения тестов и ключей к ним выступают файлы в формате xml. Прототип программной системы предоставляет возможность испытуемым не только прохождение этих тестов, но и получения результирующей оценки тестирования.

Речевым интерфейс работает в фоновом режиме и воспринимает команды на русском языке.

Включение речевого интерфейса в стандартный интерфейс системы

В качестве инструмента разработки движка для компьютерного распознавания речи использовался Sphinx4, который, написан на объектно-ориентированном языке Java и работает на основе скрытых Марковских моделей. Предложенные движком алгоритмы хорошо себя зарекомендовали для распознавания коротких фраз (см. рис. 4) или изолированных слов из небольшого словаря.

Следуя принципу декомпозиции Sphinx состоит из нескольких моделей, отвечающих за его работу.

В инструментальной среде Sphinx акустико-лингвистические параметры распознаваемых речевых команд описаны следующими моделями:

- акустическая модель;
- фонетическая модель;
- языковая модель.

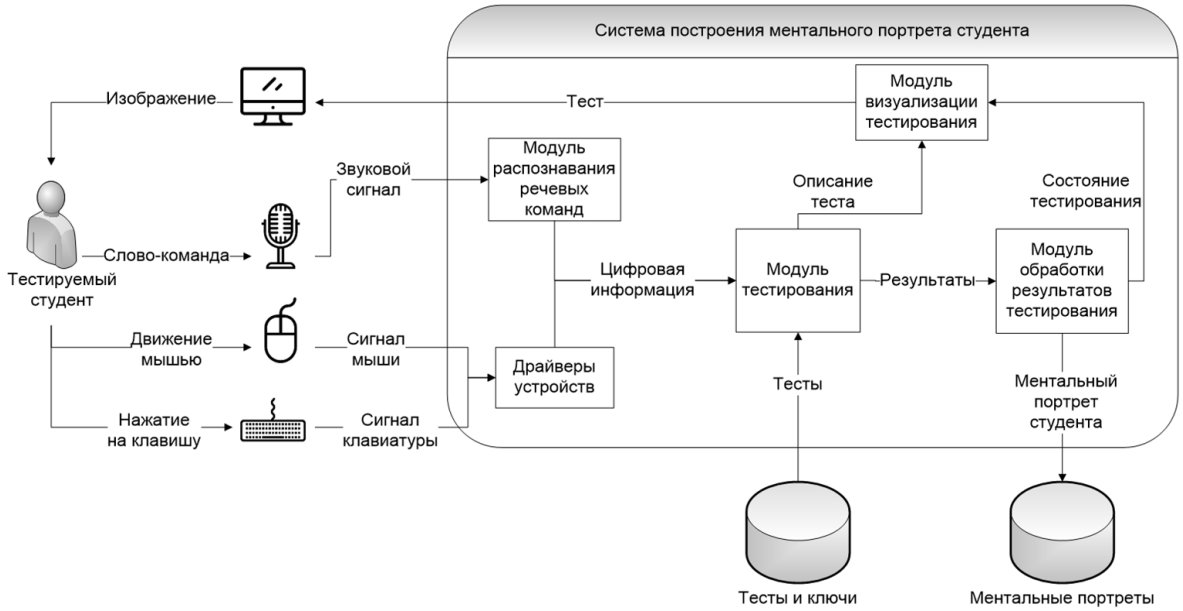


Рисунок 1 — Многофункциональный интерфейс и структура системы построения ментального портрета студента

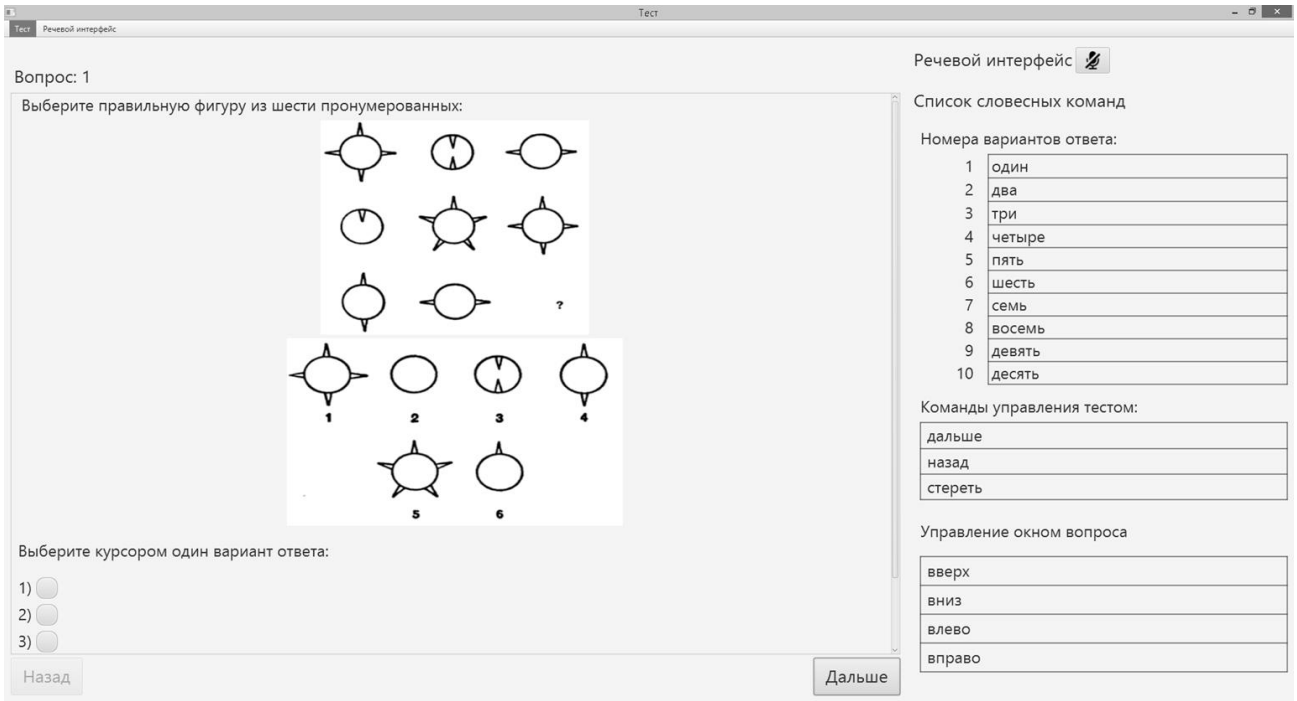


Рисунок 2 — Окно диалога при решении тестовой задачи с вводом ответа голосом или координатно-тактильными устройствами (мышью, клавиатурой)

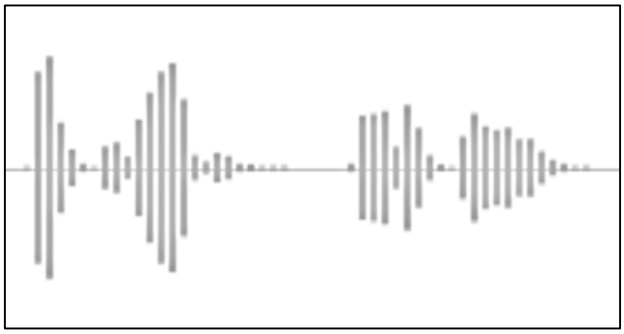


Рисунок 3 — Пример амплитудно-временного отображения речевой команды “Ответ четыре”

Акустические свойства звуковых детекторов представлены акустической моделью. Средством отображения слов на звуки выступает фонетическая модель. Языковая модель при работе движка повышает точность распознавания благодаря синтаксическим конструкциям распознаваемых фраз. В разработанном модуле распознавания речевых команд (см. рис. 1) используются два компонента языковой модели. Первый компонент - базовая статистическая языковая модель, содержащая вероятности слов и словосочетаний языка в целом. В другом компоненте языковой модели описывается грамматика.

Словарь команд речевого интерфейса для системы содержит двадцать команд, специально подобранных с учетом специфики диалога с компьютером при тестировании. Поэтому речевое управление не предусматривает ввод произвольных ответов на тесты речью. В словарь вошли команды, управляющие выбором ответа при прохождении теста и команды, позволяющие управлять процессом тестирования и окном отображения тестов. Команды были поделены на группы, где каждой группе соответствует слово-команда (см. рис. 4). В качестве средства группировки использовалась грамматика, реализованная в формате JSGF (Java Speech Grammar Format).

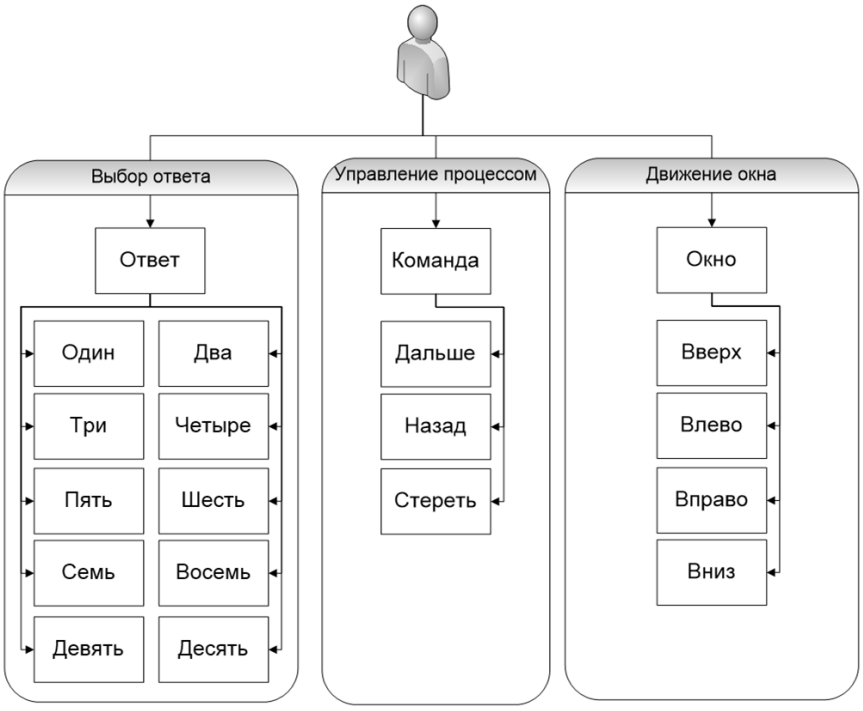


Рисунок 4 – Группы речевых команд управления системой

Было проведено тестирование частоты ошибок распознавания (в системе Sphinx частота ошибок обозначена именем WER), которое вычислялось по формуле (1).

$$WER = (I + D + S) / N,$$

где N – длина тестируемого текста из команд; I - количество вставленных слов; D - количество удаленных слов; S – количество замещенных слов. WER обычно измеряется в процентах.

В эксперименте, состоящим из проверки распознавания текста длиной 30 команд получено значение частоты ошибок равное 27%, то есть точность распознавания составила приблизительно 73%.

Выводы

В работе разработан прототип системы построения ментального портрета студента на основе психофизиологического тестирования личности. Помимо стандартных тактильно-сенсорных устройств управления компьютером (клавиатура, мышь) прототип системы предоставляет пользователям канал речевого управления, способный распознавать ответы на предоставляемые тесты и другие команды речью с точностью приблизительно 73%.

Дальнейшее развитие системы предполагает улучшение системы тестирования, добавление новых методов психологической оценки. В основе некоторых методов лежит ограничение по времени, поэтому необходимо выполнить эти требования. Также необходимо повысить точность распознавания и уменьшить объем памяти занимаемой системой. В существующей прототипе системы используется базовая версия русскоязычного словаря, предоставленного системой Sphinx, содержащая приблизительно полмиллиона слов. Поэтому имеет смысл уменьшить объем словаря, удалив из него все слова, не связанные со словарем выбора

ответов и управления процессом тестирования.

Проблему малой точности распознавания можно частично решить, увеличив обучающее множество. Sphinx предоставляет возможность постоянного обучения системы распознавания, наполняя её новыми аудиозаписями слов различных дикторов.

Литература

1. Калинин А.М., Тарасов В.Г. Использование данных электронного обучения для оценки и прогнозирования результатов. V Международная научно-практическая конференция «Электронное обучение в непрерывном образовании, ЭОНО-2018» (Россия, Ульяновск, 18-20 апреля 2018 г.): сборник научных трудов. – Ульяновск: УлГТУ, 2018. - С. 493-501.
2. Тимофеев О.Г., Хмелевская Т.А., Горбачёв И.В., Гадалина Н.Н. Анализ обучения студентов ИДДО по курсам, направлениям, ступеням. V Международная научно-практическая конференция «Электронное обучение в непрерывном образовании, ЭОНО-2018» (Россия, Ульяновск, 18-20 апреля 2018 г.): сборник научных трудов. – Ульяновск: УлГТУ, 2018. - С. 586-592.
3. Федяев, О.И. Формирование зависимости остаточных знаний студентов от их ментальности с помощью нейронных сетей / Федяев О.И., Грабчук О.П., Елифёров В.В. // Труды конференции ИАИ-2015, КПИ, Киев, 2015. - С. 258-265.
4. Современные тенденции в управлении персоналом. Учебное пособие / Дейнека А.В. [и др.]; - М.: Изд-во "Академия естествознания", 2009. - 294 с.
5. Федяев О.И. Нейросетевая модель процесса профессионального обучения молодых специалистов // Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2015): материалы 5-й межд. научно-техн. конф. (Минск, 19-21 февраля 2015 г.). – Минск: БГУИР, 2015.
6. Айзенк, Г. Как измерить личность: учеб. пособие / Айзенк Г., Вильсон Г. - М.: Изд-во Когнито-Центр, 2000. - 27 с.
7. Айзенк Г. Новые тесты IQ - М.: Изд-во «ЭСКМО», 2003. – 189 с.
8. Ильин Е.П. Психология творчества, креативности, одарённости. - СПб.: Питер, 2004. – 537 с.
9. Рабинер, Л.Р. Цифровая обработка речевых сигналов / Рабинер Л.Р., Шафер Р.В. - М.: Изд-во Радио и связь, 1981. – 42 с.
10. CMUSphinx Documentation [Electronic resourse] / Интернет-ресурс. - Режим доступа: https://cmusphinx.github.io/wiki/. - Загл. с экрана.
11. Гончаров, К.Д. Система построения ментального портрета студента с речевым интерфейсом [Электронный ресурс] / Гончаров К.Д., Федяев О.И. //Информатика, управляющие системы, математическое и компьютерное моделирование в рамках IV форума «Инновационные перспективы Донбасса» (ИУСМКМ - 2018): IX Международная научно-техническая конференция, 22-24 мая 2018, г. Донецк: / Донец. национал. техн. ун-т. – Донецк: ДонНТУ, 2018. – Режим доступа: http://iuskm.donntu.org/elektronnyj-arxiv-konferencii/.

Гончаров К.Д., Федяев О.И. Речевое управление системой оценивания ментальности студентов. *Для студентов со слабыми навыками работы с периферийными устройствами управления компьютером разработан канал речевого управления программой построения менталитета студента. Ментальность студента определяется психологическим тестированием в течение длительного диалога с компьютером. Диалог на основе речевых команд повышает интеллектуальность человеко-компьютерного взаимодействия, что облегчает процесс тестирования студентов.*

Ключевые слова: *ментальность студента, психологическое тестирование, распознавание речи, речевой интерфейс*

Goncharov K.D., Fedyaev O.I. Speech control evaluation system of student's mentality. *For students with poor skills in working with peripheral computer control devices developed a channel of speech control program to build the mentality of the student. The mentality of a student is determined by psychological testing during a long dialogue with a computer. Dialogue based on speech commands increases the intelligence of human-computer interaction, which facilitates the process of testing students.*

Key words: *student's mentality, psychological testing, speech recognition, speech interface*

УДК 004.457

Анализ состояний методов, алгоритмов и программных средств в поддержке управления версиями

Григорьев А.В., Гурин А.Г.
Донецкий национальный технический университет
grigorievalvl@gmail.com, alexandergurin1996@gmail.com

Григорьев А.В., Гурин А.Г. Анализ состояний методов, алгоритмов и программных средств в поддержке управления версиями. *В статье представлен краткий обзор актуальных решений в области управления версиями. Произведен анализ актуальности и основных задач системы управления версиями. Произведен краткий обзор, существующий алгоритмов и программных решений. И сделан анализ каждого из них.*

Ключевые слова: *система управления версиями (СУВ, СКВ или система управления версиями), репозиторий, ветка, комит, ревизия, SVN, Git, Mercurial.*

Введение

На данный момент программная инженерия является самой востребованной и перспективной инженерной специальность. А задачи программной инженерии являются актуальными.

Весь комплекс работ, объединённый терминов - программная инженерия, можно разбить на ряд потоков. Перечислим имеющиеся потоки программной инженерии [1]:

- деловое моделирование – формирование требований;
- управление средой – создание регламентов работы;
- анализ требований;
- управление проектом – стратегическое и тактическое планирование;
- анализ и проектирование;
- разработка;
- тестирование.

Последние три потока выполняются параллельно и тут важно при работе, чтобы программисты не мешали работе друг другу. В потоке «анализ и проектирование» происходит обратная связь при выявлении несоответствий в требованиях. На данном потоке важна достаточная информативность и способность корректно информировать, что изменяется на этапе разработки. В связи с этим система управления версиями является актуальным решением в данной ситуации. Одной из важнейших проблем является организация программного комплекса для поддержки решения задачи управления версиями.

Целью данной работы является анализ методов и программных реализаций в области управления версиями, выявление проблем и обсуждения возможных путей по их решению.

Место и роль задачи контролей версий в потоке работ программной инженерии

Во время разработки программного продукта производится частое изменения программного кода. Иногда изменения какого-либо участка кода влечет за собой фатальные проблемы. И нужно не мало сил потратить для выявления проблемного участка. Становится вопрос о контроле изменений.

Также если учесть текущее состояние развитие программной инженерии, а именно то, что проекты в большинство случаев разрабатывает не одним человеком, а группой людей. То это влечет ряд проблемных ситуации, которые важно учитывать. Например, когда два человека могут изменять один и тот же файл или даже часть кода; когда кому-то нужно произвести эксперимент новой идеи, без затрагивания основного проекта и не создавая новый. Тут появляется второй вопрос, а именно, как спроектировать работу так, чтобы не возникало коллизий во время разработки программного продукта.

И так, подытожим и получим общую проблему, которая заключается в возможности работать группой людей без вмешательства в работу каждого члена группы и контролируя изменения как на уровне всего проекта, так и каждой отдельной под части его.

Анализ функций и роли системы контроля версиями

Система управления версиями — это ПО для упрощения работы с документами, которые часто меняются. Система управления версиями может хранить разные версий одного и того же документа, и позволяет перемещаться по разным версиям проекта и даже к самым первым. Так же, при получении любой версии, можно посмотреть все изменения документа, и кто в какое время производил изменения.

Данные системы часто применяются при разработке ПО для хранения и управления документов с исходным кодом. Но это не исключает ситуацию, что их можно использовать и в других сферах, в которые работа проходит в часто изменяющихся документах и нужно отслеживать все изменения с ними. Так же стоит отметить, что очень часто СУВ используются в САПР, которые используются в составе системе управления данным об изделии (PDM). Еще можно встретить СУВ при использовании конфигурационного управления [2].

Традиционные системы контролей версий имеет централизованную модель. Система имеет определенное хранилище и управление этой системы происходит на сервере. Порядок основных действий с системой контролей версий:

1. Пользователь, который работает с файлом, получает нужную версию с хранилища. Получить можно абсолютно любую версию, как и самую последнею, так и самую первую.
2. Производится создание локальной копии на компьютере пользователя.
3. Затем пользователь вносит изменения и сохраняет файл, который автоматически будет помещен в хранилище, но важно понимать, что старый удален не будет.

Как было сказано выше, файл могут изменять два и более пользователя. В такой ситуации может произойти то, что один пользователь случайно отменить изменения другого. Система контролей версии отслеживает подобные конфликтные ситуации и способна предоставлять их решения. Если изменения двух пользователей не влияют друг на друга, то система способна сама объединить изменения обоих пользователей в один результирующий файл.

В различных системах управления встречаются такие возможности, как:

– создание разных вариантов одного файла, так называемые ветки, которые содержат полную историю изменения документа до ветвления и историю каждой ветки отдельно;

– просмотр полного описания изменений версии одного документа, в частности кто вносил изменения в определенные части строк документа и когда

– ведение журнала изменения, который содержит в себе пользовательские пояснения о данных изменения в каждой версии документа

– контроль прав доступа каждого пользователя, например, разрешая и запрещая чтение, либо изменения данных, в зависимости запрашиваемого доступ.

Существуют еще распределенный системы управления версиями. Их главное отличие в том, что они имеют распределенную модель, а не традиционную клиент-серверную. Они не нуждаются в наличии определенного хранилища: вся история изменения и все версии хранятся на машине, в локальном хранилище, и только при необходимости нужные части документов синхронизируются с подобным хранилищем на другом компьютере. Это дает преимущество перед централизованными в том, что каждый пользователь работает над своей локальной копией, а затем уже добавляет (производит push) на централизованное хранилище. Но перед данным этапом нужно загрузить актуальную копию на своей компьютер (произвести pull). Это делается с той целью, чтобы, если есть конфликты в двух версиях документа (того, который на другом компьютере и того, который на локальном компьютере), их разрешить. У распределенных есть ряд недостатков:

– увеличенное требования к объему дисковой памяти, ибо на локальной машине будут хранится все версии документов;

– нет возможности блокировать или следить за файлом или группу файлов;

– нет единой сквозной нумерации системы.

Преимущества распределенных систем управления версиями:

– избавляет от необходимости выделять один из компьютеров под сервер;

– во время разработки небольшого проекта не нужно выделять общие ресурсы;

– во время разработки крупного распределенного проекта, в котором пользователи могут работать долгое время над своей локальной версией, при этом без обязательного постоянного наличия в интернет.

Системы контроля версий могут использоваться, так называемые дельта-компрессии – это такой подход, при котором записывается только изменения документа, а не всего версии. Простой пример может привести на цифрах, вместо записи 2, 4, 6, 7, 9, 5, 15, мы будем записывать 2, 2, 2, 1, 2, -4, 10 [3]. Так как в большинстве случаев нужно последняя версия, то на сервере записываем, заменяя предыдущую актуальную модель.

Обзор программный реализаций

В данном пункте будет рассмотрены и проанализированы следующий программные реализации: Система одновременных версий (CVS), Apache Subversion (SVN), Git, Mercurial.

CVS [4] впервые появилась в 1980-х, но несмотря на это является популярной и по сей день. CVS

использует открытой лицензионное соглашение GNU. Основной функцией CVS является получать и отправлять на сервер определенную версию документа. Целью создания CVS являлось уметь разрешать конфликта между версиями при разработке. Также стоит отметить, что у каждого пользователя была возможность иметь только самую последнюю версию проекта. Это являлось первой системой контроля версиями. Недостатком было то, что пользователь обязан был успеть отправить новую версию своего документа на сервер, пока другие его не опередили. На данный момент этот недостаток разрешили возможностью работать с ветками проекта.

SVN [5] была создавалась альтернативой CVS. Целью создания базировалось на исправлении недостатков CVS, а также не забывать про обратную совместимость. SVN так же является бесплатной системой контроля версиями с открытым исходным кодом. SVN распространятся под лицензией Apache. Данная СУВ начала использовать атомарные операции, чтобы сохранить целостность БД. Что имеется ввиду, при создании новой версии к последней измененной версии проекта применяются либо все исправления, либо ни одно из них. Это создавала защиту кода от случайных и частичных правок, которые в некоторых случаях вызывают даже ошибки. Причиной того, что разработчики отдали предпочтение SVN и отказались от CVS в том, что SVN взял все лучшие от CVS и еще дали разработчикам больше возможностей. SVN создавалась в первую очередь для более крупных проектов, в которых использовалось ветвление кода и было не одно направление разработки.

Git [6] была создана для управления разработкой ядра Linux [7]. Данная система начала использовать категоричной другой подход от CVS и SVN. При создании Git было целью создать более быструю распределенную СУВ. Если взять во внимание, что Git создавался под Linux, то в этой ОС она работает лучше всего. Но также есть возможность пользоваться на MacOS и Windows. Также стоит заметить, что каждая версия содержит всю историю, что является весомым плюсом, если ведется разработка без интернет-соединения. Присутствует навигация по всей истории.

Mercurial [8] дебютировал в одно время с Git. Mercurial, как и Git, является распределенной системой контроля версиями. Mercurial являлся альтернативой при разработке модулей для ядра Linux. Mercurial используется меньше, но многие ведущие разработчики предпочли работать с этой СУВ, к ним можно отнести и OpenOffice.org [9]. Главное отличие данной системы от остальных является то, что она полностью написана на Python, а не на C. Но некоторые модули-расширения все равно были разработаны на C [10].

Была разработана таблица 1, в которой приведены все преимущества и недостатки каждого варианта.

Таблица 1 – Сравнение программных реализаций системы управления версиями

Название программной реализации	Преимущества	Недостатки
CSV	− некоторые отдают ей предпочтение, за то, что она проверена временем.	− история не содержит переименованные или перемещённые файлы; − символические ссылки на файлы создают уязвимости в безопасности; − атомарные операции отсутствуют, что есть вероятность, что приведет к повреждениям кода; − использование операций с ветками дорого обходятся.
SVN	− может работать с атомарными операциями; − использование операций с ветвлением кода не так дорого обходятся; − не малый ассортимент плагинов IDE; − не использует пиринговую модель.	− ошибки, которые связаны с переименованием файлов и директорий, до сих пор присутствуют; − проблемы с набором команд для работы с репозиторием; − скорость относительно меньше конкурента.
Git	− значительное увеличение в скорости работы; − использований операций с ветвлением кода являются дешевыми; − можно работать без подключения к интернету; − распределенная, пиринговая модель.	− присутствуют ограниченная поддержка Windows.

Mercurial	− легкий в освоение при старте; − присутствует документация подробная; − распределенная модель.	− нет возможности слияния двух родительских веток.

Анализ главных проблем и формирования основных требований

Проанализировав основные методы и программные средства системы управления версиями. Было выявлено, что в потоки программной инженерии идет уклон на распараллеливание рабочих процессов, где очень важно, чтобы работа одного программиста не влияла на работу другого. Также к важным моментам нужно отнести недостаточную информативность при работе с разными версиями и ветками в СУВ. При получении новой более актуальной версии, иногда уходить много времени на анализ и понимание того, что было изменено именно в данной версии. Комментарий при загрузке на сервер (так называемый «комит») в большинстве случаев имеет ограниченный размер, и не все способны корректно отметить выполненное. В связи с этим способность выявлять исправления и предоставлять вместо измененных строк сформулированный анализ изменений является уместной в потоке программной инженерии.

Составлены основные требования для решения проблем и реализации программного обеспечение:

– возможность запоминать предыдущие версии проекта и каждого документа;
– наличие полной и подробной истории изменения;
– возможность перейти к любой версии проекта;
– возможность перейти к любой версии отдельного документа;
– наличие «веток» с подробной историей для каждой из них;
– возможность работать с проектом, либо отдельным файлом одновременно нескольким пользователям;
– возможность разрешения конфликтов;
– возможность работать без подключения к интернету;
– система должна самостоятельно отслеживать и автоматически анализировать любые изменения, записывать подробную информацию по каждому изменению.

Вывод

В данной работе был произведен анализ существующих методов и программных средств в области разработки и использования систему управления версиями. Были выявлены главные проблемы, с которыми могут столкнуться, как и опытные программисты, так и молодые разработчики. К каждой проблеме были рассмотрены пути их решения. В этом докладе, как итог, были сформулированы основные требования, которые будут использованы в дальнейшей разработке данного программного обеспечения.

Особенностью и новизной будет требование, которые отвечает за функцию анализа и подробного описания каждой версии проекта без вмешательства пользователя. Данная особенность предоставляет перспективы в дальнейшем развитии проекта.

Следующим шагом является анализ требований, который будет рассмотрен в следующих докладах.

Литература

1. Анализ требований и другие рабочие потоки программной инженерии [Электронный ресурс] – Режим доступа: https://studfiles.net/preview/1644196/page:17/ – Загл. с экрана.

2. Система управления версиями [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%8F%D0%BC%D0%B8 – Загл. с экрана.

3. Дельта-кодирование [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BB%D1%8C%D1%82%D0%B0-%D0%BA%D0%BE%D0%B4%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5 – Загл. с экрана.

4. CVS [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/CVS – Загл. с экрана.

5. Subversion [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Subversion – Загл. с экрана.

6. Git [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Git – Загл. с экрана.

7. Linux [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Linux – Загл. с экрана.

8. Mercurial [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Mercurial – Загл. с экрана.
9. Apache OpenOffice - свободный и открытый офисный пакет [Электронный ресурс] – Режим доступа: https://www.openoffice.org/ru/ – Загл. с экрана.
10. Система контроля версий (cvs) 2018 - Сравниваем: Git, SVN, Mercurial [Электронный ресурс] // Википедия – Режим доступа: https://biz30.timedoctor.com/ru/c%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0-%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8F-%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9/ – Загл. с экрана.

Григорьев А.В., Гурин А.Г. Анализ состояний методов, алгоритмов и программных средств в поддержке управления версиями. *В статье представлен краткий обзор актуальных решений в области управления версиями. Произведен анализ актуальности и основных задач системы управления версиями. Произведен краткий обзор, существующий алгоритмов и программных решений. И сделан анализ каждого из них.*

Ключевые слова: *система управления версиями (СУВ, СКВ или система управления версиями), репозиторий, ветка, комит, ревизия, SVN, Git, Mercurial.*

Grigoriev A.V., Gurin A.G. Analysis of the state of methods, algorithms and software in support of version control*. The article provides a brief overview of current version control solutions. An analysis of the relevance and main tasks of the version control system has been made. Produced a brief overview of existing algorithms and software solutions. And an analysis of each of them.*

Keywords: *version control system, repository, branch, commit, revision, SVN, Git, Mercurial.*

УДК 004.418

Информационные технологии и мобильное программное обеспечение для контроля за патогенезом и лечением головных болей

Зарецкий А.О., Ложкин А.В.
Научный руководитель: А.Г. Пимонов, д.т.н., профессор
Кузбасский государственный технический университет
имени Т.Ф. Горбачева г. Кемерово
ivlnwn@gmail.com, aleksandrforjob@gmail.com, pag_vt@kuzstu.ru

Зарецкий А.О., Ложкин А.В. Информационные технологии и мобильное программное обеспечение для контроля за патогенезом и лечением головных болей. *В статье представлено описание проблемы анализа данных в сфере медицинских информационных систем. Описаны основные механизмы за наблюдением течения головных болей. Описан процесс разработки мобильного приложения для сбора данных о головных болях и представления статистики на основе этих данных.*

Ключевые слова: *разработка мобильного приложения, дневник головных болей, анализ, лечение головных болей, контроль патогенеза.*

Введение

Большинство разрабатываемых в настоящее время медицинских информационных систем, в том числе больничных информационных систем, предусматривающих хранение данных о пациентах, не содержат полностью формализованной информации, допуская в значительной части полей ввод свободного текста. Анализ информации, накопленной в такого рода информационных системах является весьма ограниченным, то есть возможен лишь в отношении формализованных полей. Данное ограничение является принципиальным при решении вопроса о приобретении программных продуктов, в частности в научных медицинских учреждениях, где применяются профессиональные средства анализа данных, например, статистический анализ, нейросетевые методы, технологии выявления скрытых закономерностей.

В связи с указанными обстоятельствами необходимо подчеркнуть, что все современные пакеты прикладных программ для анализа данных предусматривают обмен данными (как импорт, так и экспорт) в распространенных форматах dbf, xls, txt, pdf и др.

Довольно широко распространено мнение, что анализ данных в медицинских исследованиях проводить не требуется: если требуется анализ данных, то это значит, что сам эксперимент плох (в смысле того, что фиксируемый эффект незначителен). Однако применение статистики в медицинских исследованиях не ограничивается анализом данных. Очень важно применить статистику на этапе планирования медицинского исследования. Мы придерживаемся мнения, что и в медицинском исследовании для анализа данных необходимо применение статистики, в противном случае выводы нельзя считать научно доказанными.

Медицинские клинические исследования намного более сложны по сравнению с биологическими экспериментами: во-первых, на человеке эксперимент возможен лишь в ограниченных рамках, поэтому возможности задания необходимых условий исследования существенно ограничены; во-вторых, фиксируемые эффекты обычно невелики (не превышают 20%); в-третьих, выборки гораздо менее однородны. Именно поэтому в клинических исследованиях статистический анализ данных значительно более сложен и трудоемок.

В настоящее время становятся все более актуальными корректное применение статистических методов, научный подход к планированию медицинских исследований. Это связано с развитием доказательной медицины, постепенной интеграцией отечественной науки в мировую, развитием грантовой системы поддержки науки и, следовательно, с повышением требований к методической стороне исследований [1].

Исследование головных болей

Головные боли (цефалгии) входят в десятку самых частых причин нетрудоспособности. Знание четырех типов цефалгий особенно важно для врача общей практики, так как они встречаются наиболее часто и определяют основную часть социально-экономического и других видов ущерба, связанных с головными болями в обществе. Ведение пациентов с этими наиболее распространенными типами головной боли в основном входит в обязанности врачей общей практики.

Задачей данных рекомендаций является помочь врачам общей практики правильно диагностировать эти распространенные типы головных болей, грамотно распознавать признаки серьезных заболеваний, проявляющихся головной болью, и, при необходимости, направлять пациентов за специализированной помощью. Целью настоящих рекомендаций является предоставление четких и легко выполнимых алгоритмов врачам, не являющимся экспертами в области головных болей.

Каждый пациент, которому было назначено лечение или схема лечения была изменена, нуждается в динамическом наблюдении для контроля оптимальности терапии. Инструментами для такого наблюдения являются дневники головных болей. [2]

Ведение дневника рекомендуется для:

- записи симптомов и временных параметров, позволяющих правильно поставить диагноз
- оценки приема лекарственных средств и выявления возможного злоупотребления препаратамin анализа взаимосвязи головной боли с менструальным циклом и другими провоцирующими факторами
- того, чтобы пациент видел положительные обнадеживающие результаты профилактического лечения
- регистрации записи эффекта лечения
- мониторинга приема препаратов и выявления возможного злоупотребления лекарственными средствами
- оценки эффективности лечения

Требования к функционалу разрабатываемого продукта

Основная задача приложения выяснить, помогает ли пациенту прием препаратов или же требуется изменить лечение. Нужно позволить пользователю указывать важные для диагностики врачом аспекты болей и при этом сократить объем информации, которая будет являться “шумом” в диагностическом исследовании.

Данные которые может указывать пользователь:

- длительность головной боли (в часах)
- интенсивность головной боли (по десятибалльной шкале)
- сопутствующие симптомы (аура, боли в шее и т.д.)
- препараты, которые были приняты и их дозировка

Следующими основными требованиями к функционалу приложения является возможность просмотра статистики по периодам, в которые пользователь вел дневник, а также выгрузка этой статистики в формате pdf для возможности отправки такого отчета лечащему врачу.

Также у пользователя должна иметься возможность добавления новых препаратов в список стандартных (которые изначально установлены в приложении).

Аналогичные решения

Исследуя рынок приложений с аналогичным функционалом, нами не были найдены наиболее подходящие варианты на русском языке, которые могли бы полностью удовлетворить требования, описанные выше. Также были найдены решения, в которых информация доступная для редактирования ввода пользователем является избыточной и не будет подходящей для того чтобы лечащий врач смог комфортно сформулировать заключение и скорректировать, либо прекратить текущее лечение пациента.

Структура базы данных

Для iOS и Android приложения была выбрана СУБД SQLite которая будет располагаться в локальном хранилище на смартфоне пользователя.

Планирование и оценка разработки приложения

На этом этапе разработчик изучает техническую документацию. Рассчитывает, сколько времени потребуется на разработку и тестирование. Выполняет не описанные сценарии и узкие места в ТЗ.

Экспресс-оценка занимает от нескольких часов до одного дня и даёт примерное представление о трудозатратах. Детальная оценка может длиться от нескольких дней до недели, но она позволяет точно определить, как, когда и какое приложение будет получено в результате. Если бизнес-аналитик подключается к проекту на этапе оценки, клиенту и разработчикам легче получить единое представление о приложении и всё точно рассчитать.

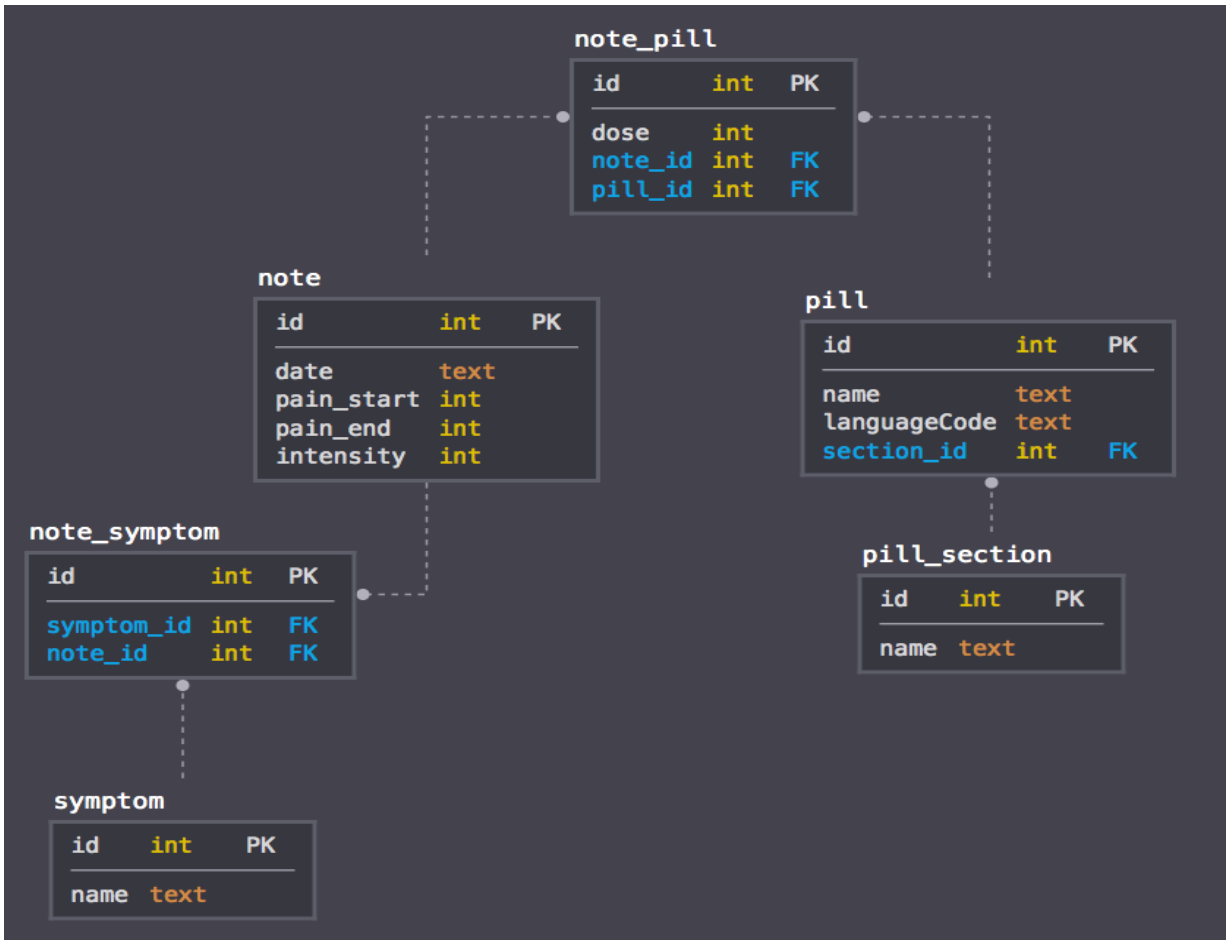


Рисунок 1 - Схема базы данных приложения

Дизайн приложения

Разработкой дизайна приложения занимался дизайнер, что позволило иметь приложению более привлекательный вид, а также UX (user experience) наиболее высокого уровня.

Наблюдая за тенденциями в дизайнах пользовательских интерфейсов приложений было принято решение реализовать для нашего продукта две цветовые темы: светлую и темную.

Разработка приложения

Когда есть развернутое ТЗ и оценка, готов дизайн и утверждён прототип мобильного приложения, начинается разработка приложения. Команда разработчиков пишет код, чтобы реализовать запланированное поведение приложения. А также требуется воплотить готовый дизайн в коде — описать все стили и элементы пользовательского интерфейса, с которыми взаимодействует пользователь приложения.

В нативной разработке мы применили языки Java для Android, Swift для iOS. В кроссплатформенных решениях пользуются React Native и NativeScript, но масштаб приложения позволил нам использовать инструменты нативной разработки для достижения оптимального качества приложения.

После того как часть функционала разрабатывается происходит ее тестирование и отладка. После того как разработанный блок функционала проходит все проверки качества, программист приступает к следующему блоку.

При реализации разработчиками пользовательского интерфейса в тестировании также принимает участие и дизайнер, который проверяет UI на соответствие макетам.

Тестирование приложения

Так как требования к конечному функционалу приложения были сформированы на начальной стадии, это позволило нам составить полный перечень unit-тестов и реализовать их прежде, чем в них появится острая необходимость. Таким образом разработка приложения ускорилась, так как время, потраченное на тестирование и отладку приложения значительно, сократилось.

Выпуск приложения

После завершения разработки и тестирования приложения, оно было выпущено в App Store, также в ближайшее время ожидается выход приложения в Play Market.

Процесс публикации приложения достаточно трудоемок и для людей, плохо знающих английский язык, может показаться сложным (для регистрации в App Store в таком случае использовать переводчика, так как в регламенте Apple обязательно подтверждение посредством телефонного разговора компании которая занимается выпуском приложения и её реквизитов).

Техническая поддержка и развитие приложения

После публикации мобильного приложения, работа над ним не заканчивается. Если клиент обнаруживает ошибки после выпуска, разработчик занимается их исправлением. Если же первые месяцы после выпуска приложения показывают, где и что нужно модернизировать или переделать, существуют два варианта решения: заключить договор на сопровождение или запустить новую фазу разработки с учётом новых переменных.

Заключение

Нами были проанализированы основные методы сбора данных о пациентах, страдающих головными болями. Выделены значимые для неврологов аспекты этих данных. Сформированы требования для функционала разрабатываемого приложения которое позволит контролировать патогенез и следить за назначенным лечением пациента.

Автоматизация процессов диагностики и наблюдение за пациентами в медицине несомненно является важным процессом и в зависимости от того как она будет развиваться, будет зависеть качество и удобство услуг, которые оказывают как государственные, так и частные медицинские учреждения.

Литература

1. Медафарм [Электронный ресурс] /Анализ данных медицинских информационных систем. – Режим доступа: http://medafarm.ru/page/stati-doktoru/informatsionnye-tekhnologii/analiz-dannykh-meditsinskikh-informatsionnykh-sistem.

2. European Headache Federation [Электронный ресурс] / T.J. Stainer [et al.] // Европейские принципы ведения пациентов с наиболее распространенными формами головной боли в общей практике. – 2010. – С. 19-20. – Режим доступа: http://ehf-org.org/wp-content/uploads/2013/12/European-Principles_Final-Russian-Version_2010.pdf.

Зарецкий А.О., Ложкин А.В. Информационные технологии и мобильное программное обеспечение для контроля за патогенезом и лечением головных болей. *В статье представлено описание проблемы анализа данных в сфере медицинских информационных систем. Описаны основные механизмы за наблюдением течения головных болей. Описан процесс разработки мобильного приложения для сбора данных о головных болях и представления статистики на основе этих данных.*

Ключевые слова: *разработка мобильного приложения, дневник головных болей, анализ, лечение головных болей, контроль патогенеза.*

Zaretsky A.O., Lozhkin A.V. Information technology and mobile software to control the pathogenesis and treatment of headaches. *The article describes the problem of data analysis in the field of medical information systems. The basic mechanisms for monitoring the flow of headaches are described. Describes the process of developing a mobile application for collecting data on headaches and presenting statistics based on these data.*

Keywords: *mobile application development, headache diary, analysis, treatment of headaches, pathogenesis control.*

УДК 004

Проблемы многопоточного программирования на платформе .NET Framework

Жильцов В.А.
Донецкий национальный технический университет
jiltsov01@yandex.ru

Жильцов В.А. Проблемы многопоточного программирования на платформе .NET Framework. *Целью данного доклада является поиск и решение проблем связанных с многопоточным программированием. В работе были рассмотрены основные понятия и проблемы, связанные с многопоточным программированием. После были предложены решения данных проблем.*

Ключевые слова: *многопоточное программирование, синхронизация потоков, асинхронное программирование, атомарные операции, .NET Framework.*

Введение

В начале своего существования операционные системы не поддерживали концепцию потоков. Следствием этого было то, что приложения, которым необходимо обработать большое количество данных замедляли работу других приложений. При разработке нового ядра операционной системы разработчики приняли решение запускать каждое приложение в отдельном процессе. Формально, процесс – это концепция уровня операционной системы, которая используется для описания набора необходимых ресурсов и памяти. Простыми словами, процесс – это выполняющаяся программа. Процессы изолированы друг от друга, следовательно, не могут повредить данные и код других процессов. В том случае, если программа зацикливается, а процессор всего один, то это негативно отразится на работе всех программ. Потоки стали решением данной проблемы. Поток – это некоторая независимая последовательность инструкций для выполнения того или иного действия в программе. Если код приложения входит в бесконечный цикл, то блокируется только это приложение, в то время как другие продолжают свою работу.

У каждого новшества есть свои минусы, потоки не являются исключением. В данной статье будут рассмотрены проблемы, связанные с многопоточным программированием, преимущественно, совместном разделении ресурсов, в дальнейшем, данные из этой статьи помогут читателям разрабатывать высокопроизводительные и отказоустойчивые приложения. Цель: указать на недостатки и проблемы, связанные с многопоточным программированием; показать способы решения данных проблем, с помощью языка C#.

Ресурсозатратность потоков

Во время создания потоков необходимо соблюдать осторожность, так как большое количество потоков может заполнить до предела оперативную память, а также загрузить процессор на 100% (обычно этот показатель не превышает 15%).

Для каждого потока резервируется память, основную часть использует стек пользовательского режима, в котором хранятся передаваемые в метод локальные переменные и аргументы. Данный стек резервирует 1 Мбайт памяти, при необходимости количество памяти может быть увеличено. Однако стоит заметить, что Windows отображает только адресное пространство, и не обязательно будет выделена физическая память в полном объеме.

Из-за высокой нагрузки процессора система охлаждения может не справляться со своей работой, что негативно отражается на продолжительности жизни некоторых компонентов компьютера. Высокий показатель нагрузки связан не только с работой, которую выполняют потоки, но и с тем, что процессор может осуществлять только что-то одно. Появляется понятие переключение контекста. Операционной системе необходимо планировать работу процессора, в произвольный момент времени Windows передает процессору на исполнение один поток, который выполняется в течение некоторого интервала времени, этот интервал обычно называют тактом или квантом, и он длится около 30 мс. После завершения или прерывания кванта происходит переключение контекста, и процессорное время выделяется другому потоку. Из-за контекстов снижается производительность и увеличивается количество потребляемой памяти, но в то же время, они повышают надежность и скорость реагирования на действия пользователей. Негативно отражается на производительности сборка мусора, в ходе которой, CLR останавливает все потоки и очищает стеки потоков.

Частичным решением проблемы, связанной с затратами на память, является пул потоков. Пул потоков – абстрактное хранилище готовых потоков, которые можно использовать в приложениях. Пулом можно управлять с помощью класса System.Threading.ThreadPool. В отличие от простых потоков, потоки из пула не удаляются

после завершения своей работы, а возвращаются и становятся в режим ожидания. Это избавляет от необходимости тратить ресурсы на удаление потоков. Когда количество запросов в очереди превышает количество свободных потоков, то создаются новые потоки. Если приложение прекращает отправлять запросы в очередь и появляются незанятые потоки, то CLR, спустя некоторый промежуток времени, зависимый от версии, сокращает количество потоков до необходимого минимума. Все потоки в пуле являются фоновыми. Программа завершается после завершения или прерывания всех активных потоков, при этом принудительно завершаются все фоновые. Этот процесс происходит немедленно и без исключений. На практике стоит отказаться от создания активных потоков, так как в случае невнимательности разработчика, приложение не будет завершаться, а подобную ошибку найти крайне трудно.

В некоторых ситуациях, значительно уменьшить количество потребляемой памяти в серверных приложениях может помочь класс System.Threading.ExecutionContext. С его помощью можно управлять контекстом исполнения любого потока. Контекст исполнения состоит из параметров безопасности, параметров хоста, а также контекстных данных логического вызова. При создании вторичных потоков в них копируется контекст исполнения исходного потока. Это обеспечивает использование одинаковых параметров безопасности и хоста, а также доступ вспомогательного потока к данным сохраненным в контексте логического вызова исходного потока. Однако это негативно отражается на производительности, так как контекст исполнения содержит много информации, которая будет копироваться и во внутренние вспомогательные потоки. С помощью статического метода SuppressFlow() можно запретить копирование контекста исполнения, однако, это следует делать только в том случае, если точно известно, что поток не будет использовать содержащуюся там информацию.

На платформе .NET существует паттерн скоординированной отмены. Для каждой затратной по времени операции стоит добавлять возможность отмены. За это отвечает класс System.Threading. CancellationTokenSource. В том случае, если пользователю необходимо отменить операцию, это уменьшит нагрузку на процессор.

Недостатки асинхронного программирования

В современных языках программирования существуют модели синхронного и асинхронного программирования. Синхронно – значит последовательно [1]. В то время как асинхронное программирование является непредсказуемым [2], из-за того, что CLR берет все заботы на себя.

Рисунок 1 – Синхронное выполнение задач

Рисунок 2 – Асинхронное выполнение задач

Выполнение задач асинхронно повышает производительность (особенно на компьютерах, имеющих более одного процессора, где можно распараллелить потоки) и улучшает время отклика в клиентских приложениях, благодаря чему, интерфейс быстрее реагирует на действия пользователя. Однако не все задачи можно выполнять асинхронно. Например, мы не можем одновременно подключаться к базе данных, обрабатывать данные из этой же базы и закрывать ее, это необходимо делать синхронно.

Существует более насущная проблема, а именно одновременный доступ к общим данным. Следует избегать ситуаций, когда несколько потоков могут менять одни и те же данные, так как это может повредить данные при одновременном обращении к ним из разных потоков. Для этого следует использовать типы значений, так как каждый поток будет работать с копией. Данные, доступные только для чтения, также нет смысла синхронизировать. Примером является тип String, созданные строки неизменны. Ссылки на созданные объекты с помощью ключевого слова new остаются в потоке и не могут быть использованы в других потоках, следовательно, нет необходимости их синхронизировать. Часто возникают ситуации, когда множество потоков работают с общими данными и приведенными выше способами эту проблему устранить невозможно (не следует забывать, что CLR самостоятельно планирует выполнение асинхронных потоков), в таком случае необходимо синхронизировать потоки. Это значительно увеличивает потребляемые ресурсы.

Синхронизация обеспечивается за счет блокирования потоков, пока другой работает с данными. Если поток из пула блокируется, то пул, скорее всего, создаст новый поток, на что приходится тратить процессорное время и память. Однако это наименьшая проблема, связанная с блокированием. Дело в том, что синхронизировать потоки крайне трудно. В начале, необходимо определить какие данные могут обрабатываться несколькими потоками одновременно, после заключить их в другой код, обеспечивающий блокирование и разблокирование. Упустив небольшой фрагмент кода, все данные будут повреждены. После необходимо протестировать

программу на компьютере с наибольшим доступным количеством процессоров, чтобы увеличить вероятность нахождения ошибки. Снижение производительности более существенное, чем кажется на первый взгляд. Так как из-за увеличения количества потоков увеличивается частота переключений контекста и повышается количество потоков обслуживающихся в пуле. Также негативно на производительность влияет тот факт, что процессорам необходимо координировать совместную работу, определяя, какой поток необходимо блокировать первым. Даже при использовании наиболее эффективного метода блокировки работа с данными замедляется в несколько раз.

Способы синхронизации потоков

Не все операции необходимо синхронизировать, так как существуют атомарные операции, которые выполняются настолько быстро, что даже в случае прерывания потока, посторонние потоки не прочитают нестабильные данные. Для следующих типов данных CLR обеспечивает атомарное чтение и запись: Boolean, Char, Byte, UByte, Int16, UInt16, Int32, UInt32, IntPrt, UIntPrt, Single и всех ссылочных типов. В атомарных операциях все байты переменной читаются или записываются одновременно.

Стоит подробно рассмотреть тип System.Threading.Interlocked, в котором находятся статические атомарные методы, с их помощью можно безопасно выполнять некоторые арифметические операции и операции присваивания. Методы, находящие в этом классе, оптимизированы. Наиболее часто используемые методы этого типа: Increment(), Decrement(), Add(), Exchange(), CompareExchange(). Создатели языка перегрузили эти методы для разработчиков, а методы Exchange() и CompareExchange() способны принимать не только типы значений, но и все ссылочные типы.

Наиболее популярной конструкцией синхронизации потоков является класс Monitor и, связанное с ним, ключевое слово lock. Для применения слова lock необходимо определиться с маркером блокировки, который будет получен потоком для входа в контекст блокировки. В общем случае синтаксис слова lock описан на рисунке 3.

```
private void SomeMethod(){
   lock (this){
      //Код, который необходимо синхронизировать
   }
}
```

Рисунок 3 — Синтаксис слова lock в общем случае

Однако если код блокируется внутри открытого члена, то рекомендуется (и является более безопасным) объявлять закрытую переменную член типа object, и использовать ее в качестве маркера блокировки.

Как всем известно, ключевые слова являются “синтаксическим сахаром” и все они связанны с определенным типом. Слово lock связанно с классом Monitor. Если модифицировать предыдущий код с использованием методов из класса Monitor, то получится конструкция, которая описана на рисунке 4.

```
private void SomeMethod() {
   Monitor.Enter(this);
   try {
      //Код, который необходимо синхронизировать
   }
   finally {
      Monitor.Exit(this);
   }
}
```

Рисунок 4 — Конструкция, полученная после модификации предыдущего кода с использованием методов из класса Monitor

Метод Monitor.Enter() в качестве аргумента получает маркер блокировки, весь код, который необходимо синхронизировать помещается в блок try, в блоке finally, с помощью метода Monitor.Exit(), маркер блокировки освобождается. Сразу виден недостаток данной конструкции: в случае появления исключения, данные будут повреждены. На практике не рекомендуется использовать этот способ синхронизации, лучше дать приложению зависнуть, чем продолжить выполнение работы с поврежденными данными. Также не желательно передавать в качестве аргумента переменную типа String. Это связанно с тем, что строки допускают интернирование, следовательно, две переменные с разными значениями могут ссылаться на один и тот же объект в памяти.

Методы класса Monitor принимают параметр типа Object, при передаче типа значения произойдет упаковка, в результате поток блокирует упакованный объект, при вызове метода Monitor.Enter() блокируется другой объект, из-за чего синхронизация потоков не происходит.

Для дальнейших способов блокировки необходимо знать о существовании абстрактного базового класса System.Threading.WaitHandle. Такие классы как: EventWaitHandle, Semaphore и Mutex являются производными, они будут рассмотрены далее. Наиболее полезные методы из класса WaitHandle: WaitOne(), WaitAll(), WaitAny(). Стоит обратить внимание, что при работе с объектами наследников этого класса нежелательно вызывать метод Dispose(), так как он закрывает дескриптор объекта ядра. Вызывать данный метод можно только при условии, что вам точно известно, что объект ядра не используется другими потоками.

EventWaitHandle. События представляют собой переменные типа bool. Ожидающий события поток блокируется, когда оно имеет значение false, и освобождается в случае true. Существует два вида событий: с автосбросом и ручным сбросом. Когда событие с автосбросом имеет значение true, оно освобождает один заблокированный поток, после освобождения первого потока ядро автоматически возвращает событию значение false. Событие с ручным сбросом освобождает все ожидающие потока, так как в этом случае значение false автоматически не присваивается, это необходимо делать вручную. Исключительные методы класса: Set(), Reset().

Semaphore. Семафоры представляют собой переменные типа Int32. Поток, ожидающий семафора, блокируется при значении 0 и освобождается при значениях больше 0. При снятии блокировки с ожидающего семафора потока автоматически вычитается единица. Текущие показания счетчика семафора не могут превысить максимальное значение Int32, иначе сгенерируется исключение SemaphoreFullException. Событие с автосбросом равносильно семафору, разница заключается в том, что метод Set() можно вызвать множество раз, но каждый раз будет освобождаться один поток, в то время как многократный вызов метода Release(метод класса Semaphore) увеличивает на единицу внутренний счетчик семафора, это дает возможность снять блокировку с большего количества потоков.

Mutex. Мьютекс предоставляет взаимно исключающую блокировку. Он функционирует аналогично объекту Semaphore со значением счетчика 1 и освобождает за раз один ожидающий поток. Мьютексы имеют дополнительную логику, следовательно они более сложны. Объекты Mutex сохраняют информацию о том, какие потоки ими владеют, это осуществляется за счет того, что они запрашивают идентификатор потока. При вызове метода ReleaseMutex(), объект сначала убеждается, что это владеющий им поток, иначе генерируется исключение AbandonedMutexException. Если это исключение не обработать, то завершится весь процесс. Чаще всего так и происходит, иначе поток будет работать с поврежденными данными. Объекты Mutex управляют рекурсивным счетчиком, показывающим, сколько раз поток-владелец уже владел объектом. Если поток владеет мьютексом в настоящий момент и ожидает его еще раз, счетчик увеличивается на единицу и потоку позволяется продолжить выполнение. При вызове метода ReleaseMutex() рекурсивный счетчик уменьшается на единицу. Если значение достигнет 0, то владельцем мьютекса может стать другой поток. Данная логика негативно отражается на производительности. Но этот способ используется при двойной блокировки, когда в методе вызывается другой метод, которые тоже необходимо блокировать.

Выводы

В ходе работы были рассмотрены такие понятия как: поток, процесс, контекст, фоновые и активные потоки и другие понятия. Найдены способы снижения количества потребляемой памяти и загрузки процессора. При исследовании модели асинхронного программирования были обнаружены недостатки данной модели и показаны примитивные способы решение проблем, связанных с многопоточным программированием.

Для более глубокого изучения данных проблем можно использовать книги: CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#, Джеффри Рихтер; Тонкости программирования, Джон Скит. Множество информации по данной теме можно найти на официальном сайте документации Майкрософт (https://docs.microsoft.com). Также можно перейти на более низкий уровень и рассматривать то, как операционная система планирует выполнение потоков.

Литература

1. Джеффри Рихтер. *CLR via C#.* Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. - 4-е изд. – Санкт-Петербург: Питер, 2017. – 896 с.

2. Эндрю Троелсен. Язык программирования C#5.0 и платформа .NET Framework 4.5. - 6-е изд. – Санкт-Петербург: Питер, 2013. – 1312 с.

3. docs.microsoft [Электронный ресурс] // Справочник C# – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/.

Жильцов В.А. Проблемы многопоточного программирования на платформе .NET Framework. *Целью данного доклада является поиск и решение проблем связанных с многопоточным*

программированием. В работе были рассмотрены основные понятия и проблемы, связанные с многопоточным программированием. После были предложены решения данных проблем.

Ключевые слова: *многопоточное программирование, синхронизация потоков, асинхронное программирование, атомарные операции, .NET Framework.*

Zhiltsov Vladimir. The problems of multithreaded programming on the platform .NET Framework. The aim of this report is the search of problems, connected with multithreaded programming, and their solutions. Main concepts and problems, which concern multithreaded programming, were reviewed in this work. Then some solutions of given problems were suggested.

Key words: *multithreaded programming, asynchronous programming, synchronization of threads, atomicity operations, asynchronous programming, .NET Framework.*

УДК 004.588, 004.432

Создание обучающей системы языка программирования php в виде квест-игры

Журавлёв А.В.
Донецкий национальный технический университет,
кафедра компьютерного моделирования и дизайна
E-mail: alekptitsa@yandex.ua

Журавлёв А.В. Создание обучающей системы языка программирования php в виде квест-игры. *В статье рассмотрена задача создания обучающей системы языка программирования php представленной в виде веб-приложения(сайта). Проект реализован с помощью чистого php кода, чтобы наглядно продемонстрировать, какие возможности имеет этот язык программирования. Основываясь на анализе популярных сайтов-учебников по php предложены основные функциональные возможности веб-приложения – курс обучения в виде увлекательных лекций, подкрепленных интересными задачами и тестами, и всё это представлено в формате квеста-игры для людей с базовыми знаниями о веб разработке.*

Ключевые слова: *язык программирования, php, веб-приложение, обучающая система, сайт, квест-игра.*

Введение

Программирование — процесс создания компьютерных программ. Программирование основывается на использовании языков программирования, на которых записываются исходные тексты программ.

Веб-программирование – это, бурно развивающийся раздел программирования, ориентированный на создание динамических веб приложений(сайтов). Принято условно разделять языки веб-программирования на две пересекающиеся группы: клиентские и серверные.

Программы, написанные на клиентских языках, выполняются на стороне пользователя, отсюда и придумано название, как правило их выполняет браузер. Результат выполнения программы полностью зависит от браузера пользователя, его типа и версии, что является главной проблемой клиентских языков. То есть пользователь может запретить выполнять клиентские программы и как бы не хотел того программист, исполняться они не будут. Кроме того, может произойти такое, что в разных браузерах или в разных версиях одного и того же браузера один и тот же скрипт программы будет выполнен по-разному. Но с другой стороны, программист может снизить нагрузку на сервер за счет программ, исполняемых на стороне клиента, тем самым упростить работу серверных программ, так как клиентские не всегда требуют перезагрузку страницы. Самыми популярными клиентскими языками программирования являются:

- Javascript;
- Java;
- VBScript.

Серверные языки

При подаче запроса пользователем на какую-либо страницу (переходе на нее по ссылке или введении адреса в адресной строке своего браузера), вызванная страница сначала обрабатывается на сервере, и только потом возвращается посетителю по сети в виде файла.

Работа программ уже полностью зависит от сервера, на котором расположен сайт, и от того какая версия того или иного языка поддерживается.

Наиболее популярные серверные языки программирования:

- PHP;
- Perl;
- Ruby;
- Python.

Важным преимуществом работы серверных языков является возможность осуществления непосредственного взаимодействия с системой управления базами данных(СУБД) – сервером, на котором в таблицах упорядоченно хранится информация, которая может быть вызвана в любой момент. Самые часто используемые СУБД:

- MySQL;
- Oracle;

- SQLite;
- Microsoft SQL Server.

Анализ современных средств реализации

Самый распространенный язык веб-программирования на данный момент – это PHP. PHP (Hypertext Preprocessor) – это язык программирования общего назначения с открытым исходным кодом. PHP был сперциально сконструирован для веб-разработок и его можно внедрять непосредственно в html.

В качестве основного средства реализации обучающей системы взята самая актуальная версия PHP 7.3, которая является и самой стабильной на сегодняшний день.

Почему именно PHP**.** Главной областью применения PHP является написание скриптов, работающих на стороне сервера. Это значит, что PHP способен выполнять все то, что может любая CGI программа, например, генерировать динамические страницы, отсылать и принимать cookies и обрабатывать дынные форм. Но способностей PHP намного больше.

Существует три основные области применения PHP:

- создание скриптов для выполнения на стороне сервера;
- создание скриптов для выполнения в командной строке;
- создание оконных приложений, выполняющихся на стороне клиента.

Ещё одним преимуществом PHP над многими другими языками является то, что скрипты можно создавать и редактировать в любом текстовом редакторе на любо операционной системе.

Анализ структуры сайтов посвященных изучению PHP

Для анализа были выбраны ведущие сайты-учебники языка программирования php, представленные в таблице 1. Для данных сайтов был проведен детальный анализ их структуры.

Таблица 1. Структуры разделов сайтов-учебников

Сайт	Структура
PHP720	Главная страница – Как начать – Задания – Снипетты – Книги – Поддержать – Поиск.
PHP-S	Главная страница – Фото галерея – Новости - Рейтинг – Комментарии – Гостевая книга – Обратная связь – Загруска файлов – опросы – музыкальная библиотека – регистрация и авторизация – Генератор QR кода – Скрипты обзора хостингов – Клоны скриптов – Баннеры и реклама – Скрипты чата и сообщения
PHP.SU	Главная – Статьи – Изучение PHP – Функции PHP – FAQ – PHP скрипты – MySQL – Установка – Учебники – Уроки – Download – Форум.

В качестве наиболее характерных были выделены следующие основные разделы для структурирования содержимого сайта: Главная страница, Уроки, Задачи.

Особенности организации веб- приложения

К базовым элементам отдельных страниц сайта относят заголовок страницы, навигационное меню, контент и подпись организации. Как правило, подпись располагают внизу после содержимого страницы. Навигационное меню является важной частью страницы сайта и его расположение должно не только привлекать внимание нового посетителя, но и быть удобным для постоянного посетителя. Следовательно, целесообразно разместить навигационное меню, в левой части страницы на уровне ленты содержимого для более удобного использования.

Учитывая особенности физиологии и психологии восприятия человеком информации, просмотр содержимого страницы им осуществляется в соответствии с правилами чтения (в большинстве стран мира – слева-направо, сверху-вниз). Следовательно, располагать основные элементы страницы необходимо по аналогии в порядке их важности. Разработанный макет структуры страницы интернет сайта представлен на рисунке 1.

Все пользователи сайта делятся на зарегистрированных и незарегистрированных. Зарегистрированные на сайте пользователи могут пользоваться всеми возможностями обучающей системы, а незарегистрированным пользователям доступны лишь пара разделов с общей информацией об интернет проекте и о его создателях.

Прохождение курса, доступное зарегистрированным пользователям, проходит последовательно – каждая следующая лекция станет доступна после прохождения предыдущей, задачи открываются так же по мере прохождения лекций, каждой лекции соответствует одна или несколько задач и тестов.

Рисунок 1 – Макет страницы веб-приложения

Права доступа к разделам сайта зарегистрированных и не зарегистрированных пользователей представлены в таблице 2. Возможности работы с материалами сайта для зарегистрированных и незарегистрированных пользователей представлены в таблице 3.

Таблица 2 – Права доступа к разделам веб-приложения

	Доступ	
	Открытые разделы	Скрытые разделы
Незарегистрированный	+	--
Зарегистрированный	+	+

Таблица 3 – Возможности работы с материалами обучающей системы

Пользователи	Работа с материалами		
	Прохождение курса и решение задач	Общение с другими пользователями	Доступ к общей информации
Незарегистрированный	--	--	+
Зарегистрированный	+	+	+

Реализация веб-приложения

Для создания обучающей системы был выбран язык программирования PHP версии 7.3. Данная версия поддерживает все нововведенные функции, а также последние версии языков программирования, разметки и стилей JavaScript, HTML и CSS. Редактирование и настройка компонентов веб-приложения будет осуществлена вручную с помощью редактирования скриптов PHP.

В реализации данной обучающей системе не будут использованы CMS (системы управления содержимым), дабы наглядно показать на примере данного веб-приложения возможности PHP. В процессе пользования обучающей системы в автоматическом режиме будет редактироваться база данных MySQL, содержащая в себе информацию о пользователях, о лекциях и практических задачах.

Преимущества обучающей системы. Из анализа сайтов следует заметить, что сайтов-учебников и справочников по PHP достаточно много, во многих присутствуют справочные материалы и задачи. Но рассматриваемая обучающая система имеет ряд преимуществ, такие как:

1. Интерактивное решение задач – проверка корректности введенного кода, проверка тестов в самой обучающей системе, что является удобным для пользователей, нет необходимости открывать текстовый редактор, запускать локальный сервер для проверки своего кода, проверка происходит на сервере обучающей системы.

2. Представление лекционного материала в виде квест-игры(комикса с большим количеством изображений) ориентированной в основном на взрослую аудиторию. Что делает процесс обучения намного интереснее и увлекательнее, благодаря чему материал будет усваиваться гораздо эффективнее.

3. Присутствие системы уровней и баллов. Чем выше уровень, тем больше лекций доступно к изучению, за решенные задачи начисляются баллы, за которые открываются лекции. Система уровней и баллов необходима, чтобы создать некий соревновательный процесс между пользователями, дополнительная мотивация к изучению.

4. Возможность общения с другими пользователями позволяет, например, новичку задать вопрос или спросить совет у более опытных учеников.

Выводы

В данной статье был описан процесс проектирования и реализации веб-приложения обучающей системы языка программирования PHP, для создания которой и был использован чистый PHP код, чтобы наглядно продемонстрировать возможности данного языка программирования, который на данный момент занимает лидирующую позицию в области создания веб-технологий.

Основываясь на анализе популярных сайтов-учебников по PHP описаны основные преимущества данной обучающей системы перед другими, а также предложены основные функциональные возможности веб-приложения – целый курс обучения PHP разбитый на множество увлекательных лекций, представленных в формате квест-игры с интересным и захватывающим сюжетом, подкрепленных интересными тестами и задачами, которые интерактивно решаются в самой обучающей системе, без необходимости использования сторонних текстовых редакторов и прочего ПО.

Список литературы

1. Википедия свободная энциклопедия / Интернет-ресурс. – Режим доступа: www/URL:https://ru.wikipedia.org/wiki/Веб-программирование

2. Словарь “Академик”/ Что такое PHP? / https://dic.academic.ru/dic.nsf/ruwiki/641

3. Официальный справочник по PHP /Возможности PHP/ http://php.net/manual/ru/intro-whatcando.php

4. Блог “Твой Шанс” / Особенности восприятия контента сайта / http://www.tvoyshans.com.ua/article/web-site-creation/60-osobennosti-vospriyatiya-kontenta-sayta.html

Журавлёв А.В. Создание обучающей системы языка программирования php в виде квест-игры. *В статье рассмотрена задача создания обучающей системы языка программирования php представленной в виде веб-приложения(сайта). Проект реализован с помощью чистого php. Основываясь на анализе популярных сайтов по php предложены основные функциональные возможности веб-приложения – курс обучения в виде увлекательных лекций, подкрепленных интересными задачами и тестами в формате квеста-игры для людей с базовыми знаниями о веб разработке.*

Ключевые слова: *язык программирования, php, веб-приложение, обучающая система, сайт, квест-игра.*

Zhuravlev A.V. Creating a learning system of the php programming language in the form of a quest game*. The article describes the task of creating a teaching system of the php programming language presented in the form of web applications (website). The project is implemented using php. Based on the analysis of popular sites on php offered the basic functionality of web applications - a course in the form of exciting lectures, supported by interesting tasks and tests in the format of a quest game for people with basic knowledge of web development.*

Keywords: *programming language, php, web application, training system, website, quest game.*

УДК 004.65

Разработка базы данных для хранения документов на предприятии

Кутелёв Р.С., Рычка О.В.
Донецкий национальный технический университет
rokut98gmail.com, red_rose_2005@mail.ru

Кутелёв Р.С. Рычка О.В. Разработка базы данных для хранения документов на предприятии. *В статье представлен краткий обзор разработки информационной системы для хранения документов на предприятии на примере "Базы данных для фондов с документами" Рассмотрены основные требования для системы, структуры базы данных, разработана серверная и клиентская часть.*

Ключевые слова: *база данных, веб-приложение, ASP.NET, C#, MS SQL 2008, документы.*

Введение

В наше время огромное количество фирм используют персональные компьютеры для сохранения и обработки любого вида информации. Эта информация содержится в базах данных [1]. Базы данных играют важную роль в развивающемся мире технологий. Всё, с чем мы каждый день взаимодействуем в жизни, по всей видимости, зафиксировано в какой-нибудь базе. Базы данных формируются и работают под управлением специальных программных средств, называемых системами управления базами данных.

Множество современных организаций не могут обойтись без базы данных для документов. Это учебные заведения, банки, магазины, заводы, любые предприятия и государственные учреждения. Они используют их для перевода данных в электронный вид и объединения данных, а также оперативного доступа к ним. Это позволяет экономить время и средства на затраты.

Самая главная задача развития информационных технологий в приобретении той или иной организацией исключительно новых качеств, придающих ей существенную конкурентоспособность. А это дорогого стоит.

Актуальность темы заключается в том, что в новых системах управления базами данных есть функция не только хранения данных в своих структурах, однако можно и сохранять программный код, при поддержки которого и идёт взаимодействие с пользователем или программно-аппаратным средством. Рассмотрим на примере базы данных для фондов с документами, определяющими право на собственность.

Характеристики системы

Системой является сервер базы данных MS SQL Server [2] и клиент, веб-приложение ASP.NET [3]. Функциональные характеристики системы:

- стабильная работа, обработка всех возможных ошибок, в случае ошибки сообщение с подробным описанием об ошибке;
- качественный, интуитивно понятный интерфейс;
- обеспечение разграничения доступа пользователей к данным. Доступ к таблицам с пользователями имеет только администратор. Каждый пользователь может просматривать только записи только своего подразделения;
- обеспечение хранения отсканированных копий документов. безопасна, имеет системы защиты от несанкционированного доступа к данным;
- контролируема, присутствует логирование в системе, возможна настройка резервного копирования, для возможности восстановления базы данных в случае сбоя;
- обеспечивает возможность сбора данных с других подразделений (синхронизация с другими серверами).

Проектирование и реализация

Рассмотрим концептуальную модель базы данных данной предметной области, а также все компоненты, которые были реализованы на сервере базы данных.

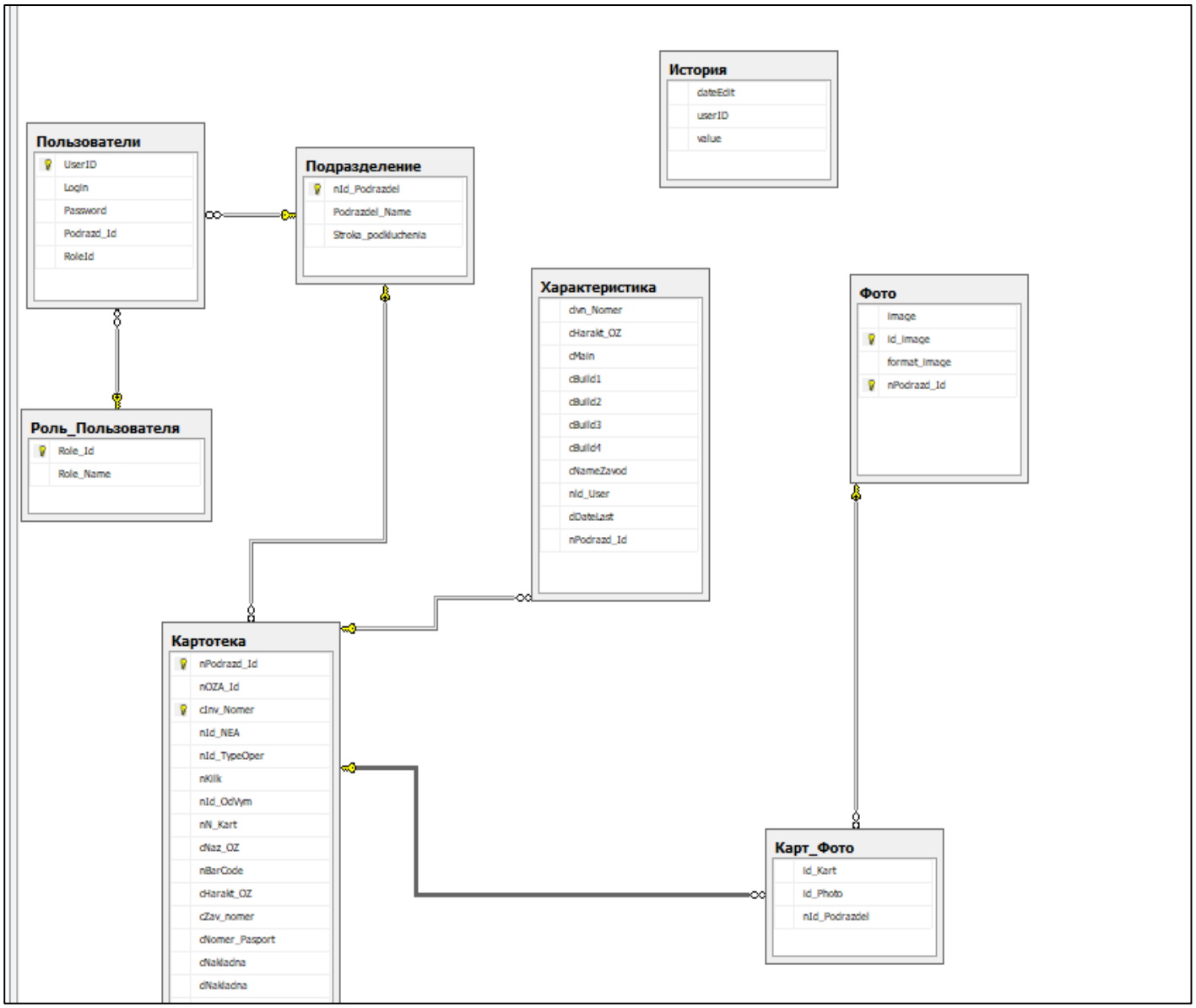


Рисунок 1 – Схема разрабатываемой базы данных

На данном рисунке 1 изображена модель, которая уже была приведена в третью нормальную форму, была изменена для улучшения скорости выполнения запросов. Определены базовые таблицы и таблицы-справочники. Указы первичные и внешние ключи, типы и атрибуты полей таблицы. База данных состоит из 8 таблиц, из них 2 таблиц-справочников и одна специального назначения.

1. В таблице “Фото” содержится вся отсканированные копии документов. Сканированные копии документов хранятся в формате varbinary[4] с атрибутом FILESTREAM[5]. Такой способ позволяет размещать данные больших двоичных объектов типа varbinary в файловой системе в виде файлов.
2. В таблице “Характеристика” содержится вся информация о объектах в картотеке, которые присутствуют в базе данных.
3. В таблице “Подразделения” содержится вся информация о подразделения, которые присутствуют в базе данных.
4. В таблице “Картотека” содержится вся информация о объектах в базе данных.
5. Таблица “Карт_фото” необходима для реализации связи “многие-ко-многим”, т.к. один объект в таблице “Картотека” может иметь несколько отсканированных копий документов и одно изображение в таблице “Фото” может быть прикреплено к нескольким документам.
6. В таблице “Пользователи” содержатся все информация о пользователях в системе, их права доступа, данные для входа. Доступ к данной таблице имеют только администраторы. Пароли хранятся в хешированном виде.
7. В таблице-справочнике “Роль_пользователя” содержится информация о возможной роли пользователя (администратор, обычный пользователь). Доступ к данной таблице имеют только администраторы.

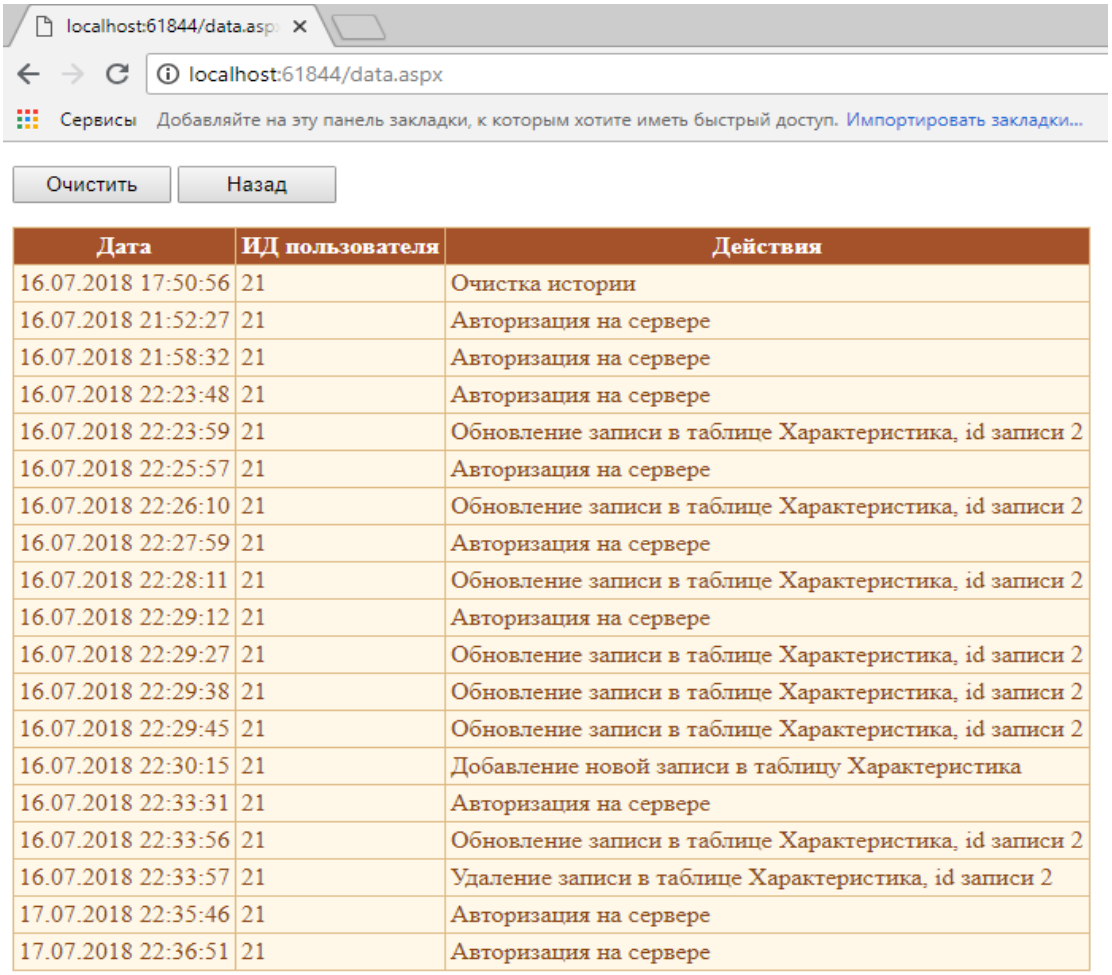
8. В таблице-справочнике “История” содержится информация о всех действия пользователей.

При разработке системы были использованы компоненты SqlConnection [6], SqlCommand, SqlDataReader из библиотеки System.Data.SqlClien [8] для работы с базой данных.

Connection подключает пользователя к базе данных, поэтому в программе присутствует лишь один объект.

Объекты SqlCommand выполняют запросы к базе данных, возвращает одну или несколько строк из запроса [7] (для этого используется функция ExecuteReader (String str).

Для отображения данных, полученных с запроса используется компоненты GridView и DetailsView.



Дата	ИД пользователя	Действия
16.07.2018 17:50:56	21	Очистка истории
16.07.2018 21:52:27	21	Авторизация на сервере
16.07.2018 21:58:32	21	Авторизация на сервере
16.07.2018 22:23:48	21	Авторизация на сервере
16.07.2018 22:23:59	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:25:57	21	Авторизация на сервере
16.07.2018 22:26:10	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:27:59	21	Авторизация на сервере
16.07.2018 22:28:11	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:29:12	21	Авторизация на сервере
16.07.2018 22:29:27	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:29:38	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:29:45	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:30:15	21	Добавление новой записи в таблицу Характеристика
16.07.2018 22:33:31	21	Авторизация на сервере
16.07.2018 22:33:56	21	Обновление записи в таблице Характеристика, id записи 2
16.07.2018 22:33:57	21	Удаление записи в таблице Характеристика, id записи 2
17.07.2018 22:35:46	21	Авторизация на сервере
17.07.2018 22:36:51	21	Авторизация на сервере

Рисунок 2 – Форма для отображения таблицы “История”

В программе представлены формы для редактирования (изменение, добавление, удаление, выборка) всех таблиц. Так же представлена форма авторизации, регистрации нового пользователя.

Для работы с данными были использованы различные динамические массивы, в частности для выполнения поиска, фильтрации и глобальной замены данных.

В формах редактирования, пользователю предоставляется несколько компонентов для изменения данных: TextBox (для заполнения названий, первичных ключей, номера телефона предприятия, названий предприятий, инвентарного номера), ComboBox (для выбора внешнего ключа из другой таблицы, таким образом, обезопасив пользователя выбрать не то данное, а также удобно в использовании), Button для инициирования обновления записи из таблицы новыми данными и для изменения даты, Label для отображения названий и ошибок.

Тестирование

После создания программного продукта клиент-сервер, было проведено тестирование приложения на наличие ошибок. Было проведено тестирование на защиту от несанкционированного доступа, то есть проверка работоспособности пароля пользователя, проверка доступа к таблицам “пользователи” и “роль_пользователя”, а также проведено тестирование одновременной работы с данными двух пользователей. Была проведена проверка прав на редактирование данных обычных пользователей. Проверено каскадное удаление записей в программе, было проведено тестирование правила RESTRICT, а также добавление записей с правилом UNIQUE, NOT NULL,

а также каскадное обновление с правилом CASCADE. Была проверена возможность загрузки изображений в базу данных и их дальнейшую выборку. Был выполнен тестовый сбор данных с другого сервера.

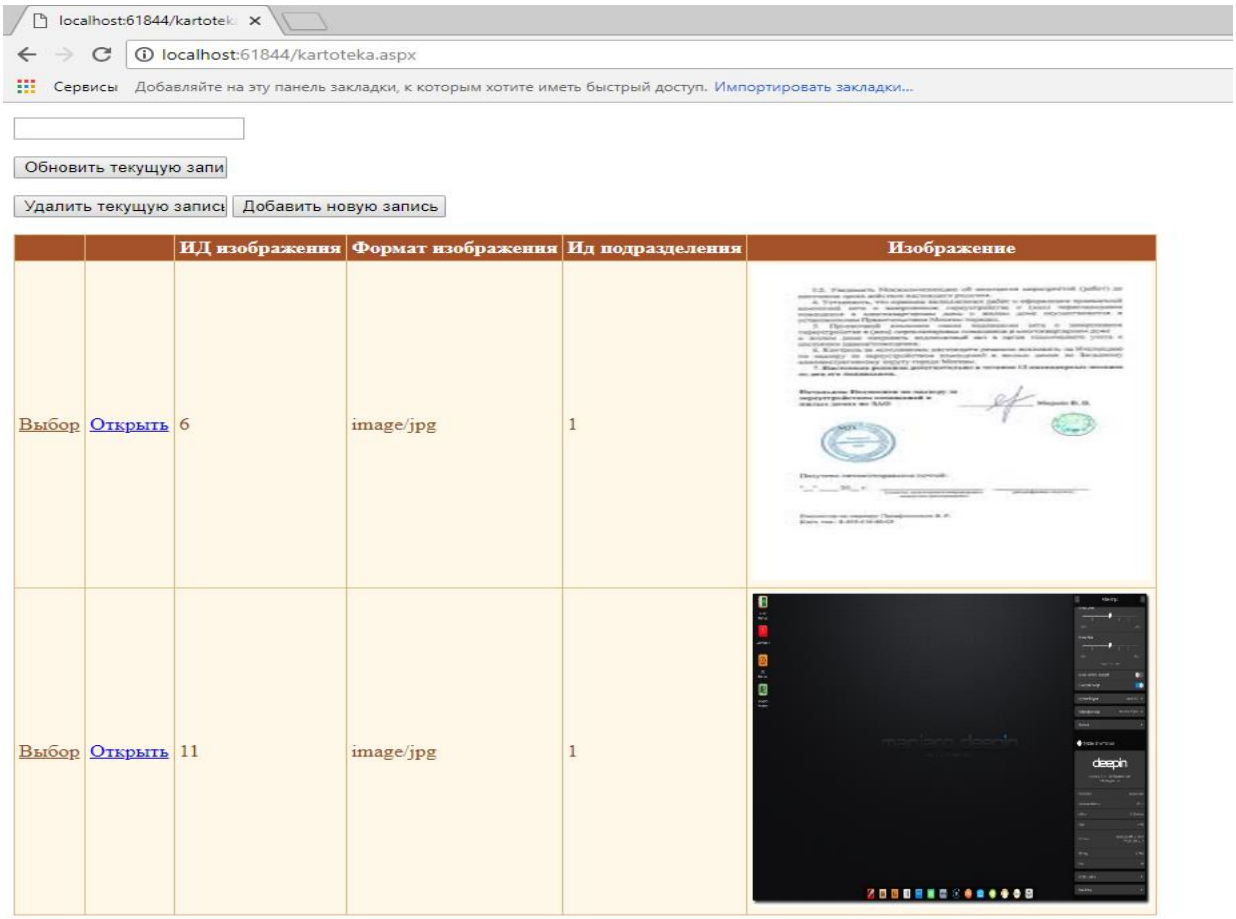


Рисунок 2 – Форма для отображения таблицы “Фото”

Выводы

В результате выполнения работы были рассмотрены выявлены основные характеристики системы, а также на основе их функциональности были разработаны требования к создаваемой системе. Была разработана модель системы, предусматривающая основные требования к системе (спроектирована база данных серверной части, разработано клиентское веб-приложение). Дальнейшими направлениями разработки будут: расширение функциональности системы, общее повышение производительности.

Литература

1. База данных [Электронный ресурс] // Определение базы данных. – 2011. – Режим доступа: https://ru.wikipedia.org/wiki/База_данных, свободный. - Загл. с экрана.

2. MS SQL [Электронный ресурс] // Полнятие и возможности MS SQL. – 2015. – Режим доступа: https://ru.wikipedia.org/wiki/Microsoft_SQL_Server, свободный. - Загл. с экрана.

3. ASP.NET [Электронный ресурс] // Полнятие и возможности. ASP.NET– 2014. – Режим доступа: https://ru.wikipedia.org/wiki/ ASP.NET, свободный. - Загл. с экрана.

4. Varbinary [Электронный ресурс] // Определение типа данных. – 2017. – Режим доступа: https://docs.microsoft.com/en-us/sql/t-sql/data-types/binary-and-varbinary-transact-sql?view=sql-server-2017, свободный. - Загл. с экрана.

5. SQL Server Connection [Электронный ресурс] // Подключение к серверу. – 2012. – Режим доступа: https://msdn.microsoft.com/en-us/library/jj653752(v=vs.110).aspx.- Загл. с экрана.

6. FILESTREAM SQL [Электронный ресурс] // Хранение данных в файловой системе. – 2017. – Режим доступа: https://docs.microsoft.com/ru-ru/sql/relational-databases/blob/filestream-sql-server?view=sql-server-2017.- Загл. с экрана.

7. Запрос [Электронный ресурс] // Определение и виды запросов. – 2014. – Режим доступа: http://citforum.ru/operating_systems/sos/glava_7.shtml, свободный. - Загл. с экрана.

8. SqlDataSource [Электронный ресурс] // Определение SqlDataSource. – 2013. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/system.web.ui.webcontrols.sqldatasource(v=vs.110).aspx, свободный. - Загл. с экрана.

Кутелёв Р.С. Рычка О.В. Разработка базы данных для хранения документов на предприятии. *В статье представлен краткий обзор разработки информационной системы для хранения документов на предприятии на примере "Базы данных для фондов с документами" Рассмотрены основные требования для системы, структуры базы данных, разработана серверная и клиентская часть.*

Ключевые слова: *база данных, веб-приложение, ASP.NET, C#, MS SQL, документы.*

Kutelyov R.S. Rychka O.V. Development of a database for storage of documents at the enterprise. *In article the brief review of development of information system for storage of documents at the enterprise on an example "Databases for funds with documents" is presented are considered fundamental requirements for system, structures of a database, the server and client part is developed.*

Ключевые слова: *database, web application, ASP.NET, C#, MS SQL, documents.*

УДК 004.93

Сравнение популярных библиотек компьютерного зрения для использования в приложении по распознаванию транзисторов

Макогон С.А., Зори С.А.
Донецкий национальный технический университет
makogon_94@mail.ru, ik.ivt.rec@gmail.com

Макогон С.А., Зори С.А. Сравнение популярных библиотек компьютерного зрения для использования в приложении по распознаванию транзисторов. *В данной статье произведен обзор и сравнение библиотек OpenCV, AForge.NET, LTI и VXL для использования в приложениях по распознаванию образов и доказана целесообразность использования библиотеки OpenCV для приложений по распознаванию транзисторов.*

Ключевые слова: *компьютерное зрение, распознавание образов, библиотека, OpenCV, AForge.NET, LTI, VXL, документация*

Введение

В настоящее время у разработчиков программного обеспечения существует большой интерес к отрасли компьютерного зрения. Это обусловлено это тем, что данная технология может быть использована в разнообразных прикладных областях и гибко настроена под выполнение определенных типовых заданий.

Так как сообщество разработчиков постоянно развивается, на данный момент существует определенное количество библиотек для разных языков программирования, подстроенных для выполнения типовых классов задач. В статье произведен анализ и сравнение наиболее популярных на сегодняшний день библиотек компьютерного зрения и рассмотрена возможность их применения для решения практических задач по распознаванию транзисторов.

Библиотека OpenCV

OpenCV (Open Source Computer Vision Library) представляет собой библиотеку программного обеспечения для компьютерного зрения с открытым исходным кодом и компьютерного обучения. OpenCV была создана для обеспечения общей инфраструктуры приложений для компьютерного зрения и ускорения использования восприятия машины в коммерческих продуктах. Будучи лицензированным BSD продуктом, OpenCV упрощает бизнес для использования и модификации кода [1].

Целью разработки данной библиотеки является повышение эффективности вычислений в приложениях реального времени. Язык C, на котором была написана библиотека, является оптимизированным. Библиотека OpenCV способна использовать многоядерные процессоры. В случае, если понадобится автоматическая оптимизация на различных аппаратных платформах Intel, возможно дополнительное приобретение библиотеки IPP, с английского Integrate Performance Primitives. В состав данной библиотеки входят процедуры с низкоуровневой оптимизацией, которые могут применяться для разнообразных алгоритмических областей. В спектр возможностей OpenCV входит автоматическое применение IPP во время выполнения какой-либо программы. На рис. 1 приведена диаграмма, сравнивающая быстродействие библиотек OpenCV, OpenCV + IPP, VXL и LTI [2].

В библиотеке более 2500 оптимизированных алгоритмов, которые включают в себя полный набор классических и современных алгоритмов компьютерного зрения и машинного обучения. Эти алгоритмы могут использоваться для обнаружения и распознавания лиц, идентификации объектов, классификации действий человека в видео, отслеживания движения камеры, отслеживания движущихся объектов, извлечения 3D-моделей объектов, создания 3D-облаков точек из стереокамер, сшивания изображений вместе для получения изображения высокого разрешения для всей сцены, найти похожие изображения из базы данных изображений, удалять красные глаза с изображений, сделанных с помощью вспышки, следить за движениями глаз, распознавать декорации и устанавливать маркеры, чтобы накладывать их на дополненную реальность и т. д. OpenCV имеет более 47 тысяч пользователей сообщества и предполагаемое количество загрузок, превышающих 14 миллионов. Библиотека широко используется в компаниях, исследовательских группах и правительственных органах.

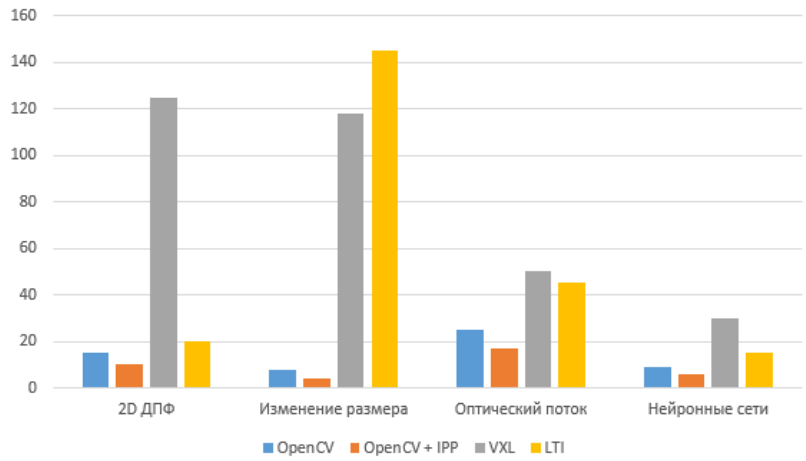


Рисунок 1 – Сравнение библиотек компьютерного зрения

Наряду с известными компаниями, такими как Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, которые используют библиотеку, есть много стартапов, таких как Applied Minds, VideoSurf и Zeitera, которые широко используют OpenCV. Развернутые применения OpenCV охватывают диапазон от сшивания изображений уличного обзора вместе, обнаруживая вторжения в видео наблюдения в Израиле, контролируя оборудование шахт в Китае, помогая роботам перемещаться и собирать объекты в Willow Garage, обнаруживать несчастные случаи в бассейнах в Европе, запускать интерактивное искусство в Испании и Нью-Йорке, проверяя взлетно-посадочные полосы на наличие мусора в Турции, проверяя этикетки на продуктах на заводах по всему миру, чтобы быстро обнаружить лицо в Японии.

Библиотека имеет интерфейсы для C++, Python, Java и MATLAB и поддерживает Windows, Linux, Android и Mac OS. OpenCV ориентируется в основном на приложения в режиме реального времени и использует преимущества команд MMX и SSE, когда они доступны. В настоящее время активно развиваются полнофункциональные интерфейсы CUDA и OpenCL. Существует более 500 алгоритмов и около 10-кратного числа функций, которые составляют или поддерживают эти алгоритмы. OpenCV написан на языке C ++ и имеет шаблонный интерфейс, который работает без проблем с контейнерами STL. На рис. 2 представлена общая структура проекта OpenCV.

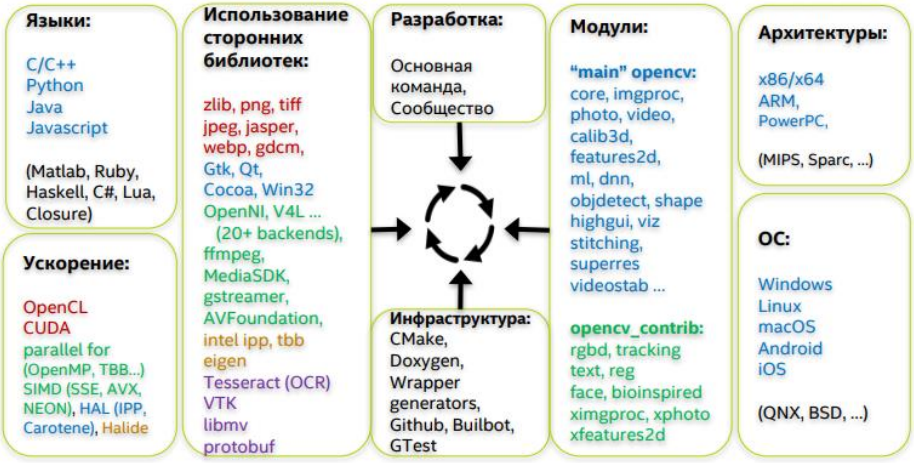


Рисунок 2 – Карта проекта OpenCV

Библиотека AForge.NET

AForge.NET является библиотекой с открытым исходным кодом, созданной на языке C#, которая предназначена для разработчиков и исследователей в области компьютерного зрения. Кроме того, в библиотеке есть функционал для разработчиков в области искусственного интеллекта. Спектр возможностей библиотеки довольно широк: обработка изображений, нейронные сети, генетические алгоритмы, нечеткая логика, машинное обучение, робототехника и многое другое [3].

Библиотека включает несколько основных компонентов. AForge.Imaging — библиотека подпрограмм для обработки изображений и фильтров. AForge.Vision — библиотека компьютерного зрения. AForge.Video — набор библиотек для работы с видеоинформацией. AForge.Neuro — библиотека для выполнения разнообразных

действий и операций с нейронными сетями. AForge.Genetic — библиотека подпрограмм для использования генетических алгоритмов для решения различных задач. AForge.Fuzzy — библиотека для работы с нечеткой логикой. AForge.Robotics — библиотека, обеспечивающая поддержку некоторых методов, применяемых в сфере робототехники. AForge.MachineLearning — библиотека для работы с элементами машинного обучения [4] AForge.NET постоянно улучшается и прогрессирует. По данной библиотеке существует большое количество примеров, демонстрирующих ее работу, и html-документация, которая поддерживается в актуальном состоянии и помогает разработчикам в использовании данного фреймворка. Существует, так же как и у библиотеки OpenCV, активное сообщество, в котором можно подчерпнуть необходимую информацию, задавая вопросы разработчикам, или поделиться собственными наработками. Но, к сожалению, количество участников данного сообщества уступает своим количеством аналогичным по OpenCV. Еще одним ограничением на пути разработчика является тот факт, что вся документация по библиотеке написана только на английском языке. Ввиду этого возможны трудности в изучении и освоении данного фреймворка. На рис. 3 приведена общая структура библиотеки AForge.NET.

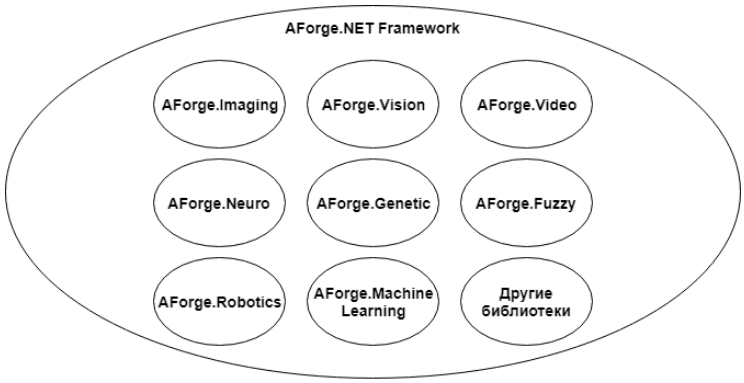


Рисунок 3 – Структура фреймворка AForge.NET

Библиотека VXL

VXL — это набор библиотек, написанных на языке C++, которые предназначены для научных исследований и реализации технологий компьютерного зрения [5]. VXL была написана в ANSI/ISO C++ и предназначена для портативных платформ. Библиотека состоит из нескольких основных составляющих: VNL (числа) — численные алгоритмы и контейнеры, например, матрицы, векторы, оптимизаторы и т.д., VIL (изображения) — загрузка, сохранение и редактирование изображений во многих наиболее распространенных форматах (также существует возможность работы с очень большими изображениями), VGL (геометрия) — геометрия точек, кривых и других элементарных объектов в одно-, двух- и трехмерном пространствах, VSL (входный и выходной потоки), VBL (основные шаблоны), VUL (утилиты) — разный функционал для независимых платформ. Кроме основных библиотек, входящих в состав VXL, существуют и дополнительные. Они отвечают за такие понятия, как численные алгоритмы, обработка изображений, системы координат, геометрия камеры, стерео, манипуляции с видеопотоком, восстановление структуры при движении камеры, графический дизайн, функции отслеживания, топология, классификаторы, 3d визуализация и многое другое. Особенность библиотеки заключается в том, что каждый ее компонент может использоваться отдельно, не ссылаясь на другие компоненты. Таким образом, в приложении можно использовать только то, что действительно необходимо.

VXL используется по всему миру. Библиотека применяется в сфере обучения и промышленности, некоторые ведущие мировые эксперты в сфере компьютерного зрения пользуются данной библиотекой. Существует документация по VXL с описанием каждого класса и функций. Однако у данной библиотеки существует такой же недостаток, как и у AForge.NET. Разработчику, не знающему английский язык будет весьма проблематично освоить и использовать данный набор библиотек. На рис. 4 приведено иерархическое строение ядра VXL [6].

Библиотека LTI

LTI или LTI-lib – объектно-ориентированная библиотека алгоритмов и структур данных. Она часто применяется при обработке изображений и в сфере компьютерного зрения. LTI-lib была разработана как часть научно-исследовательских проектов в области компьютерного зрения с технологиями робототехники, распознавания объектов, голоса и жестов. Основной целью разработки данной библиотеки является создание объектно-ориентированной библиотеки на языке C++, что во многом упрощало бы использование кода и его обслуживание, но при этом были бы обеспечены быстрые алгоритмы, которые можно было бы использовать в реальных приложениях.

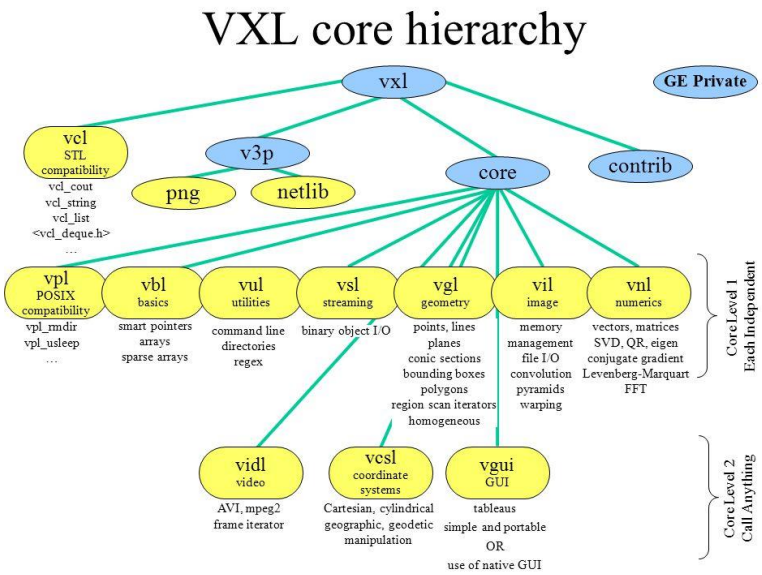


Рисунок 4 – Иерархическое строение ядра VXL

Библиотека была разработана с применением GCC (набор компиляторов, применяемый для разнообразных языков программирования) под Linux и Visual C++ под Windows NT. Многие классы инкапсулируют Windows/Linux функциональность для того, чтобы упростить решение системных или аппаратных задач (например, классы для многопоточности и синхронизации, измерения времени и доступ к последовательному порту) [7].

Библиотека снабжена документацией [8], которая поддерживается разработчиками в актуальном состоянии и может быть найдена на главной странице проекта. Вся информация в ней предоставляется на английском языке, как и в большей части других библиотек. Библиотека LTI, так же как и библиотека OpenCV, является свободно распространяемым программным обеспечением, согласно GNU Lesser General Public License. Существует огромное количество проектов, при разработке которых применялись технологии компьютерного зрения, для которых использовалась именно библиотека LTI.

Система распознавания и классификации изображений транзисторов и выбор библиотеки компьютерного зрения для нее

Для решения типовых проблем распознавания и классификации изображений транзисторов поставлена задача разработки программной системы с клиент-серверной архитектурой.

Основная часть работы приложения будет выполняться на стороне клиента и заключается в распознавании символьной маркировки на транзисторе. Далее, распознанный текст с маркировкой будет передаваться на сервер, на котором находится хранилище (база данных) информации по транзисторам. После поиска соответствий на сервере, информация по транзистору с данной маркировкой возвращается клиенту.

Система должна иметь высокую отказоустойчивость, стабильное подключение к сети Интернет для обмена данными, корректно организованную политику конфиденциальности для доступа к информации на сервере БД. Клиент планируется разрабатывать под платформу Android, как одну из самых распространенных и доступных мобильных систем.

На рис. 5 представлена укрупненная схема работы данной системы.

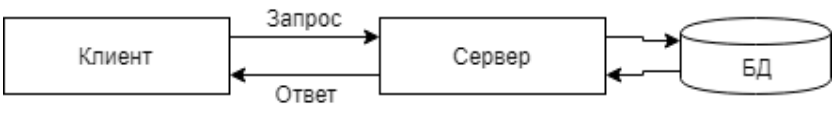


Рисунок 5 – Укрупненная структура системы распознавания и классификации изображений транзисторов

Среди рассмотренных вариантов библиотек компьютерного зрения оптимальным будет использование библиотеки OpenCV, так как она является быстро работающей [1, 2, 5], в ее составе имеются функции, предназначенные не только для распознавания текста, но и для обработки изображения в целом [1, 2, 5], что упрощает структуру будущего приложения и, также имеется реализация библиотеки специально под ОС Android.

Выводы

При анализе данных о приведенных выше библиотеках компьютерного зрения были получены следующие выводы в пользу использования библиотеки OpenCV для разработки приложения по распознаванию транзисторов.

1. Основным преимуществом библиотеки компьютерного зрения, как и любого другого программного обеспечения, является ее производительность. Как видно из диаграммы на рис. 1, производительность OpenCV превышает производительность аналогов (VXL и LTI-lib) даже без использования дополнительного компонента IPP.

2. Библиотека содержит огромное количество функций для решения разнообразных задач, начиная с обработки изображений, компьютерного зрения и заканчивая обучением машин. Кроме того, библиотека имеет открытый исходный код и лицензию, позволяющую использовать весь функционал для разработки коммерческих продуктов.

3. Преимуществом OpenCV является наличие русскоязычной документации, наличие огромнейшего числа туториалов, уроков, научных материалов и книг по применению функционала и методов работы с библиотекой. Нельзя не упомянуть о весьма активном сообществе разработчиков и пользователей библиотеки, которые могут поделиться своим опытом и ответить на интересующие вопросы.

4. Решающим преимуществом в выборе OpenCV стала ее кроссплатформенность и возможность использовать в связке практически с любым языком программирования.

Литература

1. OpenCV library [Электронный ресурс]. – Режим доступа: https://opencv.org/. - Загл. с экрана.
2. OpenCV vs VXL vs LTI: Performance Test - AI Shack[Электронный ресурс]. – Режим доступа: http://www.aishack.in/tutorials/opencv-vs-vxl-vs-lti-performance-test/. - Загл. с экрана.
3. AForge.NET :: Framework [Электронный ресурс]. – Режим доступа: http://www.aforgenet.com/framework/. - Загл. с экрана.
4. Применение библиотеки AForge.NET и ее расширения Accord.NET Framework при распознавании лиц в режиме реального времени | Статья в журнале «Молодой ученый» [Электронный ресурс]. – Режим доступа: https://moluch.ru/archive/154/43602/ . - Загл. с экрана.
5. VXL - C++ Libraries for Computer Vision [Электронный ресурс]. – Режим доступа: http://vxl.sourceforge.net/. - Загл. с экрана.
6. What is VXL ? Vision Something Libraries A collection of Computer Vision libraries Open Source, grass roots effort, 53 developers –Supported by good community, - ppt download [Электронный ресурс]. – Режим доступа: https://slideplayer.com/slide/7433547 . - Загл. с экрана.
7. LTI-Lib [Электронный ресурс]. – Режим доступа: http://ltilib.sourceforge.net/doc/homepage/index.shtml. - Загл. с экрана.
8. LTI-Lib [Электронный ресурс]. – Режим доступа: http://ltilib.sourceforge.net/doc/html/index.shtml. - Загл. с экрана.

Макогон С.А., Зори С.А. Сравнение популярных библиотек компьютерного зрения для использования в приложении по распознаванию транзисторов. *В данной статье произведен обзор и сравнение библиотек OpenCV, AForge.NET, LTI и VXL для использования в приложениях по распознаванию образов и доказана целесообразность использования библиотеки OpenCV для приложений по распознаванию транзисторов.*

Ключевые слова: *компьютерное зрение, распознавание образов, библиотека, OpenCV, AForge.NET, LTI, VXL, документация*

Makogon S.A., Zori S.A. Comparison of popular computer vision libraries for use in transistor recognition applications. *This article reviewed and compared the OpenCV, AForge.NET, LTI and VXL libraries for use in pattern recognition applications and proved the feasibility of using the OpenCV library for transistor recognition applications.*

Keywords: *computer vision, image recognition, library, OpenCV, AForge.NET, LTI, VXL, documentation*

УДК 004.9

Приложение для поиска на местном рынке услуг общественного питания

Махорин С.Н., Коломойцева И.А.
Донецкий национальный технический университет
gvozdevik@gmail.com, bolatiger@gmail.com

Махорин С.Н., Коломойцева И.А. Приложения для поиска. *Приложение для поиска на местном рынке услуг общественного питания. В статье рассмотрена разработка и проектирование приложения для поиска на местном рынке услуг общественного питания. Определены спецификации и этапы разработки*

Ключевые слова: *приложение, поисковик, Android, еда.*

Приложение для поиска на местном рынке услуг общественного питания

Мобильные приложения [1] набирают дикую популярность с каждым годом. В данной отрасли все чаще стараются решить бытовые задачи, такие как вызов такси, поиск еды и оплата коммунальных платежей. С каждым годом мы наблюдаем рассвет приложений, автоматизирующих отрасли нашей жизни, о которых мы даже подумать не могли.

Данная статья акцентирует внимание на проблему, при который было отсутствие легкого доступа к лучшим меню в городе. Конечно, есть Yelp [2], Google+ [3], Locals [4] и многие другие замечательные инструменты, которые просматривают рестораны, но ни один из них не представляет обзор для отдельных блюд. Некоторые из списков даже не имеют соответствующих меню.

В статье рассматривается создание приложения для Android [5], которое служит интерфейсом для пользователей, чтобы внести свой вклад в детали меню. Служба работает с информацией о толпах от наших пользователей так же, как и в Google Maps [6], где каждый из них может представить новые меню, которые они обнаружили.

Для разработки данного приложения из спектра JVM [7] совместимых языков был выбран язык Java [8], данный язык является самым первым JVM-совместимым языком и используется по сей день как в Enterprise [9] приложениях, так и в Android разработке.

В качестве СУБД была выбрана Google Firebase [10] из-за простоты настройки и интеграции в уже существующие Google-сервисы, которые в свою очередь уже предустановлены на подавляющем большинстве Android-устройств. Для автоматизации разработки и системы контроля версий был выбран GitLab [11] который включает в себя несколько DevOps [12] приложений и ликвидирует потребность в установке лишнего софта.

Операционная система Android

Операционная система для смартфонов, планшетов, электронных книг, цифровых проигрывателей, наручных часов, фитнесс-браслетов, игровых приставок, ноутбуков, нетбуков, смартбуков, Очков Google Glass [13], телевизорови других устройств (в 2015 году появилась поддержка автомобильных развлекательных систем и бытовых роботов).

Основана на ядре Linux [14] и собственной реализации виртуальной машины Java от Google. Изначально разрабатывалась компанией Android, Inc., которую затем купила Google. Впоследствии Google инициировала создание альянса Open Handset Alliance (OHA), который сейчас занимается поддержкой и дальнейшим развитием платформы. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки. Android Native Development Kit позволяет портировать библиотеки и компоненты приложений, написанные на Си и других языках.

На рынке мобильных приложений телефоны на базе ОС Android занимают 76,61% рынка мобильных ОС, в отличии от своего конкурента iOS [15].

База данных Firebase

Основной сервис — облачная СУБД класса NoSQL [16], позволяющая разработчикам приложений хранить и синхронизировать данные между несколькими клиентами. Поддержаны особенности интеграции с приложениями под операционные системы Android и iOS, реализовано API для приложений на JavaScript [17], Java, Objective-C [18] и Node.js [19], также возможно работать напрямую с базой данных в стиле REST [20] из

ряда JavaScript-фреймворков, включая AngularJS [21], React [22], Ember.js [23] и Backbone.js [24]. Предусмотрено API для шифрования данных.

Система управления репозиториями GitLab

Веб-менеджер Git [25] репозитория, предоставляющий вики, отслеживание ошибок и возможности CI/CD [26], Gitlab очень быстро развивается. В течении последних нескольких месяцев мы получили отображение CI в реальном времени, первые тестовые имплементации автоматического deploy [27] в kubernetes [28] и множество других небольших улучшений. Огромный комбайн – это не всегда хорошо. Но в случае с GitLab можно предположить, если они будут продолжать развитие продукта, то можно будет получить лучшее CI-решение, бесшовно интегрированное с управлением исходниками и системой работы с задачами.

Приложение для оценок еды Yelp

В качестве аналога данного приложения, можно рассмотреть приложение Yelp, приложение для оценки пользователями еды.

Веб-сайт Yelp, Yelp.com, представляет собой локальный бизнес-обзор и сайт социальной сети, основанный на толпе. Его сообщество пользователей в основном работает в крупных городских районах. На страницах сайта есть страницы, посвященные отдельным местам, например, ресторанам или школам, где пользователи Yelp могут представить обзор своих продуктов или услуг с использованием системы оценки от 1 до 5 звезд. Компании могут также обновлять контактную информацию, часы и другую основную информацию о листинге или добавлять специальные предложения. В дополнение к написанию отзывов пользователи могут реагировать на отзывы, планировать мероприятия или обсуждать свою личную жизнь. Согласно анализу Sterling Market Intelligence [29], Yelp является «одним из самых важных сайтов в Интернете». По состоянию на четвертый квартал 2017 года он имеет 141 миллион ежемесячных уникальных посетителей и 148 миллионов отзывов.

Приложения для знакомств Tinder

В качестве графической оболочки изображения было взято за основу приложение Tinder [30], а конкретно механика карточке в которых пользователь, проводя пальцем влево даёт приложению знать что ему не интересен выбор и вправо если интересен, на основе его выбора формируется на сервере психологический портрет пользователя для того чтобы последующие разы приложение предлагало ему лишь подходящие вещи по его вкусам

Анализ требований

Для разработки приложения была спроектирована диаграмма прецедентов и диаграмма классов. Диаграмму прецедентов можно рассмотреть на рисунке 1.

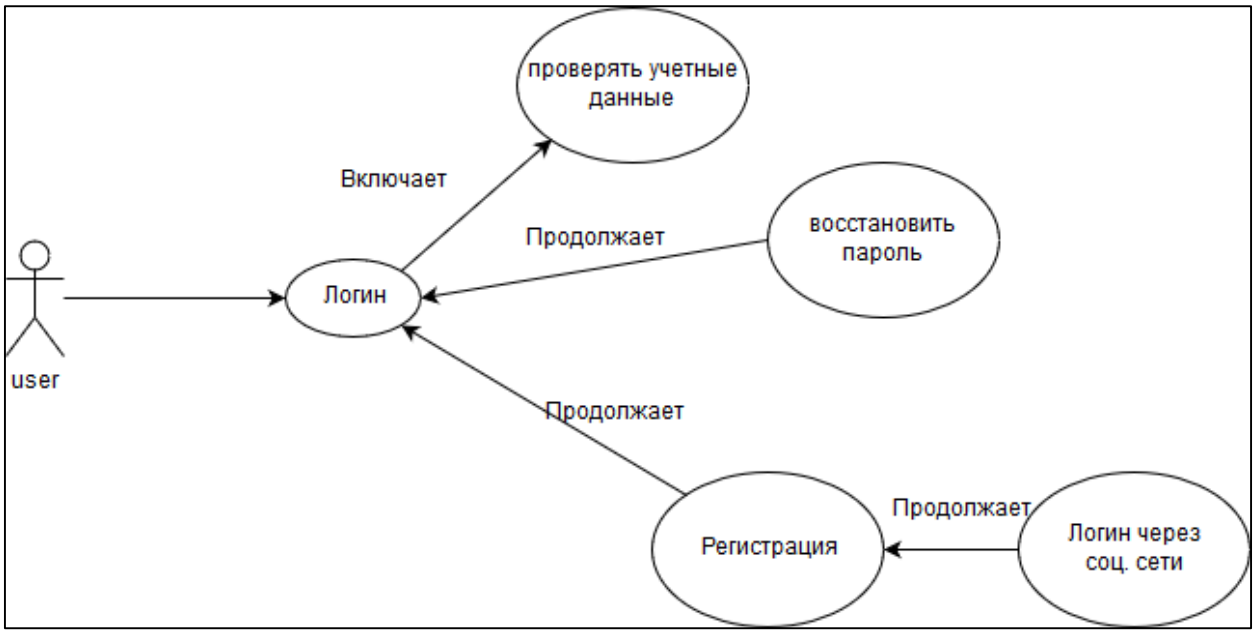


Рисунок 1 – Диаграмма прецедентов

Как можно увидеть, диаграмма прецедентов достаточно лаконичная и не включает себя много элементов, далее рассмотрим диаграмму классов, представленную на рисунке 2.

На диаграмме классов можно рассмотреть минималистический дизайн и минимальное количество зависимостей между классами. В качестве интерфейса за основу было взято приложение Entrée[31] представленное на рис.3.

Выводы

В наше время автоматизация бытовых вещей, таких как заказ такси становится реальностью и для этого в данной статье была рассмотрена разработка приложения для поиска на рынке услуг и анализ аналогов на рынке мобильных приложений. Были рассмотрены этапы разработки и составлены соответствующие диаграммы, так же был приведен пример интерфейса ссылаясь на аналогичный продукт. В дальнейшем статья будет дополняться и в планах выход на локальный рынок услуг.

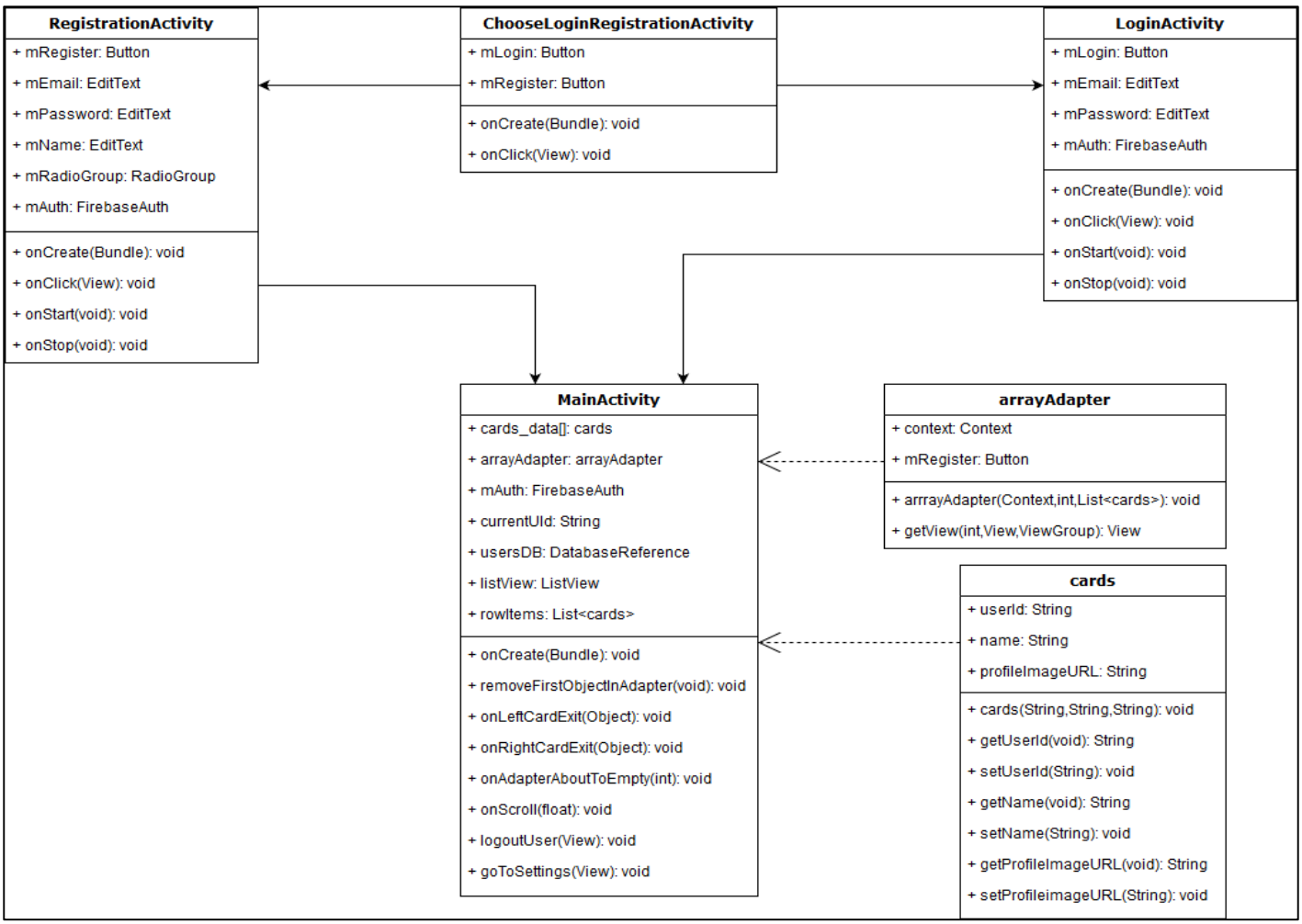


Рисунок 2 – Диаграмма классов

Рисунок 3 – Пример интерфейса [31]

Литература

1. Мобильные приложения // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Мобильное_приложение. – Загл. с экрана.

2. Yelp // Википедия [Электронный ресурс] – Электрон. дан. - 2018. – Режим доступа: http://wikipedia.green/Yelp. - Загл. с экрана.

3. Google+ // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Google+. – Загл. с экрана.

4. Locals // Locals [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://withlocals.com – Загл. с экрана.

5. Android // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://en.wikipedia.org/wiki/Android_(operating_system). – Загл. с экрана.

6. Google Maps // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Google_maps. – Загл. с экрана.

7. JVM // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Java_Virtual_Machine. – Загл. с экрана.

8. Java // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://en.wikipedia.org/wiki/Java_(Programming_Language) – Загл. с экрана.

9. Enterprise // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Enterprise_Software. – Загл. с экрана.

10. Firebase // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Firebase. – Загл. с экрана.

11. GitLab // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/GitLab. – Загл. с экрана.

12. DevOps // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/DevOps. – Загл. с экрана.
13. Google Glass // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Google_glass. – Загл. с экрана.
14. Linux // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Linux. – Загл. с экрана.
15. iOS // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/iOS. – Загл. с экрана.
16. NoSQL // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/NoSQL. – Загл. с экрана.
17. JavaScript // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Javascript. – Загл. с экрана.
18. Objective-C // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Objective-c. – Загл. с экрана.
19. Node.JS // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Node.js. – Загл. с экрана.
20. REST // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/REST. – Загл. с экрана.
21. AngularJS // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://en.wikipedia.org/wiki/Angular_(framework). – Загл. с экрана.
22. React // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/React.js. – Загл. с экрана.
23. Ember.js // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Ember.js – Загл. с экрана.
24. Backbone.js // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Backbone.js. – Загл. с экрана.
25. Git // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Git. – Загл. с экрана.
26. CI/CD // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://en.wikipedia.org/wiki/CI/CD. – Загл. с экрана.
27. Deploy // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://en.wikipedia.org/wiki/Software_Deployment. – Загл. с экрана.
28. Kubernetes // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Kubernetes. – Загл. с экрана.
29. Sterling Market Intelligence // Sterling Market Intelligence – Электрон. дан. – 2018. – Режим доступа: https://sterlingmarketintelligence.com. – Загл. с экрана.
30. Tinder // Википедия [Электронный ресурс] – Электрон. дан. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Tinder . – Загл. с экрана.
31. FoodBeast – Электрон. дан. – 2018. – Режим доступа: https://cdn.foodbeast.com/content/uploads/2016/10/Entree1.png. – Загл. с экрана.

Махорин С.Н., Коломойцева И.А. Приложения для поиска. *Приложение для поиска на местном рынке услуг общественного питания. В статье рассмотрена разработка и проектирование приложения для поиска на местном рынке услуг общественного питания. Определены спецификации и этапы разработки*

Ключевые слова: *Приложение, поисковик, Android, Еда.*

Makhorin S. Kolomoitseva I. Searching App. *This article describes the development process of an application for food searching and analyzes the existing ones. The specifics and steps were determined during the writing of this article*

Keywords*: application, search-engine, Android, food.*

УДК 004.93’12

Логическая модель свёрточной нейронной сети для распознавания лица человека

Медведев А.С., Федяев О.И.
Донецкий национальный технический университет
wwwalex_96@mail.ru, fedyaev@donntu.org

Медведев А.С., Федяев О.И. Логическая модель свёрточной нейронной сети для распознавания лица человека. *В статье рассмотрена задача распознавания лица человека на основе свёрточной нейронной сети. Выполнен анализ параметров структуры свёрточной нейронной сети и модели нейронов для разных слоёв. На основе объектно-ориентированного анализа разработана логическая структура свёрточной нейронной сети в виде диаграммы классов на языке UML.*

Ключевые слова: *свёрточная нейронная сеть, распознавание лиц, диаграмма классов, искусственный нейрон*

Введение

Распознавание образов в настоящее время встречается во многих прикладных задачах. Наиболее актуальной и сложной является задача распознавания лица человека. Необходимость в её решении возникает при проектировании систем контроля за людьми, систем безопасности и многих других [1].

Несмотря на актуальность компьютерного зрения, качественного решения задачи распознавания нет. Основные трудности компьютерного распознавания лиц, которые необходимо преодолеть, состоят в том, чтобы распознать человека по изображению лица независимо от изменения ракурса и условий освещённости при съёмке, а также при различных изменениях, связанных с возрастом, причёской и т. д. [1].

В последнее время большие перспективы в решении данной проблемы связывают с применением глубинных нейронных сетей. К этому классу относится многослойная свёрточная нейронная сеть [1]. В отличие от известных классических архитектур нейронных сетей данный тип построен и функционирует на принципах зрительной системы человека. В последнее время известными фирмами предложены мощные библиотеки, в которых реализованы модели глубоких нейронных сетей, позволяющие решать сложные задачи распознавания. Однако, использование сторонних библиотек может иметь недостаточно оптимизированное решение или нарушать информационную безопасность по отношению к распознаваемым изображениям.

Исходя из этого, в статье рассматривается разработка собственной системы распознавания лиц на основе свёрточной нейронной сети. При этом возникает задача построения такой программной модели нейронной сети, которая позволяла бы исследовать особенности её работы в реальных условиях. Архитектура свёрточной нейронной сети как объект исследования характеризуется наличием большого набора настраиваемых параметров и сложностью функционирования. Поэтому в начале целесообразно провести системный анализ объекта и построить логические модели свёрточной нейронной сети, что и является целью данной работы.

Convolutional Neural Network или свёрточная нейронная сеть (СНС) показала лучшие результаты в области распознавания лиц. Она является развитием идей таких архитектур нейронных сетей, как многослойные сети типа когнитрон и неокогнитрон [2]. Преимущества обусловлены своеобразной архитектурой СНС, позволяющей детально выделять особенности двумерной топологии изображения в отличие от многослойного персептрона.

К достоинствам СНС можно также отнести обеспечение частичной устойчивости к изменениям масштаба, смещениям, поворотам, смене ракурса и прочим искажениям. Свёрточные нейронные сети объединяют в себе три архитектурных идеи:

− локальные рецепторные поля, которые обеспечивают локальную двумерную связность нейронов;

− общие синаптические коэффициенты для обеспечения детектирования некоторых черт в любом месте изображения и уменьшение общего числа весовых коэффициентов;

− иерархическая организация с пространственными подвыборками.

На данный момент свёрточная нейронная сеть и её модификации считаются лучшими по точности и скорости распознавания объектов на изображении.

Структура и функционирование свёрточной нейронной сети

Сверточная нейронная сеть состоит из слоев нескольких видов таких как сверточные (convolutional), подвыборочные (subsampling) и слои персептрона [6]. В общей архитектуре многослойной нейросети свёрточные слои и слои подвыборки, чередуясь между собой, формируют входной вектор признаков для выходного слоя - многослойного персептрона (см. рис. 1). Свое название свёрточная сеть получила по операции с названием «свёртка», суть которой будет описана в следующем разделе.

Основной причиной эффективности СНС является концепция общих весов. Они имеют меньшее количество настраиваемых параметров по сравнению с неокогнитроном. Имеются варианты СНС похожие на неокогнитрон. В таких сетях происходит частичный отказ от связанных весов, но алгоритм обучения остаётся тот же и основывается на обратном распространении ошибки. СНС могут быстро работать на последовательной машине и быстро обучаться за счёт эффективного распараллеливания процесса свёртки по каждой карте, а также обратной свёртки при распространении ошибки по сети.

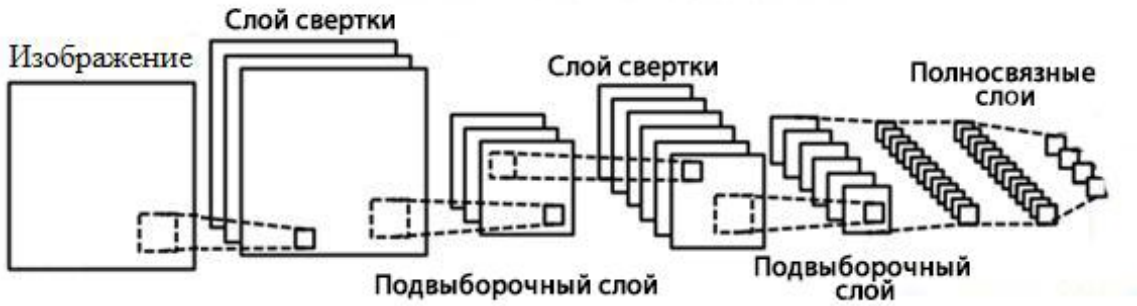


Рисунок 1 – Структура свёрточной нейронной сети

В последние годы в СНС очень часто применяется функция активации под названием «выпрямитель» (см. рис 2). Нейроны с данной функцией активации называются ReLU (rectified linear unit). ReLU имеет следующую формулу $f(x) = max(0, x)$ и реализует простой пороговый переход в нуле [5].

Положительные стороны такой функции:

1. Вычисление сигмоиды и гиперболического тангенса требует выполнения ресурсоёмких операций, таких как возведение в степень, в то время как ReLU может быть реализован с помощью простого порогового преобразования матрицы активаций в нуле. Кроме того, функция ReLU не подвержена насыщению.

2. Применение ReLU существенно повышает скорость сходимости стохастического градиентного спуска по сравнению с сигмоидой и гиперболическим тангенсом. Считается, что это обусловлено линейным характером и отсутствием насыщения данной функции.

Отрицательной стороной этой функции является то, что ReLU не всегда достаточно надёжна при аппаратной реализации и в процессе обучения может выходить из строя. Например, большой градиент, проходящий через ReLU, может привести к такому обновлению весов, что данный нейрон никогда больше не активируется. Если это произойдет, то, начиная с данного момента, градиент, проходящий через этот нейрон, всегда будет равен нулю. Соответственно, данный нейрон будет необратимо выведен из строя. Например, при слишком большой скорости обучения может оказаться, что до 40% нейронов станут «мертвыми» (то есть, никогда не активируются). Эта проблема решается посредством выбора надлежащей скорости обучения.

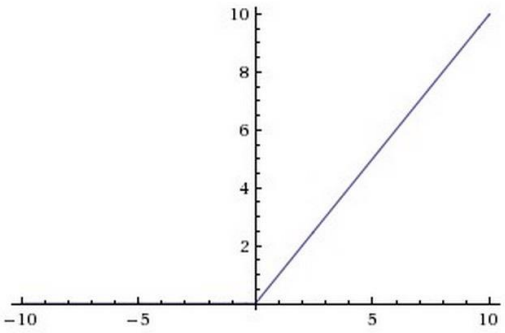


Рисунок 2 – Функция активации ReLU

На этапе анализа структуры СНС возникает важная задача, связанная с определением характеристик её архитектуры: количество слоев, размеры полносвязного слоя, характер связей между нейронами и др.

Функции и параметры свёрточного слоя

Слой свёртки — это основной слой СНС, который выполняет большинство основных вычислений. Он представляет из себя набор матриц (карт), у каждой матрицы есть синаптическое ядро (фильтр).

Количество таких матриц определяется требованиями к задаче, если взять большое количество карт, то повысится качество распознавания, но увеличится вычислительная сложность. Опираясь на анализ научных статей, предлагается брать соотношение один к двум, то есть каждая матрица предыдущего слоя связана с двумя матрицами свёрточного слоя [2].

Размеры всех карт свёрточного слоя одинаковы и вычисляются по формуле:

$$w = mW – kW + 1,$$
$$h = mH – kH + 1,$$

где w – новая ширина свёрточной карты; h – новая высота свёрточной карты; mW – ширина предыдущей карты; mH – высота предыдущей карты; kW – ширина ядра; kH – высота ядра.

Ядро является системой разделяемых весов или синапсов - это одна из главных особенностей свёрточной нейронной сети. Обычная нейронная сеть имеет очень много связей между нейронами, что весьма замедляет процесс детектирования. В свёрточной сети – наоборот, общие веса позволяют сократить число связей, а также находить один и тот же признак по всей области изображения [5].

Ядро представляет из себя матрицу весов, которая скользит по всей области предыдущей карты и находит определённые признаки объектов на изображении. Например, если сеть обучали на множестве лиц, то одно из ядер могло бы в процессе обучения выдавать наибольший сигнал в области глаза, рта, брови или носа, другое ядро могло бы выявлять другие признаки. Размер ядра обычно колеблется в пределах от 3x3 до 7x7 пикселей. Если размер ядра маленький, то оно не сможет выделить какие-либо признаки, если слишком большое, то увеличивается количество связей между нейронами.

Функции и параметры подвыборочного слоя

Подвыборочный слой также как и слой свёртки имеет матрицы и их количество совпадает с предыдущим слоем. Задача слоя состоит в уменьшении размерности карт предыдущего слоя. Если с помощью операции свёртки уже были выявлены некоторые признаки, то для дальнейшей обработки настолько подробное изображение уже не нужно, и оно уплотняется (по определённому принципу) до менее подробного (см. рис. 3).

В процессе сканирования ядром подвыборочного слоя матрицы предыдущего слоя, сканирующее ядро не пересекается в отличие от свёрточного слоя. Обычно, каждая карта имеет ядро размером 2x2, что позволяет уменьшить предыдущие матрицы сверточного слоя в 2 раза. Вся карта признаков разделяется на ячейки 2x2 элемента, из которых выбираются максимальные по значению.

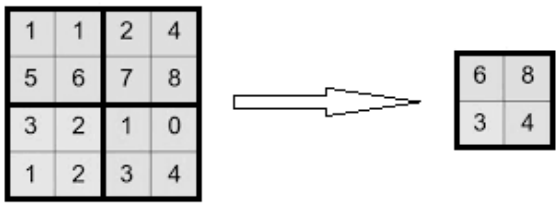


Рисунок 3 – Процесс подвыборки

Структура последнего слоя свёрточной нейронной сети

На выходе СНС установлен слой обычного многослойного персептрона (см. рис. 4). Он служит для определения класса, к которому относится распознаваемый образ лица. Работа слоя организуется путём обращения к выходу предыдущего слоя и определения свойств, которые наиболее характерны для определенного класса [4].

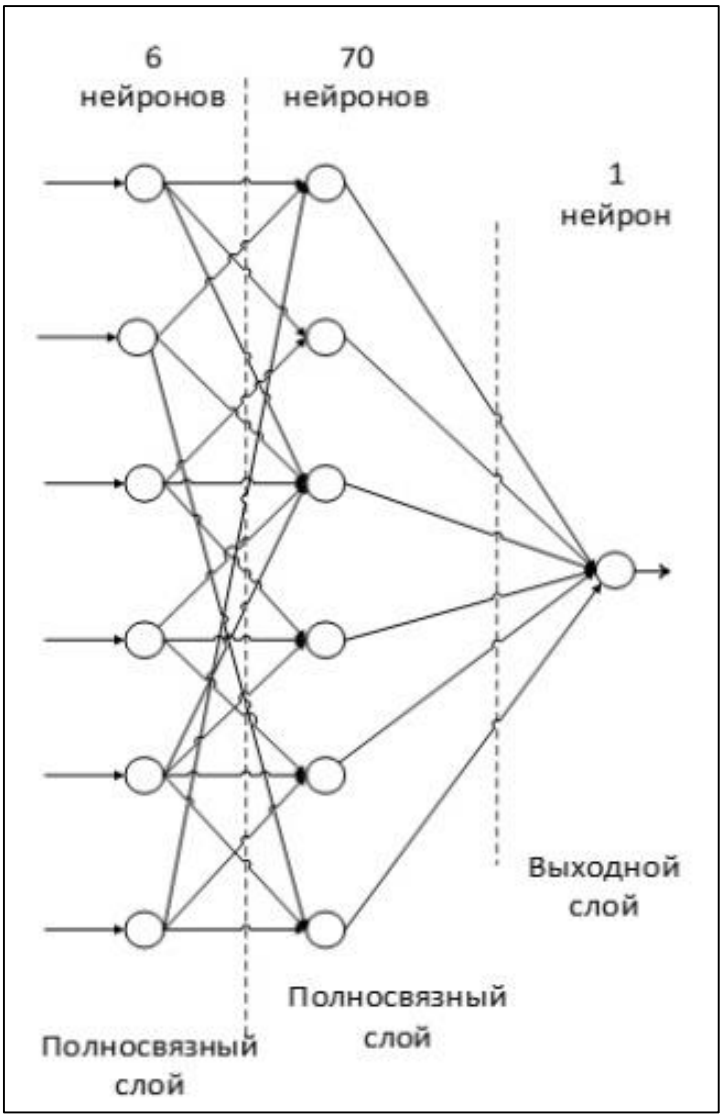


Рисунок 4 – Структура полносвязного слоя

Нейросеть данного слоя является полносвязной, т.е. нейроны каждой карты предыдущего подвыборочного слоя связаны с одним нейроном скрытого слоя. Таким образом, число нейронов скрытого слоя равно числу матриц подвыборочного слоя, но связи могут быть не обязательно полными. Например, только часть нейронов какой-либо из матриц подвыборочного слоя может быть связана с первым нейроном скрытого слоя, а оставшаяся часть - со вторым, либо все нейроны первой карты связаны с нейронами первого и второго скрытого слоя. Вычисление значений нейрона описывается формулой [5]:

$$x^l_j = f(\sum x^{l-1}_i * w^{l-1}_{i,j} + b^{l-1}_j),$$

где x^l_j - карта признаков j (выход слоя l); f() – функция активации; b^l – коэффициент сдвига слоя l; $w^{l-1}_{i,j}$ – матрица весовых коэффициентов первого слоя.

Объектно-ориентированная структура свёрточной нейронной сети

Выполненный анализ параметров СНС, позволил построить логическую модель нейронной сети, которая будет содержать следующие основные классы, представленные на языке UML (см. рис. 5):

- нейронная система – основной класс, который содержит в себе все слои и параметры СНС;
- входной слой – класс, которые отвечает за первоначальную обработку изображения, для дальнейшей

работы в СНС;

- слой свёртки – класс, обеспечивающий функциональные возможности слоя свёртки;
- слой подвыборки – класс, обеспечивающий функциональные возможности слоя подвыборки;
- слой полносвязный – класс, обеспечивающий функционал последнего слоя нейронной сети;
- активация – класс, содержащий различные функции активации;
- параметры – класс, содержащий в себе все настройки нейронной сети.

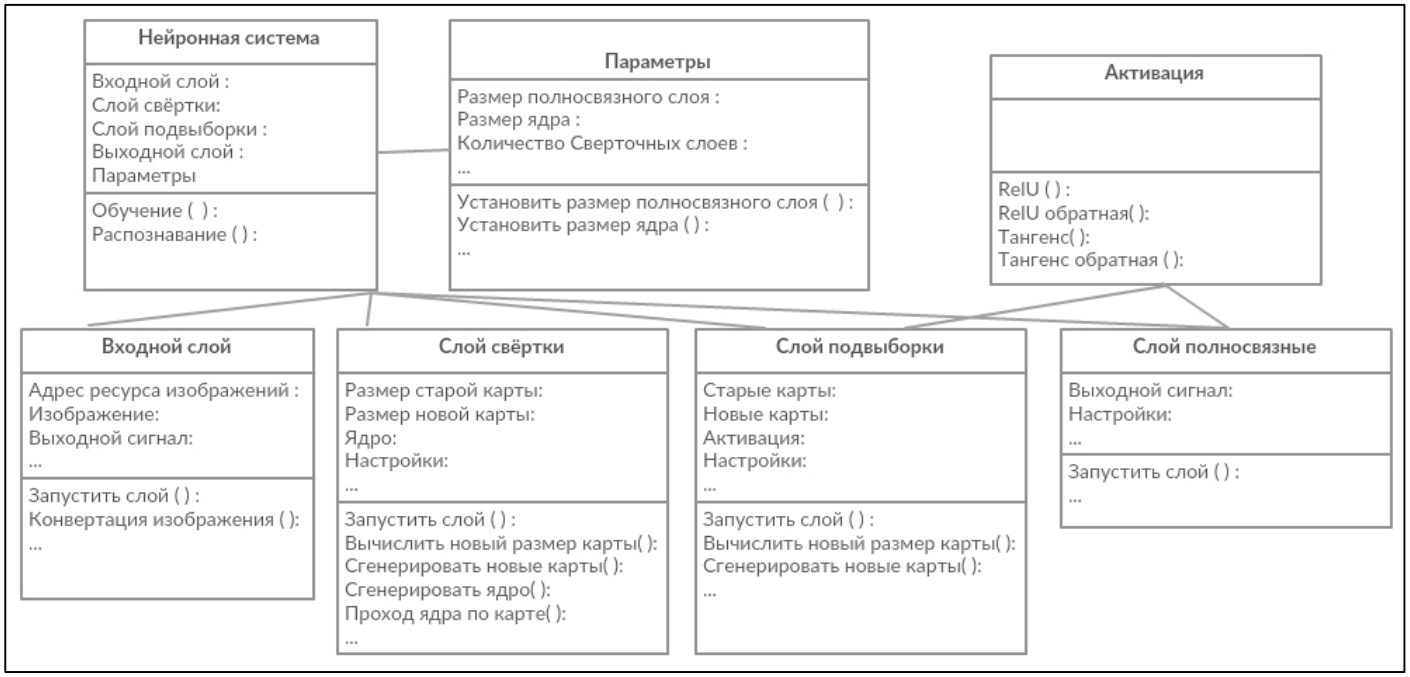


Рисунок 5 – Диаграмма классов свёрточной нейронной сети

Заключение

В работе проведен параметрический и функциональный анализ архитектуры свёрточной нейронной сети. Установлено, что она имеет ряд важных достоинств: ускоренное обучения сети и уменьшение необходимого количества обучающих данных за счёт уменьшенного количества обучаемых нейронов и за счёт большого количества абстрактных слоёв. Слои обеспечивают частичную устойчивость к изменениям масштаба, смещениям, поворотам, смене ракурса и прочим искажениям. Архитектура СНС позволяет осуществить распараллеливание вычислений, а, следовательно, имеется возможность реализации алгоритмов работы и обучения сети на графических процессорах. Но при этом у СНС есть и недостатки: слишком много варьируемых параметров сети.

Литература

1. Как работает свёрточная нейронная сеть: архитектура примеры, особенности [Электронный ресурс]. – Режим доступа: https://neurohive.io/ru/osnovy-data-science/glubokaya-svertochnaja-nejronnaja-set/

2. Применение нейросетей в распознавании изображений. [Электронный ресурс]. https://habr.com/post/74326/

3. Солдатова О.П., Гаршин А.А. Применение сверточной нейронной сети для распознавания рукописных цифр.

4. Прохоров В.Г. Использование сверточной нейронной сети для распознавания рукописных символов.

5. Что такое свёрточная нейронная сеть [Электронный ресурс]. – Режим доступа: https://habr.com/post/309508/

6. Свёрточные нейронные сети: взгляд изнутри [Электронный ресурс]. – Режим доступа: http://ru.datasides.com/code/cnn-convolutional-neural-networks/

7. Свёрточная нейронная сеть [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%91%D1%80%D1%82%D0%BE%D1%87%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0

%B5%D1%82%D1%8C

Медведев А.С., Федяев О.И. Логическая модель свёрточной нейронной сети для распознавания лица человека. *В статье рассмотрена задача распознавания лица человека на основе свёрточной нейронной сети. Выполнен анализ параметров структуры свёрточной нейронной сети и модели нейронов для разных слоёв. На основе объектно-ориентированного анализа разработана логическая структура свёрточной нейронной сети в виде диаграммы классов на языке UML.*

Ключевые слова: *свёрточная нейронная сеть, распознавание лиц, диаграмма классов, искусственный нейрон*

Medvedev A.S., Fedyaev O.I. A convolutional neural network logical model for recognizing human faces. *The article deals with the problem of recognizing a human face based on a convolutional neural network. Performs an analysis of the parameters of the structure of the convolutional neural network and the model of neurons for different layers. Based on object-oriented analysis and the logical structure of the convolutional neural network in the form of a class diagram in the UML language.*

Keywords: *convolutional neural network, face recognition, class diagram, artificial neuron*

УДК 004.932.72

Представление виртуальной кафедры на уровне визуальных моделей многоагентной среды Jack

Московченко А.В., Жабская Т.Е., Федяев О.И.
Донецкий национальный технический университет
asya.frech@gmail.com, tanya_zhabskaya@mail.ru, fedyaev@donntu.org

Московченко А.В., Жабская Т.Е., Федяев О.И. Представление виртуальной кафедры на уровне визуальных моделей многоагентной среды Jack. *Статья посвящена проектированию виртуальной кафедры университета. Агентно-ориентированный анализ учебной кафедры выполнен с помощью методологии Gaia. Для программной реализации агентов с BDI-архитектурой выбрана инструментальная среда Jack. Рассмотрена трансформация логических моделей Gaia в визуальные модели среды Jack.*

Ключевые слова: *многоагентные системы, виртуальная кафедра, методология Gaia, инструментальная среда Jack, агентно-ориентированное проектирование*

Введение

Формы получения высшего образования, используемые в настоящее время, не позволяют своевременно реагировать на изменение требований рынка труда, в полной мере учитывать индивидуальные возможности и желания студентов в освоении дисциплины за ускоренное время (экстерном). Они предполагают обязательное личное присутствие преподавателя на всех этапах передачи и контроля усвоения знаний, жесткую привязку студентов к расписанию занятий. Студенту не всегда разрешают работать по индивидуальному графику, поскольку без общения с преподавателем, несмотря на полноценное учебно-методическое обеспечение, практически сложно получить хороший уровень знаний по предметам. Несмотря на эту проверенную опытом форму обучения, современная тенденция в инженерном образовании характеризуется внедрением индивидуальных образовательных схем, в полной мере отвечающих быстрым изменениям конъюнктуры рынка. Поэтому классические схемы централизованного управления образованием с жесткой структурой должны позволять трансформироваться в более гибкие схемы [1-4].

В этой связи, учебные заведения вынуждены расширять формы образовательной деятельности, т.е. технологии профессиональной подготовки студентов. В наши дни активно происходит расширение образовательных платформ, формируются электронные информационно-образовательные среды учебных заведений, создаются новые модели образовательных организаций – виртуальные кафедры, появление которых полностью обусловлено развитием информационно-коммуникационных технологий [5]. Виртуальная кафедра предоставит студентам и преподавателям возможность взаимодействовать в процессе обучения через компьютерную сеть, что позволит внедрить в вузы дистанционное образование и различные формы онлайн-обучения профессиональным навыкам. Виртуальная образовательная среда становится всё более значимым социальным явлением реальной действительности [6].

Цель данной статьи – рассмотреть учебную кафедру университета как объект виртуальной образовательной среды, представляющей собой информационное пространство взаимодействия участников учебного процесса, порождаемое технологиями информации и коммуникации, включающее комплекс компьютерных средств и технологий, позволяющее осуществлять управление содержанием образовательной среды и коммуникацию участников [6]. Чтобы полноценно реализовать интеллектуальные функции участников и их коммуникацию, виртуальную кафедру целесообразно представить в виде многоагентной программной системы, основанной на принципах распределенного искусственного интеллекта [5]. Преимущество виртуальной кафедры, как новой среды обучения, состоит в следующем:

– гибкость, обучаемый имеет возможность заниматься в удобном для себя месте, т.к. цикл обучения осуществляется посредством Интернет-технологий;

– модульность, обучаемый имеет возможность из набора независимых курсов-модулей формировать учебную программу, отвечающую индивидуальным потребностям;

– экономическая эффективность, снижаются затраты как обучающегося, так и системы образования за счёт максимально эффективного использования учебных площадей, времени и технических средств [6].

–

Построение логических моделей виртуальной кафедры с помощью методологии Gaia

Процесс обучения, в соответствии с методологией Gaia, описывается следующими моделями: моделью ролей, моделью взаимодействий, моделью агентов, моделью услуг, моделью связей. На рис. 1 показаны взаимосвязи и содержание моделей, полученных с помощью агентно-ориентированного проектирования виртуальной кафедры.

На стадии агентно-ориентированного анализа и проектирования разработаны следующие модели: модель ролей для описания должностных обязанностей всех ролей в виде активностей, протоколов, полномочий, обязательств; модель взаимодействий для описания основных видов общения между ролями в виде протоколов; модель агентов для определения типов агентов; модель функционирования для определения действий агентов и модель связей для отображения возможных коммуникаций между агентами [7].

Поскольку программным агентам делегируется выполнение полномочий субъектов образовательного процесса, то они должны имитировать взаимодействия, которые в определённой степени соответствуют их профессиональной деятельности. Для имитации профессиональной деятельности каждый программный агент должен обладать знаниями о порученных должностных обязанностях, знаниями об агентах, с которыми возможно общение, а также правилами, определяющими его поведение в плане выполнения своих обязанностей [7].

Рисунок 1 – Взаимосвязь моделей при агентно-ориентированном проектировании процесса обучения

Из характера образовательного процесса следует, что кроме реактивности, автономности, активности и коммуникабельности, архитектура программного агента должна иметь внутренние механизмы мотивации, которые задаются ментальными свойствами, такими как убеждения, обязательства, способности и правила поведения. Из существующей классификации для создаваемой системы больше подходит архитектура, основанная на классических принципах искусственного интеллекта, т. е. архитектура интеллектуального агента на основе продукционных правил.

Благодаря разработанным агентно-ориентированным моделям осуществлен системный переход от этапа постановки задачи к этапу программной реализации компьютерной среды с элементами общения между субъектами учебного процесса изучения дисциплин кафедры [7].

Описание визуальных моделей инструментальной среды Jack

В результате анализа существующих инструментов была выбрана среда JACK, которая использует язык Java для того, чтобы внедрить концепции агентно-ориентированного программирования. JACK - это профессиональная, кроссплатформенная среда для создания, эксплуатации и интеграции коммерческих агропромышленных систем. Она построена на прочной логической основе: модели убеждений, желаний и намерений (belief, desire, and intention (BDI) model). BDI - это интуитивная и мощная абстракция, которая позволяет разработчикам управлять сложностью проблемы. В JACK агенты определяются с точки зрения их убеждений (что они знают и что они умеют делать), их желания (какие цели они хотят достичь) и их намерения (цели, которые они в настоящее время преследуют к достижению) [8].

BDI-архитектура позволяет моделировать ментальные свойства агентов, необходимых для решения задачи обучения. Агент, имеющий BDI-архитектуру, описывается тремя компонентами A = (B, D, I), где B – это убеждения агента, которые являются информацией агента о собственном состоянии и состоянии его окружения,

и рассматриваются как его информационная компонента; D – это желания агента в виде информации о его целях, которые рассматриваются как его мотивационная компонента; I – это намерения агента, которые представляют возможные направления его действий, и являются его рассудительной компонентой.

Для программной реализации убеждений, желаний и намерений агента в языке JACK предусмотрены следующие новые конструкции, расширяющие синтаксис языка Java на уровне классов:

- Agent (Агент), определяет интеллектуальных агентов;
- Event (Событие), определяет цели агента, в виде событий, для моделирования ситуаций и сообщений, на которые агент должен быть способен ответить;
- Plan (План), описывает намерения агента в отношении достижения цели в виде планов и условий их применимости;
- Beliefset (Множество убеждений), описывает знания агента;
- Capability (Способность), структурирует убеждения, события и планы в кластеры для реализации определенной интеллектуальной способности агента для достижения цели.

Чтобы работать в инструментальной среде JACK, необходимо мыслить на уровне её понятий, характерных для BDI-архитектуры. Для правильного перехода от абстрактных моделей агентов к их представлению на уровне визуальных моделей среды JACK, авторами были составлены спецификации семантики визуальных моделей данной среды [8].

Семантика визуальной модели агента, представленного формально в среде JACK в виде Agent = (N, Bel, PE, HE, SE, PS), описывается следующим образом: N – имя агента; Bel – убеждения агента; PE = {E1, E2, …, En} – множество имен событий, создаваемых собственными методами агента; HE = {{E1, E2, …, En}, HE1, …} – множество обрабатываемых событий, создаваемых самостоятельно и воспринимаемых извне; SE = {SE1, SE2, …, SEm} – множество имен событий, передаваемых агентом во внешнюю среду; PS = {P1, P2, …, Pk} – множество имен планов, определяющих поведение агента.

Более эффективно определять агента можно через его способности, тогда Agent = (N, Bel, PE, Cap), где Cap = {C1, C2, …, Cn} – множество имен способностей, которыми обладает агент для достижения поставленных целей.

Семантика визуальной модели воспринимаемого события, формально представляемая в среде JACK в виде Event = (N, Pw, Pa, EvT), имеет следующее внутреннее содержание: N – имя воспринимаемого события; Pw – метод автоматического порождения новых убеждений при соответствии собственных убеждений агента некоторым условиям; Pa – метод явного восприятия агентом события внешнего мира и формирования новых убеждений, связанных с данным событием, в результате которого агент получает цель для достижения; EvT – определение стратегии достижения цели [8].

Визуальная модель плана для описания рассуждений агента формально в среде JACK представляется в виде Plan = (N, Ev, Sev, MP, MF, Rel, C, B), где N – имя плана; Ev – имя события, для обработки которого предназначен данный план; Sev – множество имен событий, отправляемых агентом во внешнюю среду при выполнении плана; MP, MF – завершающие действия, выполняемые соответственно при успешном /неуспешном выполнении плана; Rel – метод определения применимости плана для обработки события на основе истинности логического условия, зависящего от результата сопоставления убеждений агента о данном событии; C – метод определения применимости плана для обработки события на основе истинности логического условия, зависящего от результата сопоставления собственных убеждений агента, при условии, что выполнено логическое условие метода Rel; B – основной метод рассуждений агента, выполняемый в случае применимости плана [8].

Выделенные наиболее важные аспекты семантики визуальных моделей среды JACK гарантируют качественное их построение.

Связь логических моделей Gaia с визуальными моделями среды Jack

Процесс создания визуальных моделей агента и его ментальных составляющих является основополагающим, так как именно на этом этапе разработки агентной системы происходит переход на следующий, более низкий, уровень абстракции. Возникает задача перехода от логических моделей к визуальным моделям среды JACK для представления агента в виде событий, планов, убеждений [7]. Для этого были разработаны рекомендации для отображения логических моделей Gaia в концепты среды JACK [6].

При создании модели агента в JACK все агентные типы для МАС берутся из модели агентов Gaia (см. рис. 2). Для каждого агентного типа с помощью базового графического примитива визуально создается агент со своим именем. При создании в JACK модели агента определяются множества воспринимаемых и передаваемых событий HE, PE, SE. Определение событий является ключевым вопросом при создании агента, потому что его деятельность зависит от их возникновения (если события не происходят – агент бездействует). Если возникает событие, то у агента, во-первых, появляется желание (воспринимаемое как цель) обработать это событие и, во-вторых, появляются новые убеждения об этом событии, которые играют определяющую роль при выборе агентом намерений в отношении его ответных действий [7].

Все желания, которые может иметь агент, в агентном языке JACK определяются множеством воспринимаемых событий для обработки HE, в котором выделяется два непересекающихся подмножества

событий: множество PE и множество событий, воспринимаемых из внешнего мира (элементы множества HE, не принадлежащие множеству PE). Исходя из того, что все проявления деятельности агента определены активностями протоколами в модели ролей, то предшествующие им события (множество PE) естественно определяются в соответствии с ними (рис. 2).

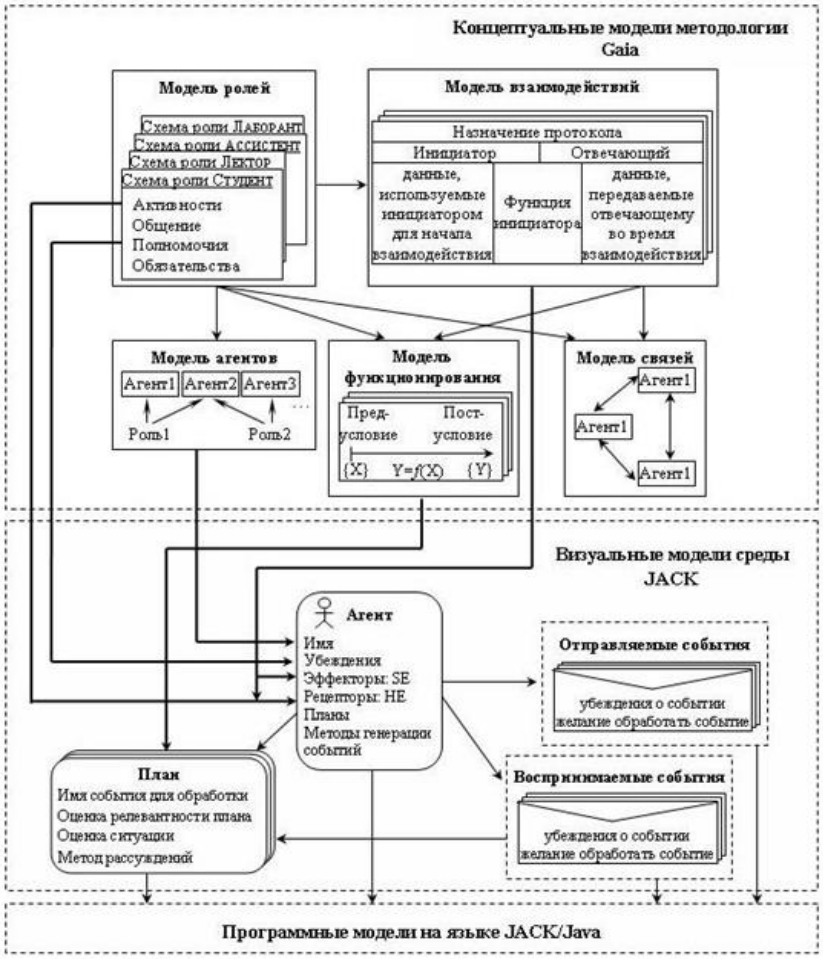


Рисунок 2 – Связь абстрактных моделей методологии Gaia с агентными моделями языка JACK

Множество передаваемых во внешнюю среду событий SE определяется в соответствии с протоколами, в которых данный агент является инициатором взаимодействия. Таким образом определяются воспринимаемые и передаваемые события для агента или, другими словами, его рецепторы и эффекторы.

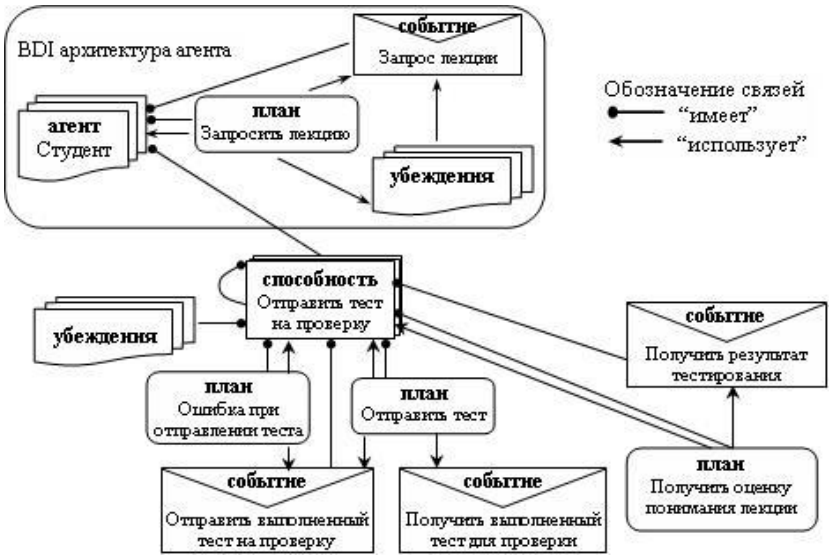


Рисунок 3 – Архитектура программного агента Студент в системе JACK

При определении множества планов действий агента для обработки каждого воспринимаемого события в зависимости от сложившейся ситуации следует исходить из того, что весь функциональный аспект агента зафиксирован в модели функционирования уровня Gaia для данного агентного типа. В этой связи, при создании конкретных планов обработки различных ситуаций, возможных для события, следует использовать его функции, описанные в модели функционирования каждого агентного типа (рис. 2).

По разработанным в среде JACK визуальным моделям агентов сгенерированы физические компоненты агентов на Java, которые представлены в виде классов с определенными отношениями между ними (рис. 3).

Заключение

Выполнен агентно-ориентированный анализ процесса обучения студентов на кафедральном уровне, благодаря которому получена новая модель индивидуального обучения студентов по конкретной дисциплине. Особенностью модели является сохранение взаимоотношений между участниками учебного процесса, близких к реально существующим, и предоставление возможности автономного и распределенного выполнения учебно-методических обязанностей. Эта модель позволяет повысить децентрализованность и индивидуальность работы преподавателей и студентов на уровне кафедры.

Литература

1. Тарасов В.Б. От многоагентных систем к интеллектуальным организациям: философия, психология, информатика.- М.: "Едиториал УРСС", 2002. -352 с.

2. Гаврилова Т. А., Яшин А. М., Фертман В. П. Взаимодействие интеллектуальных агентов для поддержки сервера дистанционного обучения// Материалы междунар. конф. "Интеллектуальные системы и информационные технологии в управлении IS&ITC", Псков, 2000, с.224-227.

3. Курейчик В.М. Эволюционная адаптация интерактивных средств открытого образования /В.М. Курейчик, Л.А. Зинченко //Открытое образование. - 2001. - N1. - С.43-50.

4. Глибовец Н.Н. Использование JADE (Java Agent Development Environment) для разработки компьютерных систем поддержки дистанционного обучения агентного типа// Educational Technology&Society 8(3) 2005 ISSN 1436-4522 pp. 325-345.

5. Шляпина С.Ф., Дубицкая С.А. Применение электронных образовательных ресурсов в учебном процессе // Электронное обучение в непрерывном образовании. Сб. трудов V Междунар. науч.-практич. конференции. – Ульяновск : УлГТУ, 2018. - С. 744-748.

6. Ваныкина Г.В., Сундукова Т.О. Педагогические условия эффективного использования виртуальной образовательной среды в обучении // Электронное обучение в непрерывном образовании. Сб. трудов V Междунар. науч.-практич. конференции. – Ульяновск: УлГТУ, 2018. – С. 143-150.

7. Федяев О.И. Проектирование виртуальной кафедры университета на основе многомодельного агентно-ориентированного подхода /О.И.Федяев, Т.Е.Жабская // Искусственный интеллект – 2010, №3. – С. 679–686.

8. David Kinny, Michael Georgeff, Anand Rao A Methodology and Modelling Technique for Systems of BDI Agents. Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent MAAMAW'96, (LNAI Volume 1038).

Московченко А.В., Жабская Т.Е., Федяев О.И. Представление виртуальной кафедры на уровне визуальных моделей многоагентной среды Jack. В статье рассмотрена учебная кафедра университета как объект виртуальной образовательной среды, с помощью методологии Gaia выполнен ее агентно-ориентированный анализ. Для программной реализации агентов с BDI-архитектурой выбрана инструментальная среда Jack. Рассмотрена трансформация логических моделей Gaia в визуальные модели среды Jack.

Ключевые слова: *агентно-ориентированное проектирование, виртуальная кафедра, методология Gaia, инструментальная среда Jack, BDI-архитектура.*

Anastasia Moskochenko, Tatyana Zhabskaya, Oleg Fedyaev. Presentation of the virtual department at the level of visual models of the multi-agent environment Jack*. The article considers the educational department of the university as an object of the virtual educational environment, using the Gaia methodology, its agent-based analysis was performed. For the software implementation of agents with the BDI architecture, the Jack tool environment was chosen. The transformation of Gaia logical models into visual models of the Jack environment is considered.*

Key words: *agent-oriented design, virtual department, Gaia methodology, Jack design tool, BDI architecture*

УДК 004.424

Система удаленного резервного копирования данных для корпоративных сетей

Ольшевский А.И., Нестеренко В.С.
Донецкий национальный технический университет, г. Донецк
кафедра искусственного интеллекта и системного анализа
a_olshevskiy@mail.ru, nes.vadim.s@gmail.com

Ольшевский А.И., Нестеренко В.С. Проектирование программного комплекса удаленного резервного копирования данных для корпоративных сетей. *Рассмотрены общие средства универсального программирования, был выбран объектно-ориентированный язык программирования. Определена структура программного комплекса, разделены задачи и функции клиентской и серверной части. Предложена модель распределенного «разбитого» хранения файла. Реализован открытый максимально понятный исходный программный код.*

Ключевые слова: *кроссплатформенность, резервное копирование данные, модель распределенного хранимого файла, корпоративные сети, открытый исходный код.*

Постановка проблемы

На сегодняшний день актуальной является проблема сохранности важных пользователю данных. Данные могут быть утеряны в любой момент в связи с множеством различных факторов, приводящих к повреждению данных. Наиболее распространенные факторы – отказ работы техники или человеческий фактор случайности. Введу такой острой проблемы потери данных, было найдено решение – создание резервной копии важных данных.

Проблема резервного копирования данных является наиболее критичной для большинства предприятий на сегодняшний день. На каждом предприятии своя структура хранения данных, будь то базы данных или обычные файлы, свои сервера, ЭВМ, и свои используемые операционные системы, потому для каждого предприятия подходит только им необходимое специфическое программное средство для резервного копирования данных, соответствующие необходимым требованиям структуры самого предприятия.

В связи с отсутствием универсального программного средства резервного копирования данных, подходящего под большинство требований любого предприятия, проблема создания такой системы резервного копирования, с открытым исходным кодом на сегодняшний день весьма актуальна [1].

Цель работы – проектирование и разработка универсальной архитектуры приложения, используя современные подходы и средства программирования для возможности внедрения приложения в корпоративную сеть большинства предприятий.

Программная реализация

Кроссплатформенность приложений является стандартом в современных реалиях общества. Множество предприятий и пользователей используют предпочтительную им операционную систему, где не всегда есть возможность использовать приложение, разработанное под другую операционную систему, а запуская, используя не предусмотренным разработчиками способ, есть вероятность некорректного функционирования отдельных возможностей приложения [2].

В связи с такой острой необходимостью, стал вопрос разработки универсального программного средства, для которого был выбран объектно-ориентированный язык программирования Java. Все приложения, разработанные на языке Java транслируются в специальный байт-код, потому они и могут работать на любой компьютерной архитектуре, с использованием виртуальной Java-машины.

Приложение представляет собой программный комплекс, разделённый на две части, клиентская часть, устанавливаемая на необходимые ЭВМ, принимает информацию и задачи с серверной части, сжимает, по необходимости, файлы и отправляет в определённое пользователем место.

Клиентское приложение разработано на языке программирования Java, общается с сервером по средству GET запросов, и принимая задачи с сервера, по расписанию отправляет необходимые файлы в определённое пользователем место.

Серверная же часть, доступная по URL внутри сети, помимо всей логики отвечает и за Web-интерфейс, позволяющий управлять и настраивать все задачи по резервному копированию, определять места хранения,

расписание для каждого компьютера, определять роли и ограничения пользователей, имеющих доступ к программному обеспечению, подключенных к серверу, где установлена клиентская часть. Web интерфейс представляет собой приложение, разработанное с использованием языка программирования JavaScript, JavaScript-фреймворка AngularJS, систему Web-шаблонов Thymeleaf, HTML препроцессор Pug/Jade, программную платформу Node.js, и СУБД MongoDB [3]. Данные современные технологии позволяют разработчикам более быстро и удобно реализовать сложный функционал Web-приложения и общение с базой данных.

AngularJS позволяет расширить MVC шаблон. Model-View-Controller – схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер – таким образом, что модификация каждого компонента может осуществляться независимо [4].

Thymeleaf. Шаблонные файлы Thymeleaf на самом деле является обычными текстовым файл, имеющий формат XML/XHTML/HTML5. Thymeleaf Engine (машина Thymeleaf) читает шаблонный файл и комбинирует с объектами Java, чтобы генерировать другой документ. Данная система представляет собой «обещание», интегрируя объекты Java в Web-интерфейс [5].

HTML препроцессор Pug/Jade даёт возможность написания разметки с рядом преимуществ по сравнению с обычным HTML [6].

Node.Js – программная платформа превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (написанный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера [7].

MongoDB – документоориентированная система управления базами данных (СУБД) с открытым исходным кодом, не требующая описания схемы таблиц. Классифицирована как NoSQL, использует JSON-подобные документы и схему базы данных. Написана на языке C++ [8].

Распределенное хранение и считывание данных

Реализация распределенной файловой системы крайне сложна, однако видя её особенности работы и возможности, был предложен набор механизмов работы, и добавления их в систему резервного копирования данных, а именно – считывание одних и тех-же данных с разных мест хранения, что увеличивает скорость считывания, и не влияет на пропускную способность подключения к сети. Для более безопасного хранения данных и защиты от кражи, была взята идея разбиения файла на части, хранение этих частей на разных носителях информации (в нескольких копиях) и генерация map-файла, для последующей «сборки» файла из частей (см.рис.1). Данная особенность не только увеличивает скорость считывания данных, но и обеспечивает большую надежность хранения данных [9].

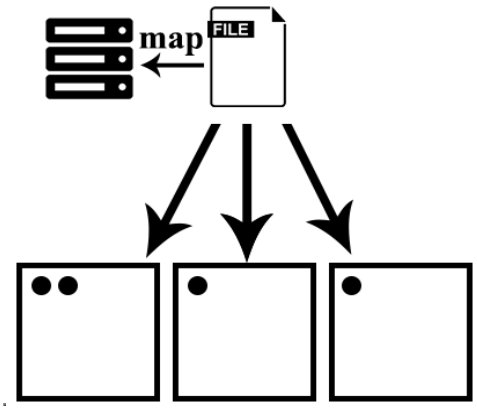


Рисунок 1 – Схематический вид распределенного «разбитого» хранения файла

Ограничение ширины канала сети. На сегодняшний день большинство данных и баз данных различных компаний и корпораций располагаются на собственных хранилищах под управлением одной из операционной системы вида Unix, что и обеспечивает меньшую нагрузку на оборудование и простоту выполнения резервного копирования по средству запуска скриптов по расписанию [10]. Данный метод резервного копирования предполагает наличие обученного специалиста для работы, накладывая на него задачу слежения за резервным копированием. Специалисту необходимо реагировать на различные сбои, помнить расположение и число резервных копий, а также отвечать за их целостность.

Стоит отметить остро стоящую проблему пропускной способности подключения в сети и сети-интернет. Обычно загрузка и выгрузка файлов занимают всю доступную ширину канала, ограничивая подключение других пользователей сети. Помимо разбиения файлов на части при их хранении, и хранении частей на различных носителях информации, существует возможность дальнейшей реализации BitTorrent протокола в разрабатываемом сервисе резервного копирования. Данный протокол используется для передачи файлов частями, каждый torrent-клиент, получая (скачивая) эти части, в то же время отдаёт (закачивает) их другим клиентам, что снижает нагрузку и зависимость от каждого клиента-источника и обеспечивает избыточность данных [11]. Реализация этого протокола востребована в программном продукте из-за таких особенностей как: отсутствие очередей на скачивание, закачивание файлов небольшими фрагментами (чем менее доступен фрагмент, тем чаще он будет передаваться), клиенты обмениваются сегментами непосредственно между собой, по принципу «ты – мне, я – тебе», скачанные фрагменты становятся немедленно доступны другим клиентам, контролируется целостность каждого фрагмента, на фрагменты разбиваются не отдельные файлы, а вся раздача целиком, поэтому у пользователя, пожелавшего скачать лишь некоторые файлы из раздачи, для поддержания целостности фрагментов нередко храниться также небольшой объём избыточной (для него) информации. В качестве объекта раздачи могут выступать несколько файлов (например, содержимое каталога).

Среди многих программных средств стоит выделить Resilio Sync, – платный сервис для синхронизации файлов и резервного копирования по протоколу BitTorrent между произвольными устройствами. Поддерживается компанией Resilio, Inc., выделенной из BitTorrent, Inc.. BitTorrent Sync синхронизирует файлы, используя самоорганизующуюся одноранговую сеть (P2P), основанную на протоколе BitTorrent. Этот протокол зарекомендовал себя эффективной передачей больших файлов между множеством устройств. В отличие от облачных сервисов с аналогичной функциональностью, при использовании для этих целей BitTorrent Sync пользовательские данные находятся на локальном носителе и требуют, как минимум, одного устройства, подключенного к сети для доступа к ним [12].

Интеграция с популярными СУБД. У всех существующие СУБД на сегодняшний день имеется защита от доступа к БД, пока СУБД работает, потому функционал, позволяющий работать с СУБД, является важным в программном обеспечении для резервного копирования.

Хранение данных на популярных облачных сервисах-хранилищах. На сегодняшний день существует множество облачных сервисов, позволяющих хранить данные на собственных носителях информации. В разрабатываемом программном продукте разработана возможность распределенного хранения данных на популярных облачных хранилищах (Dropbox, Google Drive, Mega, OneDrive и пр.). В программном продукте всё дисковое пространство с выбранных пользователем сервисов представлен, как единый. Так же спроектированы элементы искусственного интеллекта для отслеживания загружаемых и считываемых данных, и автоматического распределения наиболее используемых файлов на сервис с лучшей скоростью обмена данными, а менее используемые файлы, соответственно, на сервис худшей скоростью обмена данными [13].

Так же стоит отметить ftp хранилища на локальных и выделенных серверах. Данный протокол так же позволяет реализовать у себя BitTorrent протокол, получая данные с определенной позиции файла.

Хранение изменений данных. Уже существует возможность дописывать в конец несжатых файлов добавленные пользователем данные, вычисляя изменения, ещё никто не реализовал механизм хранения разницы данных несжатых файлов. Идея заключается в отслеживании изменения файла, а именно отслеживания добавленных и удалённых данных. Всегда храниться первый эталонный файл, а последующие хранят лишь изменённые данные и места где эти изменения нужно применить. Так «накладывая» на эталонный файл все изменения и получается нужный файл, при этом сами изменения будут занимать намного меньше места на дисковом пространстве чем хранение всех версий несжатого файла пользователя.

Помимо такой технологии, стоит отметить синхронизацию данных, а именно отслеживание изменений выбранных пользователем каталогов, и автоматической загрузки/скачивания изменённых файлов, что позволит пользователю не отвлекаться на ручное управление распределения файлов.

Выводы

Проведенные исследования позволили выяснить, что для реализации системы необходимо использовать кроссплатформенность. Определена структура программного комплекса, разделены задачи и функции клиентской и серверной части. Предложена модель распределенного «разбитого» хранения файла. Установлено, что для отслеживания загружаемых и считываемых данных, и автоматического распределения наиболее используемых файлов на сервис с лучшей скоростью обмена данными, необходима подсистема анализа с элементами искусственного интеллекта.

Литература

1. Ольшевский А.И. Исследование и проектирование программного комплекса удаленного резервного копирования данных [Текст] / А.И. Ольшевский, В.С. Нестеренко // Сб. мат. «Информатика, управляющие

системы, математическое и компьютерное моделирование» (ИУСМКМ – 2018). – Донецк : ДонНТУ, 2017. – С. 61-64

2. Бычкова Е.В. Программное средство создания резервных копий данных [Текст] / Е.В. Бычкова, В.С. Нестеренко // Сб. науч. тр. «Информатика, управляющие системы, математическое и компьютерное моделирование» в рамках III форума «Инновационные перспективы Донбасса» (ИУСМКМ – 2017). – Донецк : ДонНТУ, 2017. – С. 381-384

3. Гома Х. UML Проектирование систем реального времени, параллельных и распределенных приложений / Х. Гома // Пер. с англ. – М.: ДМК Пресс, 2011. – 704 с.

4. Angular.js [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://angularjs.org. – Загл. с экрана.

5. Thymeleaf [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://thymeleaf.org – Загл. с экрана.

6. Pug/Jade [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://pugjs.org – Загл. с экрана.

7. Node.js [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://ru.wikipedia.org/wiki/Node.js. – Загл. с экрана.

8. MongoDB [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://www.mongodb.com. – Загл. с экрана.

9. Jepsen T.C. Distributed Storage Networks: Architecture, Protocols and Management / Jepsen T.C. – Raleigh : John Wiley & Sons, 2003. – 338 p.

10. UNIX and Linux System Administration Handbook / [Nemeth E., Snyder G., Hein T.R., Whaley B., Mackin D.] – [4th edition] – Upper Saddle River : Addison-Wesley Professional, 2017. – 1500 p.

11. BitTorrent (протокол) [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://ru.wikipedia.org/wiki/BitTorrent_(протокол). – Загл. с экрана.

12. Resilio Sync [Electronic resourse] / Интернет-ресурс. – Режим доступа : www/ URL: https://www.resilio.com/. – Загл. с экрана.

13. Gupta M. Storage Area Network Fundamentals / Gupta M. – Indianapolis : Cisco Press, 2002. – 320 p.

Ольшевский А.И., Нестеренко В.С. Проектирование программного комплекса удаленного резервного копирования данных для корпоративных сетей. *Рассмотрены общие средства универсального программирования, был выбран объектно-ориентированный язык программирования. Определена структура программного комплекса, разделены задачи и функции клиентской и серверной части. Предложена модель распределенного «разбитого» хранения файла. Реализован открытый максимально понятный исходный программный код.*

Ключевые слова*: кроссплатформенность, резервное копирование данные, модель распределенного хранимого файла, корпоративные сети, открытый исходный код.*

Olshevsky A.I., Nesterenko V.S. Designing software for remote data backup for corporate networks. *Considered general means of universal programming, was chosen object-oriented programming language. The structure of the software complex is defined, the tasks and functions of the client and server parts are divided. A model of distributed "broken" file storage is proposed. Implemented an open, highly understandable source code.*

Key words: *cross-platform, data backup, distributed stored file model, corporate networks, open source.*

УДК 003.26.09

Особенности реализации шифра хилла в компьютерных приложениях

Поздняков А.А.
Донецкий национальный университет
mail.0awawa0@gmail.com

Поздняков А.А. Особенности реализации шифра Хилла в компьютерных приложениях. *Рассматриваются все этапы программной реализации шифра Хилла на языке программирования C# с использованием Windows Forms. Описываются тонкости разработки алгоритма программы, а также некоторые трудности реализации с учётом особенностей языка C#.*

Ключевые слова: *шифр Хилла, C#, особенности языка, программная реализация, Windows Forms, криптография*

Введение

Шифр Хилла – полиграммный шифр подстановки, основанный на линейной алгебре и модульной арифметике. Этот шифр занял почетное место в истории криптографии как первый шифр, позволяющий оперировать более чем тремя символами одновременно. Кроме того, Хилл «впервые перевёл криптографию с использованием полиграмм в разряд практических дисциплин» [1]. Шифр Хилла рассматривается практически во всех учебных курсах по криптографии. Студенты изучают принцип шифрования, шифруют небольшие блоки текста вручную, а также создают программные реализации шифра Хилла. И если с теорией и шифрованием вручную проблем не возникает, то в программной реализации могут быть подводные камни, ставящие студентов, а иногда и преподавателей в тупик.

Целью данной работы является создание программной реализации шифра Хилла, описание возможных проблем, возникающих в процессе, а также предложение их решения.

Описание алгоритма шифра Хилла

Алгоритм шифрования проще разбирать на примере. Пускай необходимо зашифровать одно слово: «сегодня». В первую очередь выбирается алфавит. Пусть это будет русский алфавит (длина – 33), дополненный несколькими символами до длины в 37 символов. Каждому символу алфавита ставится в соответствие какой-то номер в пределах от 0 до 36 [2]. Далее сообщение кодируется из букв в цифры, принимая вид: «13 5 4 2 5 4 29». Ключ задаётся в виде матрицы n x n, где n – размер блока. Часто ключ задают словом длиной в n^2, которое затем кодируется и записывается в виде матрицы. Но иногда сразу указывают матрицу, особенно если n > 3 и слово подобрать проблематично. Пусть n = 3, а слово-ключ будет: «революция». В закодированном виде: «17 5 2 15 12 31 23 9 32». А в матричном виде:

$$K = \begin{pmatrix} 17 & 5 & 2 \\ 15 & 12 & 31 \\ 23 & 9 & 32 \end{pmatrix}$$

Чтобы шифрование было обратимым, необходимо чтобы определитель матрицы ключа был не равен нулю (det K ≠ 0), то есть матрица должна быть невырожденной, а также наибольший общий делитель детерминанта матрицы ключа и длины алфавита должен равняться единице (НОД(det K, 37) = 1). Выбранная матрица соответствует данным требованиям иначе пришлось бы выбирать другую матрицу:

$$\det K = 2668$$

Следующим шагом алгоритма является разделение текста на блоки по n=3 символов в каждом, которые являются матрицами размера 1 x n, также можно формировать матрицы n x 1, любой из вариантов сработает. Слово «сегодня» состоит из 7 символов и при разбиении получается 2 полных блока по 3 символа и один блок в 1 символ. Последний блок должен быть дополнен до полного любыми символами, не изменяющими семантику сообщения, пусть это будут пробелы:

$$P_1 = (18 \quad 5 \quad 3)$$
$$P_2 = (15 \quad 4 \quad 14)$$
$$P_3 = (32 \quad 35 \quad 35)$$

Далее для получения шифротекста каждая матрица открытого текста умножается на матрицу ключа по модулю 37:

$$C_1 = P_1 * K \ (mod\ 37) = (4 \quad 16 \quad 0)$$
$$C_2 = P_2 * K \ (mod\ 37) = (7 \quad 4 \quad 15)$$
$$C_3 = P_3 * K \ (mod\ 37) = (12 \quad 24 \quad 25)$$

Теперь если декодировать шифротекст с помощью алфавита, получится: «дпаждолчш». Для расшифровки текста необходимо умножить каждый блок шифротекста на обратную матрицу ключа по модулю 37:

$$K^{-1} = \begin{pmatrix} 17 & 20 & 5 \\ 12 & 32 & 33 \\ 11 & 9 & 23 \end{pmatrix}$$

$$P_1 = C_1 * K^{-1} = (18 \quad 5 \quad 3)$$

$$P_2 = C_2 * K^{-1} = (15 \quad 4 \quad 14)$$

$$P_3 = C_3 * K^{-1} = (32 \quad 35 \quad 35)$$

После декодирования получится: «сегодня». Два лишних пробела можно позже срезать.

Реализация

Для программной реализации шифра Хилла был выбран язык C# с графическим интерфейсом пользователя Windows Forms. В первую очередь был разработан интерфейс программы. Кроме форм для ввода и вывода ключа, открытого текста и закрытого текста, также были добавлены формы для вывода результатов промежуточных вычислений с целью явно продемонстрировать работу алгоритма. Интерфейс программы изображен на рисунке 1. Назначение каждого из полей указано в таблице 1.

Далее разрабатывается логика программы. Так как шифрования предполагает работу с матрицами, необходимо реализовать матрицы в виде класса. Базовой структурой данных для этого разумнее всего выбрать двумерный массив, обеспечивающий константный доступ к элементам матрицы по индексу [4].

В большинстве современных языков программирования уже есть реализация всех необходимых методов работы с матрицами и можно было бы использовать их, но в учебных целях лучше всё реализовывать самостоятельно. Стоит отметить, что так как все вычисления производятся в кольце вычетов по модулю 37, нужно уделить делению по модулю внимание. С учетом того, что значения элементов матрицы изменяются только внутри методов класса, разумно будет делить по модулю только внутри этих методов, чтобы избежать путаницы.

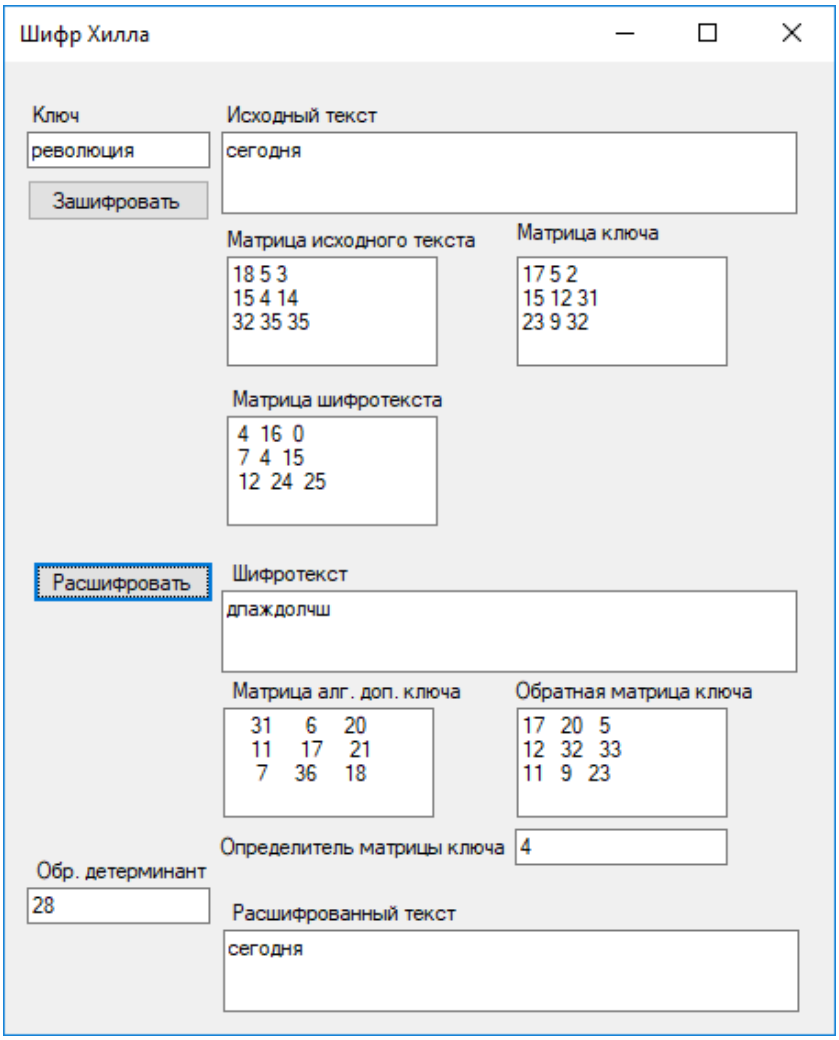


Рисунок 1 - Интерфейс программы

Таблица 1 - Назначения полей ввода/вывода

Название поля	Назначение
Ключ	Ввод слова-ключа шифрования в символьном виде.
Исходный текст	Ввод открытого текста в символьном виде.
Матрица исходного текста	Вывод открытого текста в виде матрицы чисел. Матрица содержит n столбцов. Количество строк будет зависеть от длины открытого текста.
Матрица ключа	Вывод ключа в виде матрицы чисел размера n x n.
Матрица шифротекста	Вывод закрытого текста в виде матрицы чисел.
Шифротекст	Вывод закрытого текста в символьном виде. Это же поле может использоваться для ввода закрытого текста в символьном виде для расшифровки.
Матрица. алг. доп. ключа	Вывод матрицы алгебраических дополнений ключа. Матрица алгебраических дополнений является промежуточным результатом вычисления обратной матрицы ключа. Все числа находятся в диапазоне 0..36, т.к. вычисления производятся в кольце модулей 37.
Обратная матрица ключа	Вывод обратной матрицы ключа в числовом виде.
Определитель матрицы ключа	Вывод определителя матрицы ключа. Число входит в кольцо вычетов по модулю 37.
Обр. детерминант	Вывод обратного числа в кольце вычетов по модулю 37 для определителя матрицы ключа.
Расшифрованный текст	Вывод открытого текста после расшифровки в символьном виде. Лишние пробелы в конце удаляются.

Кроме того, внимания заслуживает нахождение значения обратного к определителю ключа элемента в кольце вычетов по модулю 37. Вспомним определение: обратным к числу a по модулю m называется такое число b, что:

$$a * b = 1\ (mod\ m)$$

С учётом этого условия наивным способ нахождения такого элемента является цикл с предусловием:

```
int b = 0;
while ((a * b) % m != 1)
  b++;
```

Такое решение, конечно, сработает, но не при больших значениях m, так как скорость работы оценивается как O(m). Вместо этого следует использовать расширенный алгоритм Евклида, скорость работы которого оценивается как O(log(min(a, b))) [5].

Также во время реализации могут возникнуть проблемы с делением по модулю. Дело в том, что возвращаемое значение оператора % имеет такой же знак, как и у левого операнда. Отсюда следует, что при делении отрицательного числа по модулю, результатом будет также отрицательное число, что недопустимо в данном алгоритме. Например:

$$-7\ \%\ 37 = -7$$

Необходимо учитывать эту особенность языков C-семейства, иначе в работе программы возникнут ошибки. Такая ситуация может возникнуть в трех шагах алгоритма. Все эти варианты следует предусмотреть и проверять значения на знак:

1) При нахождении обратного элемента в кольце вычетов алгоритм Евклида может возвращать отрицательные числа;

2) При поиске матрицы алгебраических дополнений в результирующей матрице могут появляться отрицательные числа;

3) Определитель матрицы ключа может быть отрицательным числом.

Выводы

В работе представлено детальное теоретическое описание алгоритма шифра Хилла, и особенности его реализации в компьютерных приложениях. Рассмотрены трудности, связанные с особенностями используемого языка программирования, а также с непосредственной реализацией некоторых вычислений. Разработано приложение на языке C#, которое используется в качестве учебного пособия в рамках курса «Криптография».

Список литературы

1. David Kahn The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet. — New York: Scribner, 1996. — 723 p.

2. Hill Ciphers: A Linear Algebra Project with Mathematica [Электронный ресурс] / Murray Eisenberg. — Режим доступа: http://people.math.umass.edu/~murray/HillICTCM12.pdf

3. Алферов, А. П. Основы криптографии / А. П. Алферов, А. Ю. Зубов, А. С. Кузьмин, А. В. Черёмушкин. — 2-е изд. — М.: Гелиос АРВ, 2002. — 480 с.
4. Ахо Альфред В. Структуры данных и алгоритмы / Джон Хопкрофт, Джеффри Д. Ульман — М.: Вильямс, 2000. — 384 с.
5. Ривест, Р. Алгоритмы: построение и анализ / К. Штайн, Ч. Лейзерсон, Т. Кормен. — М.: Вильямс, 2013. — 1328 с.

Поздняков А.А. Особенности реализации шифра Хилла в компьютерных приложениях. *Задача состоит в изучении процесса разработки программы, реализующей шифрование по алгоритму Хилла на языке C# с графическим интерфейсом Windows Forms. В статье рассмотрены все этапы разработки, описаны возможные проблемы и предложены пути их решения.*

Ключевые слова: *шифр Хилла, C#, программная реализация, Windows Forms, криптография, особенности языка.*

Pozdniakov Alexander. Special aspect of the implementation of the Hill cipher in computer applications. *The task is to study the process of developing the application to implement the encryption using Hill algorithm with the C# language and graphic user interface Windows Forms. All stages of developing are viewed in the article. Also described probable errors and paths to solve them.*

Keywords: *Hill cipher, C#, software implementation, Windows Forms, cryptography, language aspects.*

УДК 004.021

Квантование цветового пространства в контексте решения задачи поиска изображений по содержанию

Полетаев В.А., Коломойцева И.А.
Донецкий национальный технический университет
poletaev.vladislav@gmail.com, bolatiger@gmail.com

Полетаев В.А., Коломойцева И.А. Квантование цветового пространства в контексте решения задачи поиска изображений по содержанию. *В данной статье отмечается актуальность задачи поиска изображений по содержанию, описываются существующие подходы и методы решения задачи. В статье проведен экспериментальный анализ использования алгоритма квантования цветового пространства методом k-средних при выделении цветовых и текстурных признаков изображений. Произведена оценка алгоритма в сравнении с существующими методами квантования.*

Ключевые слова*: поиск изображений, графическая БД, квантование, цветовое пространство, метод k-средних*

Введение

Задача поиска изображений, то есть нахождение автоматизированной системой подмножества набора изображений, обладающих некоторыми характеристиками, находит применение во многих прикладных задачах, требующих выполнения запросов к графическим базам данных, классификации изображений, хранящихся в них.

Учитывая увеличивающийся объем графических изображений, представленных, в первую очередь, в веб-пространстве, решение задачи поиска изображений по запросу пользователя является актуальной.

Наиболее широкого распространения получили методы поиска изображений, основанные на использовании ассоциированной с этими изображениями метаинформации, составленной человеком. Методы поиска на основании метаинформации применяются большинством современных систем поиска в сети Интернет.

Однако во многих случаях, например, при автоматическом получении изображений от автоматизированной системы, ручное аннотирование не представляется возможным. Другим случаем невозможности использования метаданных для решения задачи поиска является их низкое качество: составленное пользователем текстовое описание может не в полной мере отражать характеристики изображенных объектов. В этих случаях применяются методы поиска изображений по содержанию (Content-based image retrieval (CBIR)). Эта группа методов использует непосредственно информацию, закодированную в изображении. Методы поиска по содержанию также могут применяться в смежных областях исследования, таких как автоматизированное аннотирование изображений [1] и их классификация.

Целью работы является:

– анализ существующих методов функционирования систем поиска изображений по содержанию (анализ методов извлечения признаков изображений и поиска в пространстве векторов признаков);

– оценка эффективности метода k-средних для квантования цветового пространства в контексте решения задачи поиска изображений по содержимому с целью доказательства того, что этот метод является лучшим на текущей момент и его модификация является перспективным направлением с точки зрения поиска изображений по содержанию.

Существующие методы функционирования CBIR-систем

Разработка систем поиска изображений по содержимому является активно развивающейся областью исследований. Несмотря на это, не было разработано единого алгоритма поиска изображений с минимальным семантическим разрывом между результатами поиска и восприятием различий в изображениях человеком [2].

Большинство разработанных методов работы CBIR-систем включает процесс генерации математического описания изображения, то есть определения множества его признаков, и процесс определения количественной характеристики сходства двух изображений по их математическому описанию. При построении совокупности признаков могут использоваться как локальные и глобальные характеристики обрабатываемого изображения, так и характеристики других изображений.

Методы выделения признаков

Наиболее простым в реализации и распространенным является выделение признаков изображения на основании цветов пикселей [3]. Использование цветов каждого из пикселей в качестве элемента вектора признаков неэффективно: пространство векторов имеет большую размерность, а существующие метрики не будут корректно отражать схожесть исходных изображений. Для снижения размерности применяются гистограммы. Цветовое пространство разбивается на заданное количество областей (происходит его квантование), для каждой из которой определяется процент пикселей изображения, принадлежащих этой области [4].

Квантование цветового пространства может быть фиксировано и осуществлено на этапе разработки метода поиска, или осуществлено алгоритмически, на основании статистики цветов выборки. Сравнительно высокое качество результатов поиска показали методы, использующие квантование, которое принимает во внимание характеристики восприятия цвета человеком, в частности, методы, учитывающие закон Вебера-Фехнера, согласно которому человек воспринимает цвета в виде дискретных значений [2], при этом, если интенсивности цвета x_k и x_{k+1} определяют границы диапазона различимых цветов, выполняется:

$$\frac{x_{k+1} - x_k}{x_k} = \text{const} \; \forall k \tag{1}$$

Кроме вопроса выбора метода квантования встает вопрос выбора самого цветового пространства. Многие исследования показали, что использование пространства цветов RGB является неэффективным в связи с нелинейностью зависимости расстояния между цветами и зрительным восприятием различий. Были разработаны специализированные цветовые пространства, лишенные этого недостатка. Примером такого пространства является CIE L*a*b* (в некоторых источниках — LAB) [3].

Недостатком методов извлечения признаков на основании гистограмм является потеря информации о расположении пикселей или кластеров пикселей заданных цветов.

Другой группой методов, учитывающих локальные зависимости между цветами пикселей являются методы, основанные на текстурных характеристиках изображений. Часто эти методы используются в совокупности с методами на основании цветовых характеристик [5]. Распространен метод на основании локальных бинарных шаблонов [6].

Важной группой методов, извлечения признаков являются методы, использующие информацию о форме объектов, присутствующих на изображении. Эта группа методов выделяет на изображении контуры в виде кривых или ломаных линий. Их характеристики затем используются как элементы вектора признаков.

Методы поиска в пространстве векторов признаков

Одним из возможных методов поиска в пространстве векторов признаков является задание функции, определяющей расстояние между векторами — метрики на пространстве векторов признаков [7]. Кроме аксиом тождества, симметрии и треугольника, необходимых для задания метрики, функция расстояния должна минимизировать семантический разрыв между изображением и его представлением.

При решении задачи поиска и классификации изображений используются традиционные функции расстояния, (например, евклидово расстояние, расстояние городских кварталов), функции расстояния, основанные на статистических характеристиках элементов выборки (корреляционный метод, метрика хи-квадрат, расстояния Бхатачария) и функции расстояния, учитывающие специфику алгоритма формирования векторов [8]. Так, например, если в качестве элементов вектора признаков используются значения элементов гистограммы, для сравнения изображений может быть применена метрика EMD (Earth Mover’s Distance), разработанная для обеспечения инвариантности алгоритма к изменению освещенности [9].

Также используются методы машинного обучения для сравнения изображений по выделенным признакам. Эти методы могут быть использованы для классификации и аннотирования изображений.

Результат анализа существующих подходов

На настоящий момент существует большое количество разнообразных методов поиска изображений по содержанию. Эти методы используют характеристики изображений: цвет, текстуру, форму объектов и определяют функцию схожести изображений по численным значениям этих характеристик.

Поскольку задача поиска изображений должна учитывать особенности восприятия содержимого и оценки схожести изображений человеком, сложна оценка эффективности существующих методов поиска и классификации. Для оценки результатов методов применяются специализированные аннотированные тестовые примеры данных, которые могут не полностью соответствовать предметной области для которой разрабатывается CBIR-система; производится искусственная модификация изображений для тестирования

инвариантности алгоритма к перемещению, повороту и другим искажениям, которые могут не соответствовать реальным данным [10]; либо проводится сравнение результатов экспериментов и восприятия путем опроса [3].

Анализ метода k-средних в контексте решения задачи поиска изображений

Числовые характеристики графической информации зависят от источника происхождения этой информации и области ее применения. К этим характеристикам относятся: плотность распределения цветов в цветовом пространстве, преобладающие конфигурации текстуры компонентов изображений, часто встречающиеся шаблоны контуров изображенных объектов. При проектировании алгоритмов и методов поиска, эти закономерности могут использоваться для лучшей адаптации алгоритма к работе в заданной предметной области.

На рисунке 1 показано распределение цветов двух групп изображений в цветовых пространствах RGB, визуализированного в виде куба и HSV, визуализированного в виде конуса (совокупность оттенка и насыщенности была также спроектирована на горизонтальную плоскость для большей наглядности). Распределение было построено на основании оцифрованных копий изображений картин в жанре “пейзаж”, а также фотографий архитектурных объектов. Для построения были использованы изображения открытой базы данных “Web Gallery of Art” [12]; было извлечено 5000 точек цветового пространства из 100 изображений этих категорий. Графические файлы, цвета которых являются градациями серого не были включены в выборку.

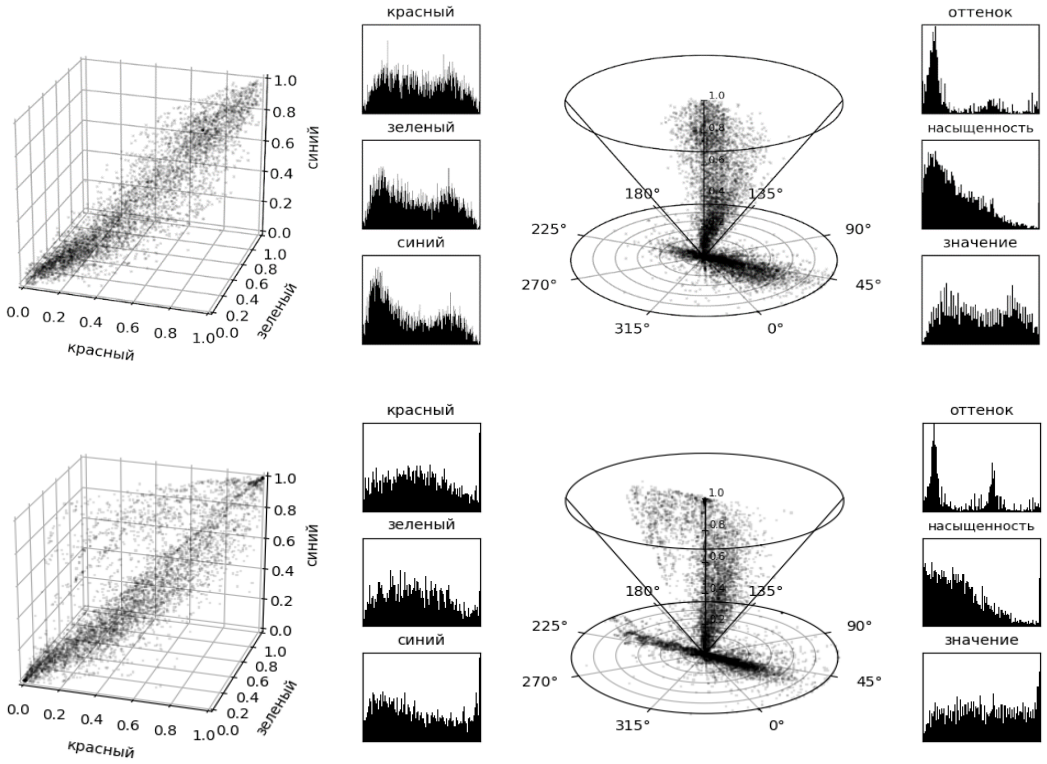


Рисунок 1 – Диаграммы рассеяния и гистограммы цветов, построенные по изображениям картин в жанре “пейзаж” (сверху) и архитектурных объектов (снизу)

По приведенным на рисунке диаграммам рассеяния и гистограммам можно сделать вывод о существовании значительных отличий между плотностями распределения цветов групп изображений. Исходя из этого возникает вопрос выбора метода квантования цветового пространства, оптимизированного для определенной предметной области.

Выбор метода квантования цветового пространства, подразумевает определение такой функции q, которая обращает множество точек цветового пространства U на некоторое конечное множество C — подмножество цветового пространства, заданной длины k:

$$U \xrightarrow{q} C \subset U;\ |C| = k \tag{2}$$

При квантовании возникает ошибка, которая зависит от определенной на цветовом пространстве метрики d и плотности распределения p цветов точек изображения [11]:

$$E(d,q) = \int_C d(x, q(x)) \cdot p(x) dx \tag{3}$$

Для минимизации ошибки предлагается алгоритм, основанный на разбиении цветового пространства на k подмножеств методом k-средних, входными данными которого является совокупность цветов пикселей всех изображений набора или значительно большой выборки. Метод k-средних обеспечивает минимизацию квадратов отклонений точек кластера от центров этого кластера, а следовательно – и минимизацию ошибки (3).

Алгоритм k-средних является одним из классических подходов к минимизации палитры изображения и применяется во многих прикладных программных системах обработки растровой графики. Его применение на большом диапазоне изображений должно позволить осуществить адаптацию функции квантования для некоторой предметной области графической информации.

Использование кластеризации в контексте осуществления поиска изображений по содержанию должно обеспечить выделение в изображениях набора как часто используемых цветов, там и сравнительно редко встречающиеся сочетаний (при достаточно большом значении k).

Псевдокод предлагаемого алгоритма квантования приведен на рисунке 2.

Вход: d — метрика, заданная на цветовом пространстве
N — количество изображений в выборке
M — размер выборки пикселей изображений
Выход: области пространства, соответствующие им цвета
1: $c \leftarrow \{\}$
2: **для** $i = 1..N$ **выполнять**
3: | открыть случайное изображение выборки
4: | $c \leftarrow c \cup$ {цвета M слуйчаных пикселей изображения}
5: **конец цикла**
6: *центры* $\leftarrow$ **kmeans**(c, d)
7: *области* $\leftarrow$ **выполнить разбиение Вороного** для цветового пространства с вычисленными *центрами*
8: **вернуть** ⟨*ценры*, *области*⟩

Рисунок 2 – Псевдокод алгоритма квантования цветового пространства

Алгоритм обладает преимуществами над некоторыми существующими алгоритмами. В процессе его работы учитывается информация о частоте включения пикселей заданного цвета в выборку. Также алгоритм не накладывает никаких ограничений на используемое цветовое пространство, в отличие от таких алгоритмов как Median cut, требующих однородности всех осей цветового пространства.

Недостатками предлагаемого алгоритма является высокая ресурсоемкость вычисления: используется итеративный алгоритм k-средних и алгоритм вычисления разбиения Вороного. Если при работе алгоритма k-средних не был найден глобальный минимум, результат квантования может быть не оптимален.

При использовании алгоритма также встает вопрос представления результатов его работы в памяти вычислительной системы. Для эффективного хранения выходных данных, т. е. областей цветового пространства могут использоваться такие структуры данных как k-мерные деревья, или их частный случай — деревья октантов.

Оценка эффективности метода квантования

Для проверки эффективности предлагаемого метода было проведено сравнение предлагаемого метода и некоторых существующих методов квантования пространства. Для этого были построены две выборки, содержащие 50 000 цветов, извлеченных из 100 изображений базы данных “Web Gallery of Art”. Первая выборка была использована для квантования цветового пространства, вторая для вычисления ошибки квантования.

Тестирования проводилось как на всех изображениях БД, так и по изображениям одной из ее категории для проверки адаптируемости алгоритма к специфике изображений некоторой предметной области. Тестирование проводилось на нормированном цветовом пространстве RGB и HSV.

Для вычисления ошибки квантования была использована формула, полученная при приведении (3) к дискретному виду:

$$E(d,q)=\sum_{i=1}^{N}\left[\int_{U} d(x,q(x))\cdot p_i(x)dx\right]=\frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{M_i}\sum_{j=1}^{M_i} d(x_{ij},q(x_{ij}))\right] \quad (4)$$

где N — количество изображений выборки,

M_i — количество точек i-го изображения выборки.

В качестве функции расстояния в пространстве RGB была использована евклидова метрика:

$$d\left(\langle r_1,g_1,b_1\rangle,\langle r_2,g_2,b_2\rangle\right)=\sqrt{(r_2-r_1)^2+(g_2-g_1)^2+(b_2-b_1)^2} \quad (5)$$

В пространстве HSV была использована метрика, полученная при переводе координат из цилиндрической системы HSV в декартову и представлении пространства в виде конуса:

$$d\left(\langle h_1,s_1,v_1\rangle,\langle h_2,s_2,v_2\rangle\right)=\sqrt{(v_1s_1\cos h_1-v_2s_2\cos h_2)^2+(v_1s_1\sin h_1-v_2s_2\sin h_2)^2+(v_2-v_1)^2} \quad (6)$$

На практике также можно использовать следующую метрику на пространстве HSV, требующую меньших вычислительных затрат:

$$d\left(\langle h_1,s_1,v_1\rangle,\langle h_2,s_2,v_2\rangle\right)=\sqrt{\left(\frac{\min(|h_2-h_1|,2\pi-|h_2-h_1|)}{\pi}\right)^2+(s_2-s_1)^2+(v_2-v_1)^2} \quad (7)$$

Результаты тестирования эффективности алгоритма: ошибка, рассчитанная по формуле (4) и дисперсия отклонений цвета и результата квантования приведена в табл. 1. При тестировании значение k принималось равным 64. Приведенный в табл. 1 алгоритм линейного разбиения осуществляет разбиение пространства плоскостями, параллельными осям: в пространстве RGB производится интервала [0; 1] на 4 равные части по каждой из осей; в пространстве HSV интервал [0; 1] насыщенности и значения разбивается на 2 и 4 равные интервала соответственно, диапазон насыщенности [0; 2π) разбивается на 8 равных интервалов. Тестирование алгоритма Median cut в пространстве HSV не производилось из-за специфики работы алгоритма.

Таблица 1 – Результат проверки тестирования алгоритма

№ п/п	Алгоритм	Цветовое пространство	На категории изображений		На всем наборе	
			Ошибка	Дисперсия	Ошибка	Дисперсия
1	Линейное разбиение	RGB	0.1224435	0.0015371	0.1215436	0.0015651
2	Линейное разбиение	HSV	0.1174651	0.0026843	0.1087013	0.0027106
3	Median cut	RGB	0.0518553	0.0010181	0.0613536	0.0017857
4	Предлагаемый метод	RGB	0.0454453	0.0005231	0.0474501	0.0007448
5	Предлагаемый метод	HSV	0.0493524	0.0006500	0.0516378	0.0011698

По результатам тестирования можно сделать вывод о том, что метод на основании k-средних дает меньшие значения ошибки для всех категорий изображений и рассматриваемых цветовых пространств. Кроме этого, проведен анализ дисперсии ошибок, в результате чего можно сделать вывод, о том, что метод k-средних дает большую эффективность, если его применять к определенной категории изображений, а не ко всему набору.

Заключение

В настоящей статье были проанализированы существующие подходы к решению задачи поиска изображений по содержимому (разработке CBIR-систем). В контексте решения задачи поиска был предложен алгоритм квантования цветового пространства с целью оптимизации процесса извлечения цветовых и текстурных признаков из графической информации. Предложенный алгоритм был сравнен с некоторыми существующими алгоритмами квантования. В качестве критерия сравнения была использована величина ошибки, возникающей при квантовании каждым из рассмотренных методов.

В результате анализа алгоритма был сделан вывод о возможной применимости метода на основании k-средних для поиска изображения определенной категории.

В статье не проанализирована эффективность алгоритма в минимизации семантического разрыва между математическим представлением цвета и его интерпретацией человеком. Также, не проанализировано использование алгоритма на других цветовых пространствах, в частности, на пространствах, минимизирующих нелинейность восприятия цвета человеком. Эти вопросы будут рассмотрены в дальнейших исследованиях.

Литература

1. Li J. Real-Time Computerized Annotation of Pictures / J. Li, J.Z. Wang // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2008. – Т. 30. – № 6. – С. 985-1002.
2. Image retrieval: ideas, influences, and trends of the new age / Datta R. [et al.] // Acm Computing Surveys. - 2008. - № 2 (40).
3. Mojsilovic A. A method for color content matching of images / A. Mojsilovic, J. Hu // 2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532). – 2000. – Т. 2. – С. 649-652.
4. The QBIC Project: Querying Images by Content Using Color, Texture, and Shape. The QBIC Project / T.J.W.I.R. Center [et al.] – IBM Thomas J. Watson Research Division, 1993. – 20 p.
5. Mäenpää T. Classification with color and texture: jointly or separately? / T. Mäenpää, M. Pietikäinen // Pattern Recognition. – 2004. – Т. 37. – Classification with color and texture. – № 8. – С. 1629-1640.
6. A comparative study of texture measures with classification based on feature distributions / M. Pietikainen [et al.] // Pattern Recognition. – 1996. – Т. 29. – С. 51–59.
7. Vasconcelos N. A unifying view of image similarity / N. Vasconcelos, A. Lippman // Proceedings 15th International Conference on Pattern Recognition. ICPR-2000. – 2000. – Т. 1. – С. 38-41.
8. Парасич А.В. Методы на основе цветовых гистограмм в задачах обработки изображений / А.В. Парасич, В.А. Парасич // Nauka-rastudent.ru. – 2015. – № 06 (18) / [Электронный ресурс] – Режим доступа. – URL: http://naukarastudent.ru/18/2742/
9. Rubner Y. A metric for distributions with applications to image databases / Y. Rubner, C. Tomasi, L.J. Guibas // Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271). – 1998. – С. 59-66.
10. Stricker M. Spectral covariance and fuzzy regions for image indexing / M. Stricker, A. Dimai // Machine Vision and Applications. – 1997. – Vol. 10. – № 2. – P. 66-73.
11. Mota C. Optimal image quantization, perception and the median cut algorithm / C. Mota, J. Gomes, M.I.A. Cavalcante // Anais da Academia Brasileira de Ciências. – 2001. – Т. 73. – № 3. – С. 303-317.
12. Web Gallery of Art, searchable fine arts image database [Электронный ресурс]. – URL: https://www.wga.hu/index.html (дата обращения: 26.10.2018).

Полетаев В.А., Коломойцева И.А. Квантование цветового пространства в контексте решения задачи поиска изображений по содержанию. *В данной статье отмечается актуальность задачи поиска изображений по содержанию, описываются существующие подходы и методы решения задачи. В статье проведен экспериментальный анализ использования алгоритма квантования цветового пространства методом k-средних при выделении цветовых и текстурных признаков изображений. Произведена оценка алгоритма в сравнении с существующими методами квантования.*

Ключевые слова*: поиск изображений, графическая БД, квантование, цветовое пространство, метод k-средних*

Vladislav Poletaev, Irina Kolomoitseva Quantization of color space in the context of the content-based image retrieval. *The relevance of the development of content-based image retrieval method is noted in the article. Existing approaches and methods of solving this problem are described. The usage of k-means algorithm for color space quantization for extraction of color and texture features of images is experimentally analyzed in the article. The algorithm is evaluated in comparison to existing quantization methods.*

Key words*: image retrieval, graphical DB, quantization, color space, k-means*

УДК 004.55

Программное обеспечение для автоматизации производственной деятельности службы организации дорожного движения

Полищук С.Ю., Незамова Л.В.
Донецкий национальный технический университет
Кафедра программной инженерии
E-mail: sp5952344@gmail.com, larkot@mail.ru.

Полищук С.Ю., Незамова Л.В. Программное обеспечение для автоматизации производственной деятельности службы организации дорожного движения. *В работе освещены задачи, которые решает система, обоснована целесообразность применения программного модуля для автоматического учета производственных показателей при выполнении работ по нанесению дорожной разметки службы организации дорожного движения, также определены перспективы ее использования.*

Ключевые слова: *мобильная платформа, платформа 1с 8.3, дорожная разметка, синхронизация, PHP, Java Script.*

Введение

Важной частью организации безопасности дорожного движения является правильное нанесение дорожной разметки. Она должна соответствовать техническим паспортам, составленным на каждый участок дороги, а также наноситься в соответствии с ДСТУ 2587:2010 [1].

При выполнении работ по нанесению дорожной разметки перед руководителем отдела организации дорожного движения ставятся следующие задачи:

— расчет и списание материалов;

— организация и учет рабочего времени сотрудников отдела, непосредственно выполняющих данные работы;

— контроль качества выполнения поставленной задачи;

— учет объемов выполненных работ;

— предоставление информации для составления сметной документации или бухгалтерской отчетности.

Были проанализированы отчеты о внедрении программных продуктов 1С на трех предприятиях:

1. ООО «Дорожное строительство «АЛЬТКОМ» [2].
2. Объединение «Дорстройпроект» [3].
3. Центр Организации Дорожного Движения (ЦОДД), г. Москва [4].

При внедрении программного комплекса на предприятии ООО «Дорожное строительство «АЛЬТКОМ» был выбран продукт «1С: Предприятие. Управление строительной организацией». В результате использования программного продукта были автоматизированы функции закупок (снабжения), управление отношениями с поставщиками, учет товарно-материальных ценностей (ТМЦ), кадровый учет, управление логистикой, продажи (сбыт), управленческий учет, бухгалтерский учет и др. [2].

При внедрении программного комплекса на предприятии Объединение «Дорстройпроект» был выбран продукт «Управление производственным предприятием (8.0)», в результате чего автоматизирован бухгалтерский учет, оперативный торговый и складской учет [3].

В Центре Организации Дорожного Движения (г. Москва) при внедрении программного комплекса использовался продукт «1С: Бухгалтерия государственного учреждения ПРОФ» и соответственно автоматизирован бухгалтерский учет [4].

Вышеназванные продукты представляют стандартизированные только для определенных задач комплексы и не включают необходимый функционал для автоматизации учета участка по разметке автодорог. В сложившейся ситуации необходимо самостоятельно выполнять все многочисленные расчеты и вести учет, а затем предоставлять отчеты в соответствующие отделы (бухгалтерия или материально-технический отдел), где представленные данные будут учитываться в программе на базе 1С. Также рассмотренные программные комплексы разрабатываются и поддерживаются штатом программистов, что значительно затрудняет их доработку.

Сложность составления отчетов отдела по разметке заключается в большом количестве коэффициентов для расчета площади окрашиваемой поверхности, расхода краски и других материалов, которые

регламентируются государственным стандартом [1]. Цель работы – разработать и обосновать целесообразность применения программного комплекса для учета показателей производственной деятельности участка по разметке автодорог.

Для этого на базе платформы «1С: Предприятие 8.3» для оптимизации учета производственных показателей участка по разметке автодорог было разработано два приложения: «1С Управление участком по разметке автодорог» и «1С АРМ мастера по разметке автодорог».

В созданных конфигурациях предусмотрено наличие справочников, документов, механизмов учета, отчетов. Тем самым приложение позволяет облегчить и синхронизировать работу сотрудников, а также, отслеживать географическое расположение при анализе выполненных работ и планирования дальнейшей деятельности.

Рассмотрим работу в предложенном программном комплексе.

1С АРМ мастера по разметке автодорог (мобильное приложение)

«1С АРМ мастера по разметке автодорог» - это мобильное приложение, разработанное для автоматизации расчетов, производимых мастером по разметке автодорог, а также для записи географических координат элементов наносимой разметки.

В приложении реализована возможность внесения исходных данных и автоматический расчет площади и расхода краски всех окрашенных элементов за смену (см. рис. 1).

Данные площади окрашиваемой поверхности каждого элемента, а также нормы списания краски указаны в справочнике «Линии разметки» (см. рис. 1). Эти данные автоматически подставляются в соответствующие поля формы (см. рис. 1) для расчета вышеуказанных показателей.

Также в программе предусмотрена форма для фиксирования географических координат наносимых элементов разметки (см. рис. 1). В этой форме необходимо указать количество элементов, координаты которой необходимо получить. После изменения значения в этом поле происходит получение координат и запись в соответствующие поля формы. Предусмотрены также две кнопки, после активации которых можно посмотреть зафиксированные координаты на карте (см. рис. 3) и обновить координаты.

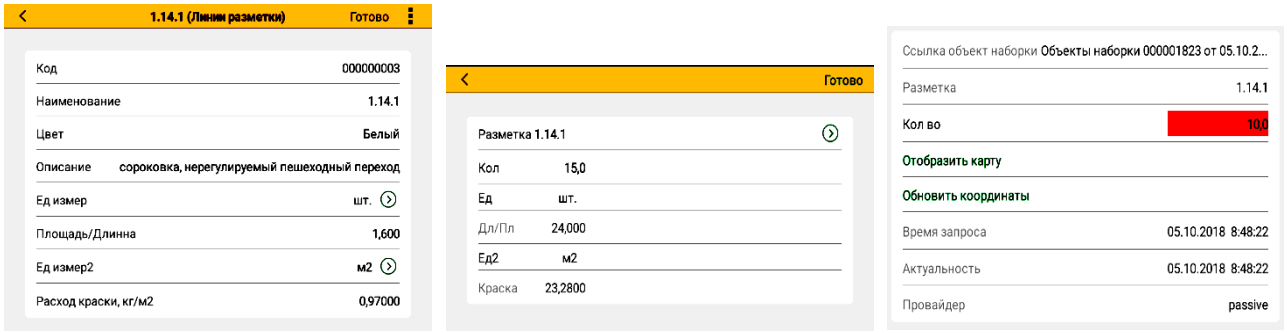


Рисунок 1 – Справочник «Линии разметки», формы учета

Также следует отметить, что объекты «Элементы разметки», при записи которых программа фиксирует координаты точки, являются подчиненными элементами документов «Объекты наборки», общие объемы окраски (длинна, площадь) и расход краски которых подсчитывается при помощи регистра накопления, который фиксирует данные по измерениям «Ссылка на объект наборки» и «Ссылку на линию разметки» а также ресурс: «Количество элементов разметки» (см. рис. 2).

По факту выполненных работ, необходимо произвести обмен данными с центральным сервером. Для этого при активном Wi-fi соединении выбрать пункт главной формы приложения «Отправить данные». Для этого необходимо произвести небольшие настройки передачи данных.

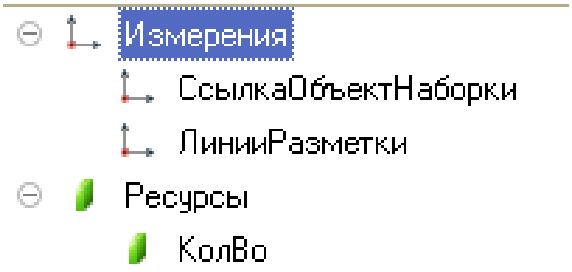


Рисунок 2 – Структура регистра накопления «Элементы разметки по объектам»

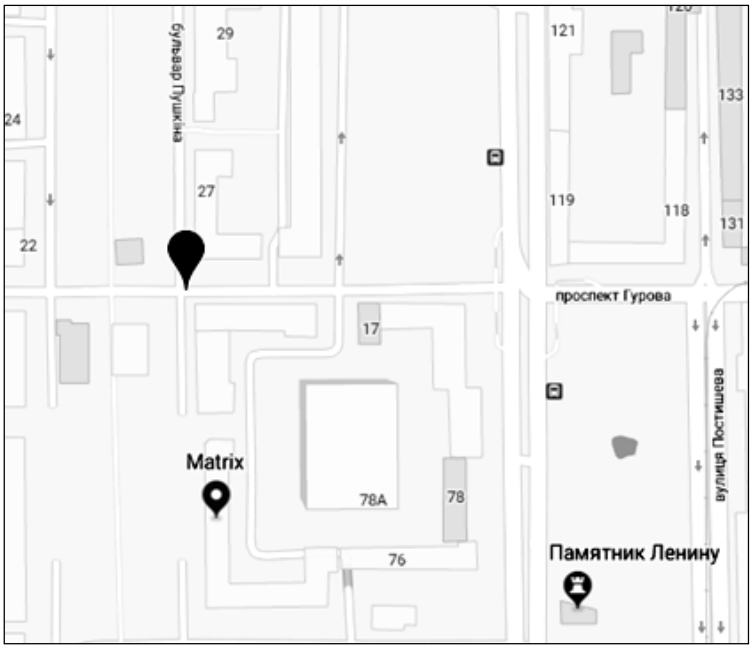


Рисунок 3 – Отображение зафиксированных координат на карте

Модуль для синхронизации данных

Синхронизация данных между приложениями реализована следующими методами:

1. На стационарном ПК установлен Web сервер Apache 2.4.
2. В приложении «1С Управление участком по разметке автодорог» опубликован Web сервис.
3. В приложении «1С Управление участком по разметке автодорог» и «1С АРМ мастера по разметке автодорог» создан план обмена «Обмен с мобильным устройством» в котором выбраны все необходимые объекты для обмена.
4. В приложении «1С Управление участком по разметке автодорог» узлы плана обмена настроены так, как показано на рисунке (см. рис. 4). Здесь ЦБ1 - это текущий узел обмена (помечен зеленой точкой), а М4 и М5 - это узлы, по которым регистрируются изменения.
5. В приложении «1С АРМ мастера по разметке автодорог» также создаются узлы обмена, но в обратном порядке (см. рис. 5). Здесь текущий узел – это М5, а БЦ1 – это узел, по которому фиксируются изменения.

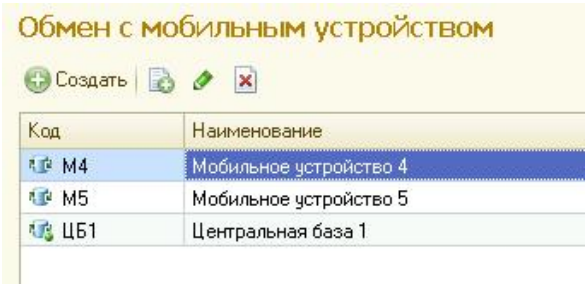


Рисунок 4 – Узлы обмена в приложении «1С Управление участком по разметке автодорог»

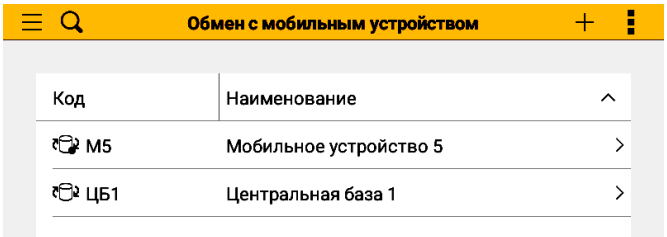


Рисунок 5 – Узлы обмена в приложении «1С АРМ мастера по разметке автодорог»

В конфигураторе мобильного приложения была создана общая команда «Выполнить синхронизацию», которая обращается к серверной функции «Выполнить синхронизацию на сервере» (см. рис. 6).

```
&НаСервере
Функция ВыполнитьСинхронизациюНаСервере()
    Попытка
        // определение веб сервиса
        ВСОпределение   = Новый WSОпределения("http://10.1.191.213/RazmetkaServer/ws/DataExchange.1cws?wsdl");
        ВСервис         = ВСОпределение.Сервисы.Получить("DataExchange", "DataExchange");
        ВТочкаВхода     = ВСервис.ТочкиПодключения.Получить("DataExchangeSoap");
        ВОперация       = ВТочкаВхода.Интерфейс.Операции.Получить("Синхронизация");

        // переменная хранит выгрузку XML, которую сжимаем и помещаем в хранилище значений
        Данные = Новый ХранилищеЗначения(ЗарегестрироватьВыгрузку(), Новый СжатиеДанных(9));
        // упаковываем Данные в пакет XDTO
        ДанныеXDTO = ВСОпределение.ФабрикаXDTO.Создать(ВОперация.Параметры.Получить("Данные").Тип, Данные);

        ВСПрокси = Новый WSПрокси(ВСОпределение, "DataExchange", "DataExchange", "DataExchangeSoap");
        // вызывается веб сервис который передает управление на сервер
        Ответ = ВСПрокси.Синхронизация(ДанныеXDTO);
        ПринятьИзмененияПоПлану(Ответ.Получить());
        Возврат Истина;

    Исключение
        Сообщить(ОписаниеОшибки());
        Возврат Ложь;
    КонецПопытки;
КонецФункции
```

Рисунок 6 – Функция, выполняющая синхронизацию данных

Данная функция создает ВС Определение, получает зарегистрированные изменения из плана обмена из функции Зарегистрировать Выгрузку (см. рис. 7) в формате сериализованной строки XML, упаковывает её в Хранилище значений (сжимает данные) и отправляет их на сервер путем вызова ws-ссылки, опубликованной приложением сервером «1С Управление участком по разметке автодорог».

```
Функция ЗарегестрироватьВыгрузку()
    ЗаписьXML = Новый ЗаписьXML;
    ЗаписьXML.УстановитьСтроку();
    ЗаписьСообщения = ПланыОбмена.СоздатьЗаписьСообщения();
    Узел = ПланыОбмена.ОбменСМобильнымУстройством.НайтиПоКоду("ЦБ1");
    ЗаписьСообщения.НачатьЗапись(ЗаписьXML, Узел);

    // после выполнения проедуры ВыбратьИзменения сообщения меняют свой номер
    ВыборкаИзменений = ПланыОбмена.ВыбратьИзменения(Узел, ЗаписьСообщения.НомерСообщения);
    Пока ВыборкаИзменений.Следующий() Цикл
        ОбъектОбмена = ВыборкаИзменений.Получить();
        ЗаписатьXML(ЗаписьXML, ОбъектОбмена);
    КонецЦикла;
    ЗаписьСообщения.ЗакончитьЗапись();
    Возврат ЗаписьXML.Закрыть();
КонецФункции
```

Рисунок 7 – Функция, выполняющая сериализацию данных в формат XML.

Если сервер получил данные, он отправляет в ответ свои данные, таким образом мобильное приложение фиксирует, что данные получены и принимает переданные изменения.

1С Управление участком по разметке автодорог

В рамках данного проекта было создано приложение для ПК «1С Управление участком по разметке». В этом приложении реализованы:

1. Учет объемов выполненных работ (в точности соответствует описанному выше учету в мобильном приложении).
2. Отчет по объектам представляет собой детализированную информацию об объемах выполненных работ по конкретному объекту за определенный период времени. (см. рис. 8)
3. Отчет по видам разметки – предоставляет детализированную информацию об объемах выполненных работ по видам разметки за определенный период времени.
4. Отчет о расходе краски за определенный период.

5. Отчет об объемах проделанных работ машинистов, которые эти работы выполняли за определенный период времени (см. рис. 8).

Отбор: Наименование разметки Равно "1.14.1"

Наименование разметки.Цвет	Площадь/Длинна	Расход краски, кг
Белый	464,000	450,0800
Итого	464,000	450,0800

Наименование разметки	Ед	Наименование разметки.Цвет	Площадь/Длинна	Расход краски, кг
Наименование объекта				
1.14.1	м2	Белый	464,000	450,0800
ул. Ляшенко			59,200	57,4240
ул. 60-летия СССР			112,000	108,6400
ул. Степаненко			32,000	31,0400
Гимназия №150, ул. Армавирская, 23а			11,200	10,8640
Школа №46, ул. Лермонтова			12,800	12,4160
ул. Сомова			75,200	72,9440
ул. Югославская			52,800	51,2160
ул. Короленко (Моспино)			83,200	80,7040
ул. Литке			25,600	24,8320
Итого			464,000	450,0800

Параметры: Дата1: 29.08.2018
Дата2: 31.08.2018

Машинист		Длинна	Площадь	Часов
Дата Н	Смена			Ф
Бошган В.Н.		2 646,000	724,310	16,66
29.08.2018	2 смена	2 218,000	316,990	8,16
30.08.2018	2 смена	428,000	133,570	3,02
31.08.2018	2 смена		273,750	5,48
Дзисов С.В.		5 094,000	386,665	11,91
29.08.2018	1 смена		168,700	3,37
30.08.2018	1 смена	2 256,000	109,720	4,04
31.08.2018	1 смена	2 838,000	108,245	4,49
Растопчин Е.Ю.		2 943,000	349,220	9,40
29.08.2018	1 смена	2 943,000		2,41
30.08.2018	1 смена		241,940	4,84
31.08.2018	1 смена		107,280	2,15
Итого		10 683,000	1 460,195	37,96

Рисунок 8 – Отчёт по видам разметки и по объемам проделанных работ

Также координаты, зафиксированные с мобильного устройства, можно посмотреть на карте с возможностью отбора информации по объектам нанесения, виду разметки и периоду времени (см. рис. 9).

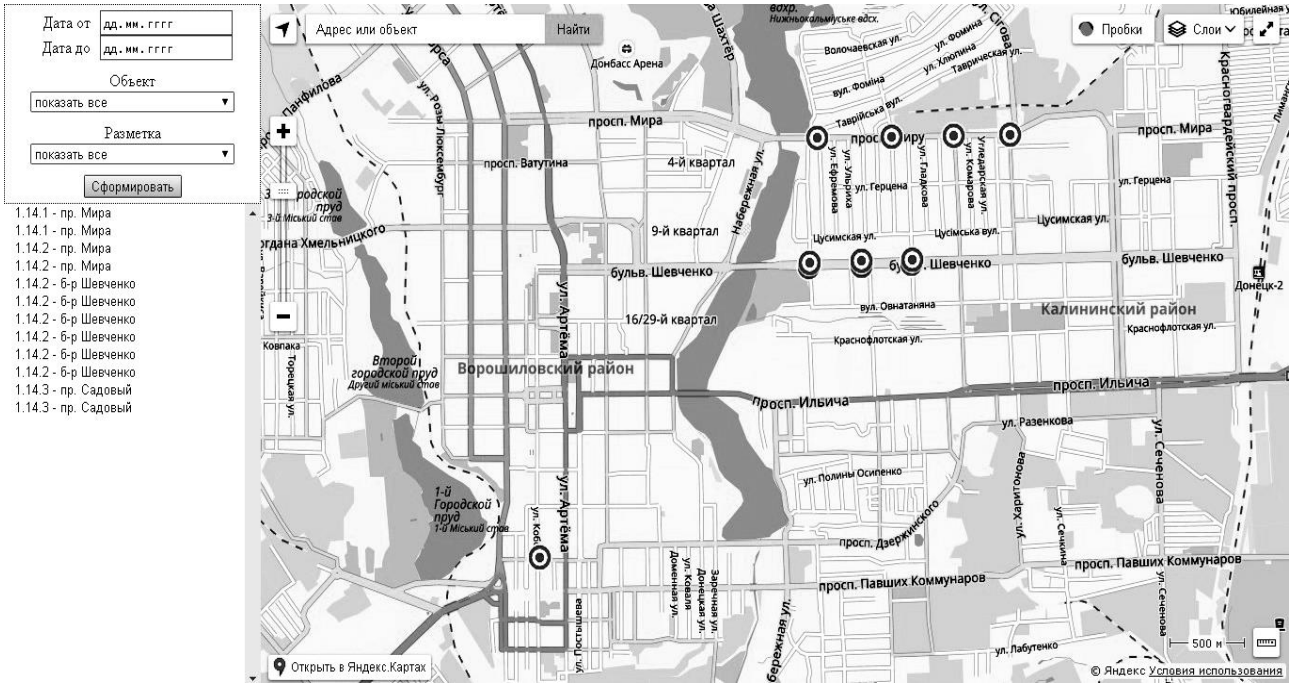


Рисунок 9 – Зафиксированные координаты объектов

При открытии web страницы в браузере выполняется загрузка информации о точках, координаты которых были зафиксированы. Загрузка выполняется посредством POST запроса (см. рис. 10) на выполнение кода PHP скрипта (см. рис. 11), который в свою очередь обращается к ws-ссылке, опубликованной приложением «1С Управление участком по разметке автодорог» на сервере Apache.

```
$.ajax ({
    url: "http://10.1.191.213/MapsInfo/php/GetMarkingObject.php",
    type: "POST",
    data: ({stringData: 'some data'}),
    dataType: "text",
    beforeSend: funcBefore,
    success: funcSuccess
    });
```

Рисунок 10 – POST запрос на выполнение кода PHP скрипта

```php
<?php
    ini_set('soap.wsdl_cache_enabled', 0);
    ini_set('soap.wsdl_cache_ttl', 0);

    $client = new SoapClient("http://10.1.191.213/RazmetkaServer/ws/DataExchange.1cws?wsdl",
        array(
            'login' => "",
            'password' => "",
            'soap_version' => SOAP_1_2,
            'cache_wsdl' => WSDL_CACHE_NONE,
            'tracer' => true,
            'features' => SOAP_USE_XSI_ARRAY_TYPE
        )
    );


    $result = $client->GetMarkingObject();
    echo $result->return;
?>
```

Рисунок 11 – PHP скрипт вызова ws-ссылки

Выводы

Обоснована актуальность тематики автоматизации производственной деятельности и разработки программных модулей для выполнения работ по нанесению дорожной разметки. Разработано мобильное приложение, приложение для ПК, а также модуль для синхронизации данных между ними и web страница для отображения фиксированных координат.

Литература

1. ДСТУ 2587:2010 – Дорожная разметка. Общие технические требования.
2. Внедрение 1С: Предприятие 8. Управление строительной организацией для Украины в ООО "Дорожное строительство "АЛЬТКОМ" https://solutions.1c.ru/projects/projects.html?project_id=528603
3. Автоматизация крупного предприятия в отрасли дорожного строительства на базе «1С: Управление производственным предприятием 8» http://www.keyelement.ru/index.php?name=Pages&op=view&id=45
4. Центр Организации Дорожного Движения (ЦОДД) г. Москва http://1c.ru/rus/partners/solutions/solution.jsp?SolutionID=326046

Полищук С.Ю., Незамова Л.В. Программное обеспечение для автоматизации производственной деятельности участка по разметке автодорог. *В работе освещены задачи, которые решает система, обоснована целесообразность применения программного модуля для автоматического учета производственных показателей при выполнении работ по нанесению дорожной разметки службы организации дорожного движения, также определены перспективы ее использования.*

Ключевые слова: *мобильная платформа, платформа 1с 8.3, дорожная разметка, синхронизация, PHP, Java Script.*

Polishchuk S.Yu., Nezamova L.V. Software to automate the production activities of the site on road markings. *The paper covers the tasks that solve the system, the validity of the use of a software module for automatic recording of production indicators when performing road marking works for the traffic management service, as well as their prospects for its use.*

Key words: *mobile platform, platform 1c 8.3, road marking, synchronization, PHP, Java Script.*

УДК 004.021

Разработка функциональной и современной системы учёта и защиты данных в контексте малого предприятия для складской документации

Ржевский К.В.
Донецкий национальный технический университет
nory4ik4@mail.ru

Ржевский К.В. Разработка функциональной и современной системы учёта и защиты данных в контексте малого предприятия для складской документации. *Статья посвящена вопросу защиты данных в контексте малого предприятия. В статье описан пример реализации продукта, позволяющего вести учёт данных и обеспечивать их сохранность и защиту.*

Ключевые слова: сохранность данных, защита данных, учёт данных, малое предприятие, базы данных, криптография.

Введение

Концепция защиты данных применяется ведущими компаниями во многих отраслях деятельности. Наиболее популярное их использование замечено в здравоохранении, торговле, телекоммуникациях, в финансовых компаниях, а также в государственном управлении.

Защита данных (криптографическими путями) является обязательной мерой для предприятий.

Актуальность работы: в данное время защита данных в контексте малого предприятиях является актуальной проблемой, рассматриваются такие подходы:

- аутентификация;
- авторизация;
- делегирование прав;
- шифрование данных;
- защита от sql - инъекций;
- защита от подмены данных;
- тестирование на наличие типовых уязвимостей.

Цель работы: исследование и анализ функциональных возможностей программного обеспечения для работы с информацией в условиях малого бизнеса.

Информационные системы предприятия малого бизнеса как объект защиты информации

Структура предприятия малого бизнеса предполагает соответствующую структуру информационной системы. Опишем компоненты структуры, задачи автоматизации и проблемы, имеющиеся в этой сфере. В качестве примера рассмотрим подсистему управления складским помещением.

Склад — помещение, предназначенное для хранения материальных ценностей.

Автоматизация работы склада. Построение современной эффективной системы управления складом заключается в сокращении объёма документа оборота.

К известным складским проблемам можно относятся:

- невысокая скорость обработки прихода/комплектации заказа;
- пересортица, ошибки кладовщиков и грузчиков при приеме, размещении, подборе товара;
- длительность проведения инвентаризации;
- данные проблемы особенно остры для складов с большим товарным ассортиментом.

Для компаний, имеющих обширные склады, которые стремятся к тому, чтобы:

- увеличить скорость отгрузки;
- увеличить скорость обработки поступлений;
- оперативно проводить инвентаризацию.

Логистика складирования. Современный крупный склад — это сложное функционирующий отдел, состоящее из модулей.

Основное назначение склада — хранение запасов, и обеспечение постоянного и выполнения заказов. Основные функции склада:

− переделывание производственного спектра в соответствии с возможностью покупки; эта функция приобретает в логистике, где торговый выбор включает большой список товаров различных производителей;

− помещение и содержание, позволяющие выравнивать временную различие между выпуском продукции и - ее потреблением;

− консолидирование и перевозка грузов.

Развитие компьютерных средств обеспечило создание и широкое использование систем обработки данных. Разработана информационная система для обслуживания работы склада.

Проблема сохранности информации делает актуальной задачей разработку соответствующей подсистемы защиты информации. Предлагается обеспечить построение подсистемы защиты информации путем ее шифровки, т.е. – методом криптозащиты.

Перечислим требования, предъявляемые к программному обеспечению информационной системы с функцией криптозащиты.

1. Базовые требования к информационной подсистеме.

1.1. Программное обеспечение должно обеспечивать учет информации о материале, а именно:

− наименование материала;

− марка;

− цена.

1.2. Программное обеспечение должно получить и понятный интерфейс.

2. Требования к информационной подсистеме с точки зрения защиты информации.

Программное обеспечение для учета товара должно обеспечивать:

− конфиденциальность доступа;

− разделение доступа пользователей к функциям информационной системы;

− хранение информации об создателе записи, времени создания, модификации и удаления любой информации в системе;

− целостность, актуальность и непротиворечивость информации.

Предлагаемая структурная и функциональная схема защищенной информационной подсистемы «склад»

Рассмотрим предлагаемую структуру информационной системы склад, построенную с учетом возможности защиты информации.

Компьютеризированная подсистема представляет собой комплекс технического, математического, информационного и программного обеспечений. Программное же обеспечение включает:

− специальное обеспечение;

− общесистемное обеспечение;

− инструментальные средства.

Для построения функционального описания подсистемы были использованы пакет BPWin, с помощью которого были построены следующие диаграммы в соответствии с моделью IDEF 0:

− контекстная диаграмма;

− диаграмма декомпозиции.

Описание функции «Накладная на списание»

Эта функция должна содержать информацию о продаже товаров. На рисунке 3 отражена диаграмма декомпозиции для функции «Накладная на списание», которая подробнее описывает данную функцию. На этом рисунке видно, на какие типы делится эта функция.

Результаты решения: результатами работы являются те, которыми доволен кладовщик.

Описание функции «Журнал приема»

Эта функция должна предоставить администратору данные о поставленном товаре согласно накладной поставщика. Внесение данных в базы данных, а именно:

− наименование;

− дата;

− единица измерения;

− количество.

На рисунке 4 отражена диаграмма декомпозиции для функции «Журнал приема», которая подробнее описывает данную функцию.

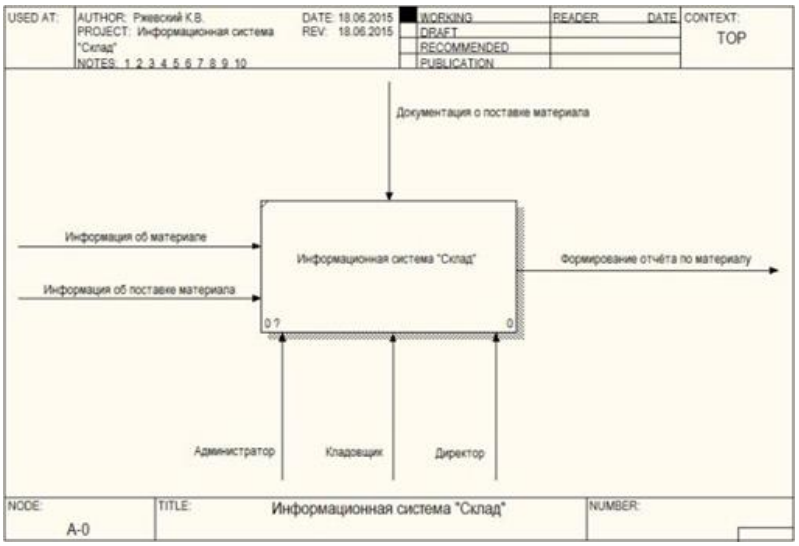


Рисунок 1 – Контекстная диаграмма

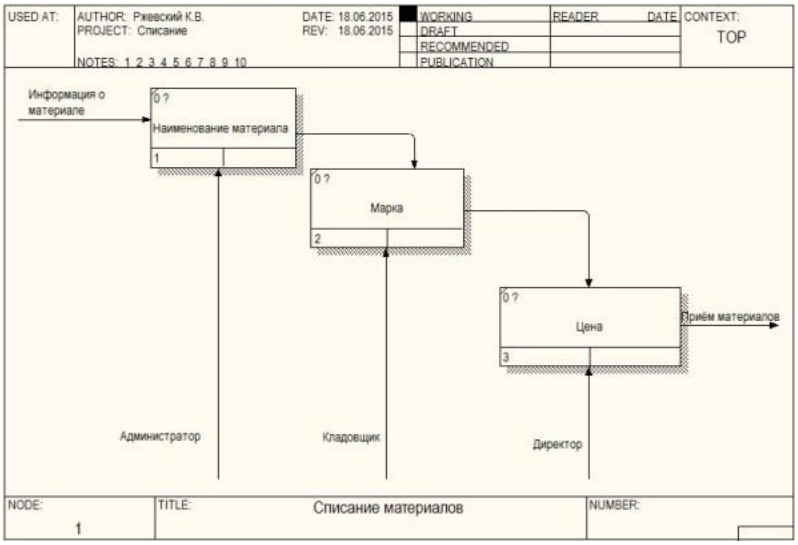


Рисунок 2 – Диаграмма декомпозиции

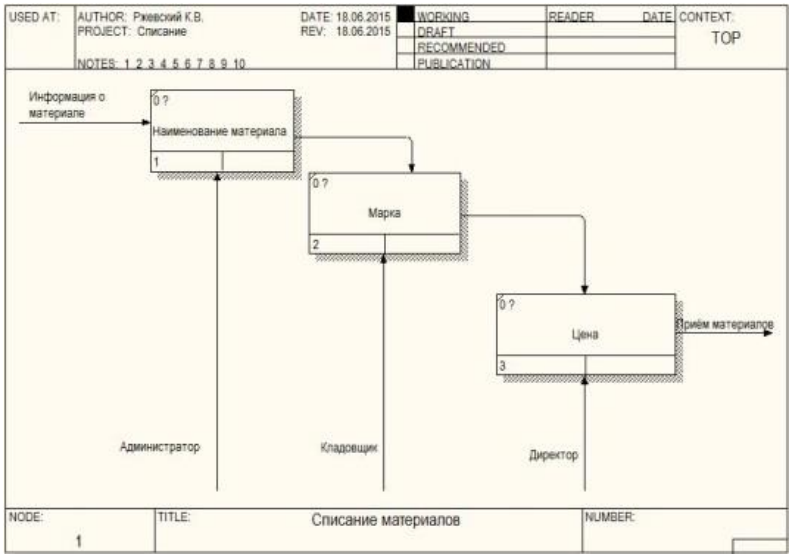


Рисунок 3 - Диаграмма декомпозиции для функции «Накладная на списание»

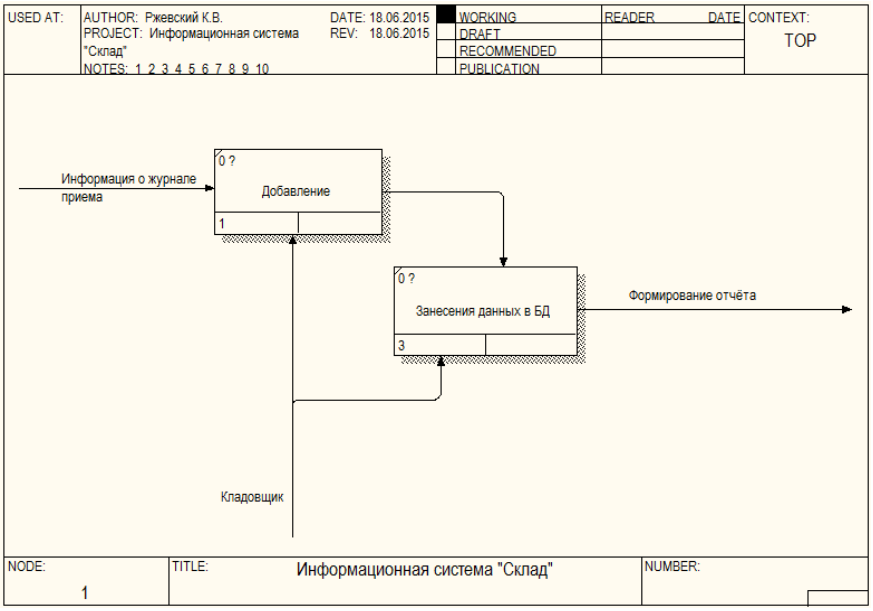


Рисунок 4 - Диаграмма декомпозиции для функции «Журнал приема»

Назначение и характеристика: с помощью программного обеспечения осуществляется учет получаемого материала.

Результаты решения: новая запись в базе данных.

Описание функции «Накладная на списание»

Эта функция должна предоставить администратору данные о поставленный товар согласно накладной поставщика. Внесение данных в базу данных, а именно:

- дата;
- количество;

На рисунке 5 отражена диаграмма декомпозиции для функции «Накладная на списание», которая подробнее описывает данную функцию.

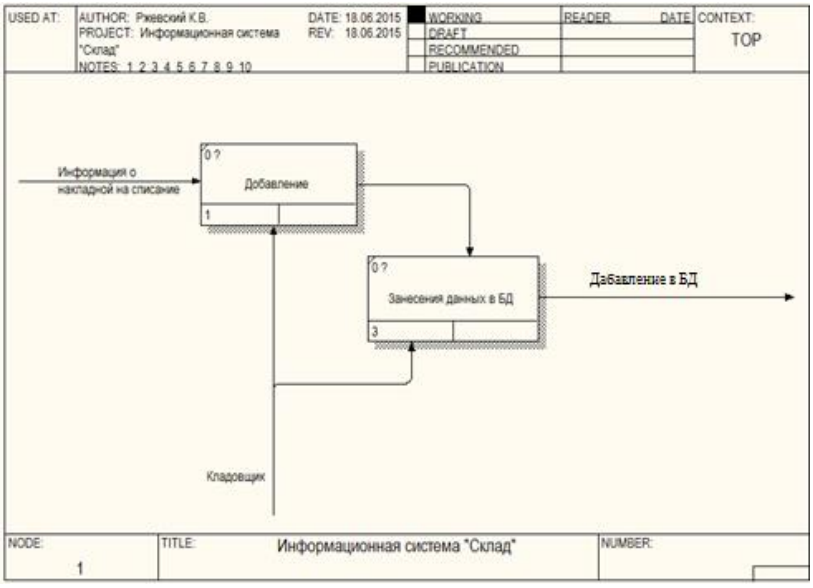


Рисунок 5 - Диаграмма декомпозиции для функции «Накладная на списание»

Назначение и характеристика: с помощью программного обеспечения осуществляется учет поставленного товара.

Результаты решения: новая запись в базу данных.

Описание функции «Формирование отчета»

Эта функция должна выводить отчеты за период директору или бухгалтеру с базы данных, а именно:

- реестр поставленных материала;
- реестр имеющихся материалов;
- реестр выданных материалов.

На рисунке 6 отражена диаграмма декомпозиции для функции "Формирование отчета», которая подробнее описывает данную функцию.

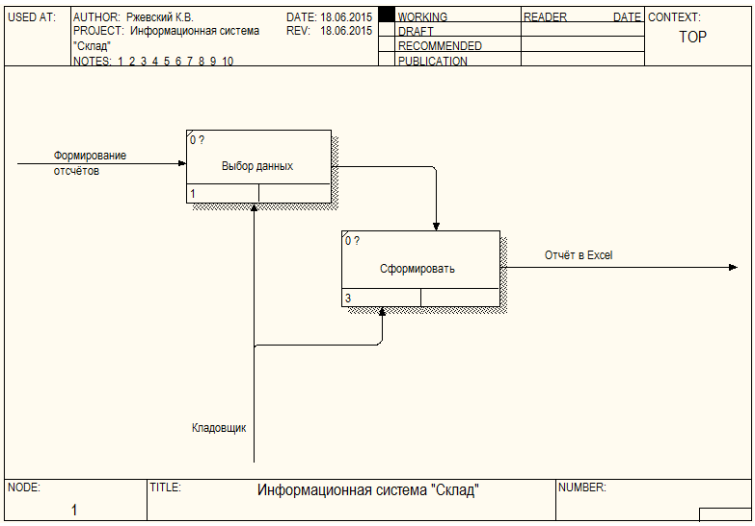


Рисунок 6 – Формирование отчетов

Назначение и характеристика: с помощью программного обеспечения осуществляется формирование отчетов за период.

Используемая информация: данные получены из реестров.

Результаты решения: результатами работы являются исходная информация в виде отчета Excel.

Результаты решения: отчет в Excel.

Описание функции «Криптозащиты»

Эта функция добавлена в стандартные шаблоны производственного цикла для обеспечения базовой защиты информации. Шифрования файла происходит с помощью алгоритма RC2 в режиме CBC. Вектор инициализации и ключ шифруется с помощью алгоритма RSA и записывается начало файла. Расшифровка происходит в обратном порядке с помощью открытого ключа.

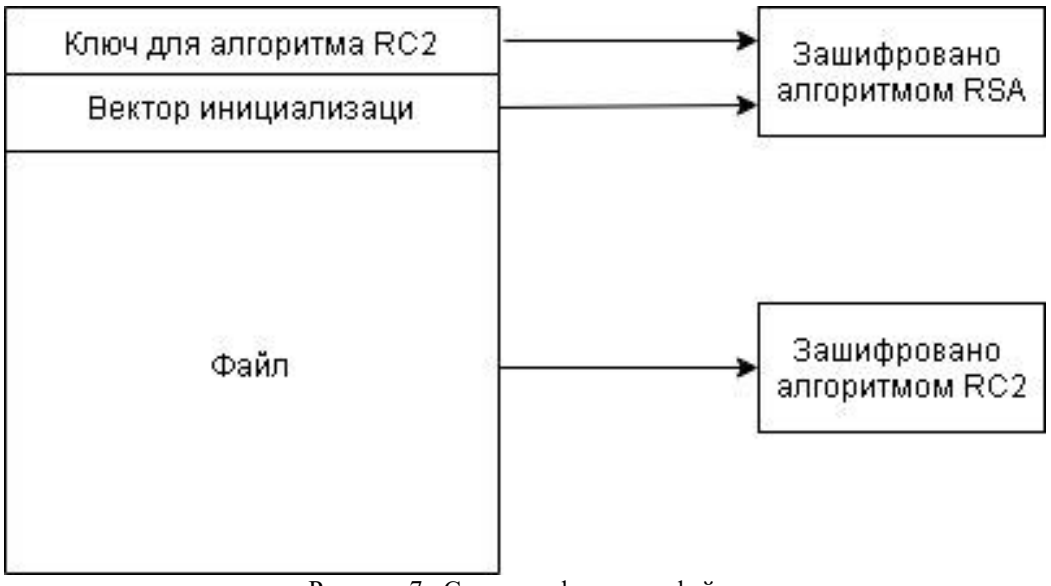


Рисунок 7– Схема шифрования файла

Заключение

Перспектива развития данного типа программного обеспечения растет с развитием малого бизнеса и его компьютеризации.

Выполнена разработка функциональной и современной системы учёта и защиты данных для малого предприятия. Данная разработка заинтересует владельцев фирм с ограниченным бюджетом и невысокими нагрузками на информационную систему, хотя на данном этапе это решение является частным случаем комплекса мер по защите информации на предприятиях, её доработка позволит использовать эту разработку намного большем перечне предприятий. Сейчас же, разработка этой функциональной и современной системы учёта и защиты данных является частным комплексом защиты и учёта.

Литература

1. Основы информационной безопасности. Часть 1: Виды угроз [Электронный ресурс] // Хабр – Режим доступа: https://habr.com/company/vps_house/blog/343110/ – Загл. с экрана.
2. Культин Н. Microsoft Visual C# в примерах и задачах. — Санкт-Петербург "БХВ-Петербург", 2009. — 320 с. [Электронный ресурс] // Авидредерс – Режим доступа: http://avidreaders.ru/read-book/microsoft-visual-c-v-zadachah-i.html – Загл. с экрана.
3. Климов А. П. C#. Советы программистам. — СПб.: БХВ-Петербург, 2008. — 517 с. [Электронный ресурс] // Авидредерс – Режим доступа: http://avidreaders.ru/read-book/c-sovety-programmistam.html – Загл. с экрана.
4. Борисунок Леонтьев Microsoft Visio 2003 Professional. Построение проектов, диаграмм и бизнес-схем в ОС Microsoft Windows XP. — СОЛОН-Р, 2002 г. — 512 стр. [Электронный ресурс] // Бвбук – Режим доступа: http://bwbooks.net/index.php?id1=4&category=comp-lit&author=lemke-dg&book=2006&page=2– Загл. с экрана.
5. RC2 [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/RC2– Загл. с экрана.
6. Режим сцепления блоков шифротекста [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Режим_сцепления_блоков_шифротекста– Загл. с экрана.

Ржевский К.В. Разработка функциональной и современной системы учёта и защиты данных в контексте малого предприятия для складской документации**.** Статья посвящена вопросу защиты данных в контексте малого предприятия. В статье описан пример реализации продукта, позволяющего вести учёт данных и обеспечивать их сохранность и защиту.

Ключевые слова: сохранность данных, защита данных, учёт данных, малое предприятие, базы данных, криптография.

Rzhevsky K.V. *Development of a functional and modern system of accounting and data protection in the context of a small enterprise for warehouse documentation. The article is devoted to the issue of data protection in the context of a small business. The article describes an example of the implementation of a product that allows you to keep records of data and ensure their safety and security.*

Keywords: *data integrity, data protection, data accounting, small business, databases, cryptography.*

УДК 03.026

Метод сокрытия сообщений на основе жаргонов

Сидорчук В.И., Губенко Н.Е., Сипаков Д.С.
Донецкий национальный технический университет
nohohone@gmail.com

Сидорчук В.И., Губенко Н.Е., Сипаков Д.С. Метод сокрытия сообщений на основе жаргонов. *В статье было рассмотрено понятие стеганографии. Была поднята тема о защите информации от несанкционированного доступа. Был проведен сравнительный анализ методов сокрытия сообщения. На основе одного из методов, был разработан авторский метод сокрытия информации.*

Ключевые слова: *стеганография, лингвистика, защита данных, текст, сокрытие текста, стегосообщение*

Введение

Стеганография часто определяется как наука и искусство сокрытия информации в так называемом контейнере, передача которого от одного лица другому не вызывает подозрений. В отличие от криптографии, в задачи которой входит защита собственно сообщения, основной целью стенографии является защита самого факта наличия скрытого сообщения.

Современные стенографические средства в основном работают в информационных средах, которые имеют большую избыточность. Если брать во внимание информацию, содержащую большое количество шумовых данных (таких как звуки или изображения), письменный текст содержит малое количество избыточной информации, которую можно использовать для сокрытия тех или иных данных.

Методы лингвистической стенографии – скрытое внедрение кодированной произвольной информации в текстах, опираясь на лингвистические ресурсы – известны еще со времен средневековья. С развитием компьютерных и информационных технологий средневековые методы лингвистической стенографии возродились на новом уровне и дают возможность, в некоторых случаях, скрыть факт тайной переписки не только от «атематического цензора», который осуществляет мониторинг сетей телекоммуникаций, но и от самого человека.

Постановка проблемы

Проблема защиты информации от несанкционированного доступа возникла еще в древние времена, и с тех пор выделилось направление решения этой проблемы, которое существуют и сегодня, стеганография. Главная задача стеганографии стоит в том, чтобы человек не подозревал, что внутри передаваемой информации, внешне не представляющей абсолютно никакой ценности, содержится секретная информация. Тем самым стеганография позволяет передавать важную информацию через открытые каналы, скрывая сам факт её передачи. Целью статьи является проведение сравнительного анализа методов в лингвистической стеганографии для разработки авторского метода сокрытия сообщения.

Сравнительный анализ методов

Первым методом является – метод жаргонов. Использование жаргона в тексте может озадачить постороннего читателя. В процессе реализации данного метода создается база данных, в которую заносятся слова и соответствующие жаргоны, которые будут заменять слова.

Данный метод несложен в реализации, и используя его, отсутствует подозрение о наличии скрытой информации, в случае сохранения текста в читаемом виде. Однако данный метод имеет и недостатки: база данных слов ограничена перечнем слов, которые используют участники переписки; получатель сообщения также должен знать используемые жаргоны; в случае неудачного употребления жаргона, злоумышленник может понять скрытый смысл сообщения.

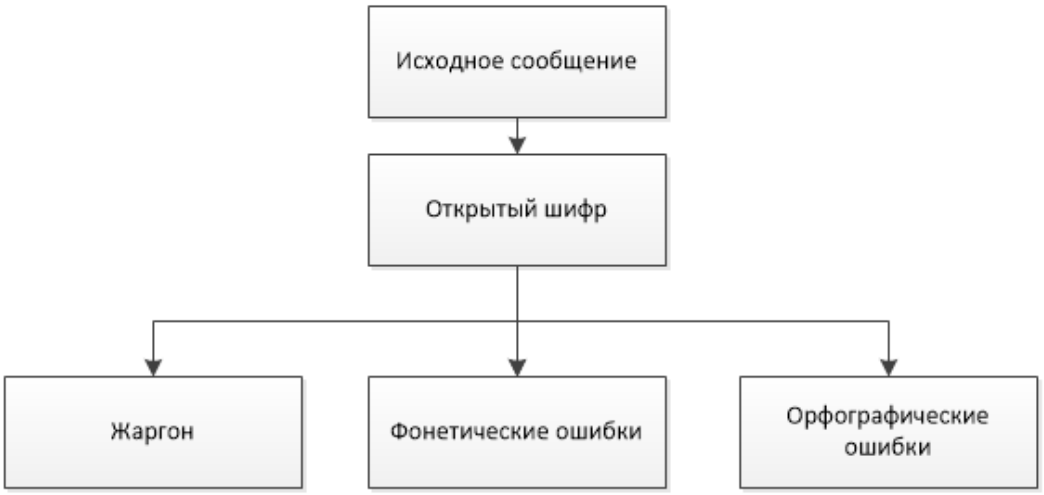


Рисунок 1 – Схема метода жаргонов

Вторым методом является – семаграмма. Семаграмма – это способ сокрыть информацию благодаря знакам или символам. Например, размещение предметов на столе в определенной последовательности, определенная последовательность чисел и т.п. Для стороннего человека, данные знаки не заметны.

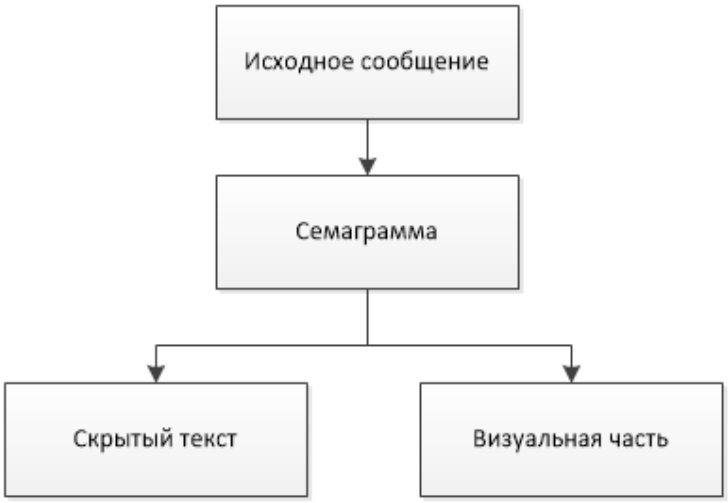


Рисунок 2 – Схема метода семаграмм

Также, существуют текстовые семаграммы. Текстовыми семаграммами являются послания, которые скрыты внутри текста. Для передачи какого-либо сообщения, могут быть использованы: заглавные буквы, пробелы между буквами или словами, особенности подчёркивания и т.п.

Среди достоинств данного метода стоит отметить, что он прост в реализации и незаметен для сторонних лиц.

Недостатком данного метода является тот факт, что из-за простоты реализации метода, злоумышленник достаточно легко может заполучить скрытую информацию.

Третьим методом является – скрытое кодирование. Скрытое кодирование считается частным случаем лингвистической стеганографии, его же наиболее сложно реализовать, однако он обеспечивает высокую скрытность информации. Данный метод использует специальную функцию, которая шифрует и дешифрует сообщение для передачи. В методе используется «контейнер», в который помещается скрываемое сообщение (см. рис.3). После получения сообщения, определенным алгоритмом, скрываемое сообщение достается из «контейнера».

Рисунок 3 – Классическая схема метода скрытого кодирования

Данный метод имеет крепкую стойкость к дешифрованию, но высокую сложность реализации.

Четвертым метод является открытое кодирование. В данном случае имеется в виду следующее: скрытое сообщение размещают в тексте таким образом, чтобы оно не бросалось в глаза стороннему лицу. Когда доходит до анализа, компьютеры и люди демонстрируют разные методы распознавания и по-разному воспринимают стенографические сообщения. Данный метод прост в реализации, но легок в получении скрытой информации сторонним лицом.

Последним методом является – фонетика. Данный метод можно применять, если точно знать, на какой язык запрограммирован фильтр, который ищет слова на том языке, который преимущественно используется жителями в стране. Конечно, нельзя с уверенностью сказать, как в точности запрограммирован фильтр. Но чтобы приблизиться к пониманию, можно использовать фонетически схожие слова. Этот способ наиболее подходит, если вы используете алфавит, отличный от принятого в вашей стране (например, латинский вместо русского).

Данный метод прост в реализации, но информация фактически не скрывается и сторонний человек сможет легко заполучить скрытую информацию и

Разобрав каждый метод, была составлена таблица сравнения, которая учитывает следующие критерии анализа методов: сложность реализации, сложность дешифрования, сложность определения наличия скрываемого сообщения. Каждый метод оценивается баллами от 1 до 3.

Таблица 1 – Сравнительная таблица методов стеганографии

Критерий анализа	Метод лингвистической стеганографии				
	Жаргон	Семаграмма	Скрытое кодирование	Открытое кодирование	Фонетика
Сложность реализации	2	2	3	1	1
Сложность дешифрования	3	2	3	1	2
Легкость определения наличия скрываемого сообщения	3	2	3	1	2

В ходе проведения исследований были проанализированы различные методы лингвистической стеганографии, а также была построена сравнительная таблица с вышеуказанными методами. Они сравнивались по сложности реализации, сложности дешифрования и сложности обнаружения сообщения. Анализируя таблицу 1 выяснилось, что метод скрытого кодирования и метод семаграмм наиболее эффекты в применении.

Разработка авторского метода

Разобрав выше каждый метод, было решено взять за основу для авторского алгоритма, метод жаргона (он же метод синонимических замен). В отличие от других исследуемых методов, данный способ может работать без

источника данных слов, то есть для работы потребуется только стеготекст.

Также, при разработке, за основу был взят механизм замены кодов в соответствии с последовательностью бит сообщения, но в данном случае кодироваться будут не пробелы, а буквы согласно их ASCII кодам. Часть букв русского и английского алфавита имеют одинаковую форму написания, но различные коды (см. рис. 4). Таким образом, появляется возможность использовать данную особенность для кодирования какой-либо последовательности бит.

Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ	Dec	Hex	Символ
64	40	@	96	60	`	128	80	Ђ	160	A0		192	C0	А
65	41	A	97	61	a	129	81	Ѓ	161	A1	Ў	193	C1	Б
66	42	B	98	62	b	130	82	‚	162	A2	ў	194	C2	В
67	43	C	99	63	c	131	83	ѓ	163	A3	Ј	195	C3	Г
68	44	D	100	64	d	132	84	„	164	A4	¤	196	C4	Д
69	45	E	101	65	e	133	85	…	165	A5	Ґ	197	C5	Е
70	46	F	102	66	f	134	86	†	166	A6	¦	198	C6	Ж
71	47	G	103	67	g	135	87	‡	167	A7	§	199	C7	З

Рисунок 4 – Пример таблицы ASCII кодов

Для того чтобы запутать стегоаналитика, необходимо добавить шум в стеготекст. Если при цифровой стеганографии в качестве шума изображения или аудио файлам применялись геометрические преобразования, то в случаи с текстовым форматом нужно провести другие действия. Таким образом, можно выбрать массив букв, которые будут нести закодированное сообщение, а изменение других букв будет осуществляться случайно.

Чтобы повысить степень защиты, следует использовать какой-либо ключ. В качестве ключа будет выступать какое-либо слово или словосочетание, длина которого будет равна длине скрываемого сообщения. При внедрении, в стеготексте будет сделан отступ на длину ключа, с того места и будет осуществляться сокрытие сообщения.

Так как, используется зашумление, даже пропущенные слова будут подвержены изменениям, однако никакой информации о сообщении они не будут нести. При извлечении, получатель согласно ключу, будет знать, откуда начинать извлечение и сколько слов в сообщении.

Таким образом, методы сокрытия сообщения в текстовый файл содержит следующие шаги:

1) проверка соответствия длины сообщения и ключа;
2) шифрование сообщения с помощью ключа;
3) перевод зашифрованного сообщения в последовательность бит (каждая буква занимает 8 бит, то есть 1 байт);
4) перевод стеготекста в массив байт;
5) отступ на длину ключа;
6) кодирование зашифрованного сообщения;
7) перевод полученных байтов стего-текста обратно в строку.

Таким образом, получатель сообщения не сможет извлечь сообщение из стеготекста без определенного ключа. При стегоанализе текста, в котором сокрыто сообщение, невозможно точно определить место, где хранится скрываемая информация и какие буквы являются существенными, при наличии в тексте предложений на русском и английском языках задача для стороннего лица еще больше усложняется.

Вывод

В статье представлены результаты анализа 5 методов лингвистической стеганографии, а также построена их сравнительная таблица. Таблица позволяет сравнивать методы по сложности реализации, сложности дешифрования и сложности обнаружения сообщения. В результате анализа был выбран метод жаргонов для разработки авторского алгоритма сокрытия сообщения в текстовом файле, который обеспечивает защиту передачи данных. Авторский метод имеет ряд преимуществ:

- не используется большие источники или хранилища данных;
- быстрое выполнение внедрения и извлечения информации;
- низкая вероятность обнаружения передаваемого сообщения;
- алгоритм может быть использован для текста любого стиля;
- отсутствие возможности извлечения сообщения, в случае обнаружения.

Литература

1. Большаков И.А., А.Ф. Гельбух. Раздельное представление словосочетаний для существительных единствен-ного и множественного числа. // Труды Международного Семинара по вычислительной лингвистике и ее приложениям. Диалог'96. Пущино, 1996.– с. 42–44.

2. FrontLine International foundation for the protection of human rights defenders [Электронный ресурс] / L.Trappeniers [et al.] // Computer. – Электрон. дан. - 2013. - Vol.46, No 2. – P. 24-25. – Режим доступа: https://equalit.ie/esecman/russian/chapter2_7.html

3. Мельчук, И. А. Опыт теории лингвистических моделей «Смысл Û Текст». Семантика, синтаксис. М.: Наука. – 1974. – 314 с.

4. Стеганография [Электронный ресурс] //Академик, 2018: [сайт] – https://dic.academic.ru/dic.nsf/ruwiki – Загл. с экрана.

5. Электронная библиотека студента [Электронный ресурс.] // Библиофонд, 2018: [сайт] – URL: https://www.bibliofond.ru/view.aspx?id=657009 –

Сидорчук В.И., Губенко Н.Е., Сипаков Д.С. Метод сокрытия сообщений на основе жаргонов. *В статье было рассмотрено понятие стеганографии. Была поднята тема о защите информации от несанкционированного доступа. Был проведен сравнительный анализ методов сокрытия сообщения. На основе одного из методов, был разработан авторский метод сокрытия информации.*

Ключевые слова: *стеганография, лингвистика, защита данных, текст, сокрытие текста, стегосообщение*

Sidorchuk V.I., Gubenko N.E., Sipakov D.S. The method of hiding messages based on jargon. *The article examined the concept of steganography. The topic of protecting information from unauthorized access was raised. A comparative analysis of the methods of hiding the message was carried out. On the basis of one of the methods, the author's method of hiding information was developed.*

Key words: *steganography, linguistics, data protection, text, text hiding, stego message*

УДК 004.4

Обзор методов аудио стеганографии

Толбатова А.С., Чернышова А.В.
Донецкий национальный технический университет, г. Донецк
кафедра программной инженерии
anntolb0789@gmail.com, chernyshova.alla@rambler.ru

Толбатова А.С., Чернышова А.В. Обзор методов аудио стеганографии. *В статье рассматривается обзор методов цифровой аудио стеганографии, представлена структура wav и mp3 файлов, как возможных контейнеров. Исследованы возможности существующих программных средств для аудио стеганографии.*

Ключевые слова: *стеганография, аудио, wav, mp3, фреймы, теги, фазовое кодирование, шифрование, сокрытие информации.*

Введение

Сейчас практически невозможно представить нашу жизнь без интернета. Более 3 млрд. человек пользуются всемирной паутиной. Так как данная технология развивается очень быстро, возникает ряд проблем. Наиболее актуальной из этого ряда является обеспечение безопасности самой сети. Каждый день пользователи обмениваются данными, которые должны оставаться засекреченными. Интернет должен обеспечить защитой передаваемые данные и самого пользователя от всякого рода атак и угроз.

Важным фрагментом информационной безопасности является стеганография. Стеганография – наука о сокрытии передачи информации. Её задачей является скрытие самого факта передачи данных [1].

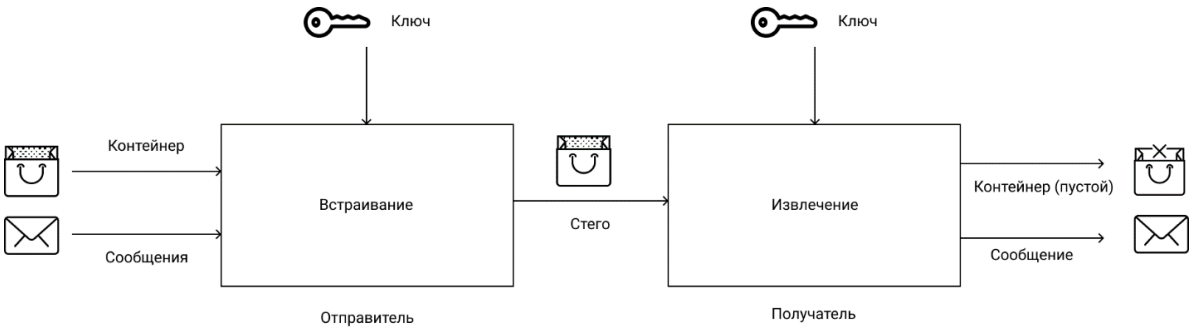


Рисунок 1 – Общая схема стеганографического процесса

В качестве контейнера могут использоваться файлы различных форматов. Цифровая аудио стеганография предполагает использование в качестве контейнера аудио файла. Рассмотрим форматы файлов WAV и MP3, как возможные форматы контейнеров для аудио стеганографии.

Структура WAV файла

WAV файл (WaveformAudioFileFormat) состоит из двух частей. В одной хранится заголовок файла, в другой - область данных. В заголовке файла содержится информация о размере файла, количестве каналов, частоте дискретизации и количестве бит в глубине звучания [2].

Область данных указывает размер звуковой информации и содержит необработанные звуковые данные.

Структура WAV файла представлена на рисунке 2.

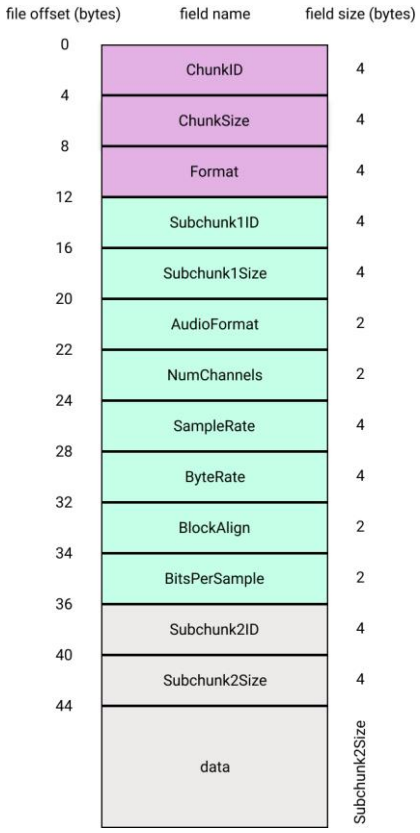


Рисунок 2 – Структура WAV файла [2]

Структура Mp3 файла

MP3 - формат файла для хранения аудиоинформации, который разработала команда MPEG. Согласно психоакустической модели, MP3 использует спектральные отсечения. Звуковой сигнал раскладывается на одинаковые по продолжительности отрезки, которые после обработки упаковываются в отдельные фреймы. В связи с тем, что каждый фрейм может иметь несколько контейнеров, можно хранить информацию о нескольких потоках. Степень сжатия можно изменять, в частности в пределах одного фрейма.

MP3-файлы могут создаваться как с высоким, так и с низким битрейтом [3]. Это влияет на качество конечного файла. Интервал возможных значений битрейта составляет 8-320 кбит/с.

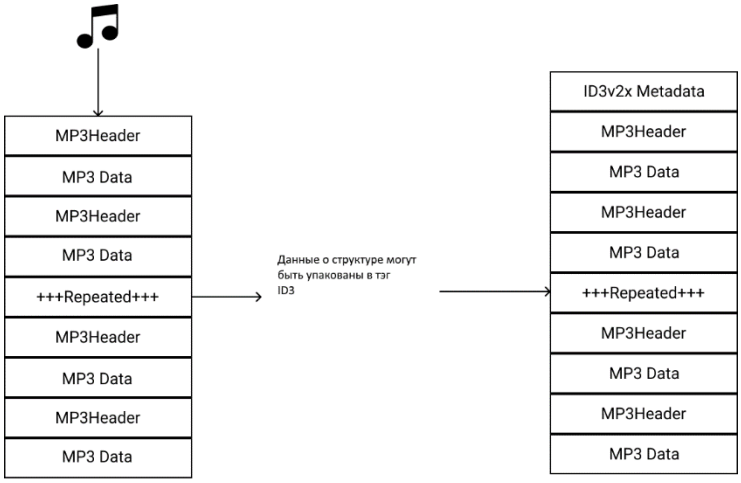


Рисунок 3 – Структура MP3 файла

Виды аудио стеганографии

Аудио стеганография – это вид стеганографии, использующий аудио файлы разных расширений в виде контейнеров для скрываемой информации.

Сейчас наиболее распространены 3 вида аудио стеганографии, которые используют различные типы сокрытия информации в файлах:

— сокрытие информации в mp3-фреймах;

— сокрытие информации в mp3-тегах;

— фазовая стеганография [4].

Рассмотрим известные способы сокрытия информации в аудио файлах.

1. Сокрытие информации в mp3-фреймах.

Существует метод, позволяющий преобразовывать файл с расширением wmv в файл с расширением Mp3 с добавлением скрытых данных.

Каждый фрейм файла формата MP3 содержит свой собственный заголовок. В заголовке фрейма находится информация о способе кодирования данного фрейма, использовании контрольных сумм заголовка, режиме стерео и остальная техническая информация. Также в заголовке находится область синхронизации, необходимая для идентификации фрейма в потоке байтов. Важной функциональной возможностью формата является то, что фреймы могут следовать друг за другом с промежутками. MP3 файл построен так, что позволяет скрывать информацию в служебных областях [5].

Смысл реализации заключается в том, что информация сначала шифруется, а затем в процессе кодирования MP3-файла (из WAV) добавляется в конечный результат. В итоге получается обычный MP3-файл без заметных для слуха искажений, но хранящий в себе закодированные данные.

2. Сокрытие информации в mp3-тегах.

ID3 – формат метаданных, который очень часто используется в звуковых файлах формата MP3. Есть две разные версии ID3-данных: ID3v1 и ID3v2. ID3v1 имеет размер 128 байт, которые дописываются в конец mp3-файла. Там хранится: название трека, исполнитель, альбом, год, комментарий, номер трека (для версии 1.1) и жанр.

Вскоре всем стало понятно, что 128 байт - это очень мало для хранения такой информации. В связи с этим, появилась усовершенствованная версия данных – ID3v2. Тегиv2 отличаются от первой версии тем, что они размещаются в начале файла и имеют переменную длину. Это позволяет поддерживать потоковое воспроизведение. Данные ID3v2 состоят из заголовка и последующих фреймов ID3v2 [6].

3.Фазовая стеганография.

В данном методе кодирования используется замена фазы исходного звукового сегмента на опорную фазу, представляющую собой секретную информацию. Фазовое кодирование является одним из наиболее эффективных методов кодирования. При значительном изменение фазового соотношения между каждой частотной составляющей, шумы становятся заметными. Если же изменение фазы будет незначительным, то изменения, которые внесены в аудио файл, будут незаметны для человеческого слуха [7].

Фазовое кодирование происходит следующим образом:

— разделение исходного звукового сигнала на мелкие сегменты так, чтобы их общая длина была равна длине сообщения;

— создание фазовой матрицы при помощи дискретного преобразования Фурье;

— вычисление разности фаз между соседними сегментами.

Из-за того, что фазовые сдвиги между двумя соседними сегментами могут быть легко обнаружены, в стегосигнале должны быть сохранены разности фаз. Поэтому секретное сообщение встраивается только в фазу первого сегмента:

$$Новаяфаза = \begin{cases} \frac{\pi}{2} \text{еслибитсообщения} = 0 \\ -\frac{\pi}{2} \text{еслибитсообщения} = 1 \end{cases} \quad (1)$$

Используя новую фазу первого сегмента создается новая матрица фаз и разница между ними;

Звуковой сигнал восстанавливается путем применения обратного дискретного преобразования Фурье с использованием новой матрицы и исходной матрицы величин, после чего звуковые сегменты сцепляются [7].

Получатель должен знать длину сегмента, чтобы извлечь секретное сообщение из звукового файла. После чего получатель с помощью дискретного преобразования Фурье может извлечь секретную информацию.

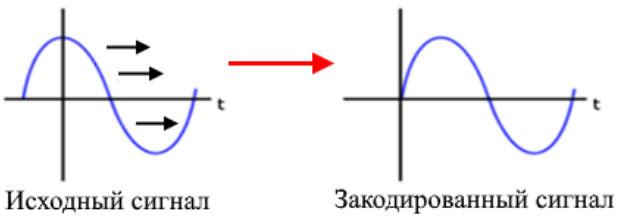


Рисунок 4 – Фазовое кодирование

Обзор программного обеспечения для аудио стеганографии

В 1999 году Флориан Хайденрик разработал программу Mp3tag. Это бесплатный редактор метаданных для большинства аудио форматов с закрытым кодом.

Mp3tag представляет собой довольно простой в использовании программный продукт, с помощью которого можно быстро редактировать теги в разных аудио файлах. Можно вносить изменения в ID3-теги, в комментарии MP3 и Ogg файлов, а также можно производить замену специальных символов или слов. Их всех возможностей программы можно отметить функции импорта/экспорта информации, поддержку Unicode и работу с онлайн-базой данных freedb, а также осуществление группового переименовывания файлов на основе информации в тегах.

Утилита поддерживает такие аудио форматы: ALAC, AAC, FLAC, APE, MP3, MP4, MPEG-4, MPC, OGG, OptimFROG OFR, OFS, SPX, TAK, TTA, WMA, WV [8].

Выводы

В рамках статьи был проведен анализ некоторых существующих методов цифровой аудио стеганографии, рассмотрены форматы некоторых аудио файлов, как возможных стеганографических контейнеров, выполнен обзор программного обеспечения для аудио стеганографии. В дальнейшем планируется спроектировать и реализовать программное обеспечение по реализации цифровых водяных знаков для аудио файлов.

Литература

1. Стеганография // Wikipedia. [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Стеганография
2. WAV // Wikipedia. [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/WAV
3. MP3 // Wikipedia. [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/MP3
4. ГолубевЕ.А., Емельянов Г.В. Стеганография как одно из направлений обеспечения информационной безопасности // Технологии информационного общества, Спецвыпуск T-Comm, 2009. —с. 185-186.
5. Прячем текст в MP3 // Хабрахабр. [Электронный ресурс]. – Режим доступа: https://habr.com/post/112914/
6. ID3 // Wikipedia. [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/ID3_(метаданные)
7. Грибунин В. Г., Оков И. Н., Туринцев И. В. Цифровая стеганография. - М.: Солон-Пресс, 2002.
8. Mp3tag // Mp3tag. [Электронный ресурс]. – Режим доступа: https://www.mp3tag.de/en/

Толбатова А.С., Чернышова А.В. Обзор методов аудио стеганографии. *В статье рассматривается обзор методов цифровой аудио стеганографии, представлена структура wav и mp3 файлов, как возможных контейнеров. Исследованы возможности существующих программных средств для аудио стеганографии.*

Ключевые слова: *стеганография, аудио, wav, mp3, фреймы, теги, фазовое кодирование, шифрование, сокрытие информации.*

Tolbatova A.S., Chernyshova A.V. Review of steganography audio methods. *The article reviews the methods of digital audio steganography, presents the structure of wav and mp3 files as possible containers. The possibilities of existing software for audio steganography are investigated.*

Keywords: *steganography, audio, wav, mp3, frames, tags, phase encoding, encryption, information hiding.*

УДК 004.021

Разработка ПО оптимизации логистики поставок продукции

Хубеджев Д.П., Ситникова О.Д.
Донецкий национальный технический университет
hubedjev.d.p@gmail.com, olga_d_s@mail.ru

Хубеджев Д.П., Ситникова О.Д. Формализация задачи логистики поставок и разработка ПО для формирования оптимальных маршрутов. *В статье представлен модифицированный метод Кларка-Райта для решения задач развозки. Модификации позволили реализовать данный метод для работы с более точными данными, полученными с помощью сервисов Google Maps.*

Ключевые слова: *логистика, задача развозки, метод Кларка-Райта, маршрут, Google Maps.*

Введение

Актуальность проблемы выражена в динамичном развитии розничной торговли, связанная с постоянно изменяющимися факторами внешней среды и обострением конкуренции. Товародвижение является достаточно сложной системой, динамически изменяющейся под влиянием внутренних и внешних факторов, имеющая диверсификационные взаимодействия. Таким образом, оптимизации и рационализации товародвижения не всегда достигают наивысших результатов, из-за сложности получения или переработки информации, способности к принятию решений. Преобразования логистических систем товародвижения розничных торговых предприятий в последние годы в значительной степени связаны с трансформацией отношений в контексте элементов логистической системы.

Целью работы является оптимизация алгоритма Кларка-Райта для более точных данных, полученных с помощью сервисов Google Maps.

Характеристика существующих программных средств

На сегодняшний день, существующие программные системы заточены под определенные особенности предприятий. При этом, программные продукты малоизвестны и в общем доступе их найти или невозможно, или крайне маловероятно. Однако продукты, что попали в сеть не являются Open Source проектами, а их функционал или не соответствуют современным платформам, или имеет ограниченный функционал.

Из этого следует, что существующие программные проекты не соответствуют критериям предполагаемого продукта, либо перенасыщены дополнительными функциями.

Особенности:

– узконаправленность;
– богатый функционал;
– специализированный интерфейс.

Достоинства:

– индивидуальность;
– возможность подключения дополнительных модулей;
– возможность подключения сторонних функциональных блоков;
– узкоспециализированность продукта.

Недостатки:

– закрытый программный код;
– коммерческий продукт;
– низкая производительность
– высокая стоимость;
– узкая специализация.

Задача развозки грузов, как оптимизация схем транспортировки

Задача развозки – это транспортная задача по доставке мелкопартионных грузов из распределительного центра, например, оптовой базы, склада, грузового терминала и пр., множеству получателей, расположенных в районе развозки. Данная задача включает в себя построение радиальных и кольцевых маршрутов для

минерализации финальной величины длинны. Когда объем, запрашиваемый получателем, больше или равен грузоподъемности автомобиля, используются радиальные маршруты,. По математической модели рассматриваемая задача является задачей нескольких коммивояжеров, а значит относится к классу NP-трудных.

Метод Кларка-Райта является одним из алгоритмов, решающих задачу развозки. В данном алгоритме вводится понятие выигрышей для оценки операций слияния маршрутов. Выигрыш являет собой меру сокращения стоимости, достигаемую благодаря комбинированию двух небольших маршрутов в один. Из этого следуют такие достоинства: простота, надежность и гибкость, при погрешности решения не превосходящей в среднем 5-10%.

Так же недостатком данного алгоритма является получение пути из одной точки в другую и обратно, как одинакового расстояния, что может не соответствовать действительности. Для решения данной проблемы в алгоритм были внесены модификации.

Описание модифицированного алгоритма Кларка-Райта

Суть метода заключается в пошаговом переходе к оптимальной схеме развозки с кольцевыми маршрутами. Для этого вводится понятие – километральный выигрыш.

На рисунке 1 показаны две схемы доставки.

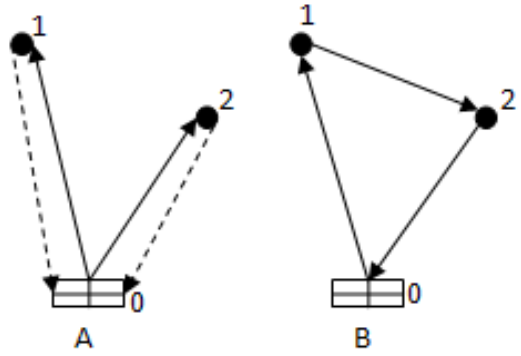


Рисунок 1 – Схемы доставки

На схеме доставке А отображается доставка грузов по радиальным маршрутам в пункты 1 и 2. В этом случае суммарный пробег автотранспорта равен:

$$L_A = d_{01} + d_{10} + d_{02} + d_{20} \quad (1)$$

Схема доставки В предполагает доставку грузов по кольцевому маршруту в пункты 1 и 2. Тогда пробег автотранспорта составляет:

$$L_B = d_{01} + d_{12} + d_{20} \quad (2)$$

Схема В, обычно дает лучший результат по пробегу, по сравнению со схемой А. При переходе к такой схеме рассчитывается километральный выигрыш:

$$S_{12} = L_A - L_B = d_{10} + d_{02} - d_{12} \quad (3)$$

В общем случае мы имеем такую формулу:

$$S_{ij} = d_{i0} + d_{0j} - d_{ij} \quad (4)$$

Где:

а) Sij – километральный выигрыш, км;

б) d0i, d0j – расстояние между пунктами i и j и складом соответственно, км;

в) dij – расстояние между пунктами i и j, км.

Описать основную часть алгоритма можно в 6 шагах:

Шаг 1 – находим ячейку (i*, j*) с максимальным километральным выигрышем S_{max} на матрице километральных выигрышей:

$$S_{max} = max_{i,j} S(i,j) = S(i^*, j^*) \quad (5)$$

При поиске максимального элемента нужно учесть ряд условий:

а) точки i* и j* не входят в состав одного и того же маршрута;

б) точка i* или j* является начальным или конечным пунктом, того маршрута, в который он войдет, а другая точка не используется в других маршрутах;

в) точки i* и j* является начальным и конечным пунктом, тех маршрутов, которые они объединяют;

г) ячейка (i*, j*) или (j*, i*) не использовались ранее.

Если удалось найти такую ячейку, которая удовлетворяет трем указанным условиям, то переход к следующему шагу. Если нет, то к шагу 4.

Если точки i* и j* не использовались в других маршрутах, то создать новый.

Шаг 2 – проверить выполнения условия грузоподъемности на протяжении всего маршрута:

$$Q + q \leq c \quad (6)$$

Где:
а) Q – общее количества груза требуемого маршрутом;

б) q – количество требуемого товара на добавляемую точку;

в) c – грузоподъемность транспорта.

Если условие выполняется, то переход к шагу 3, если нет – к шагу 1.

Шаг 3 – добавляем полученную точку в начало/конец маршрута и возвращаемся к шагу 1.
Шаг 4 – рассчитываем суммарное расстояние, которое необходимо пройти для покрытия всех маршрутов.

Оценка сложности алгоритма

За основу реализованного алгоритма был взят алгоритм Кларка-Райта и были внесены в него оптимизации позволяющие сделать вычисления с помощью более реальных данных. Оптимизации сделаны не увеличивая общей сложности алгоритма, хотя и увеличивает общее количество шагов, которые должен сделать алгоритм. В связи с этим в основу алгоритма Кларка-Райта были добавлены дополнительные проверки, не уменьшающие эффективность и уменьшающие количество проходов в цикле, что бы покрыть затраты времени на большее количество шагов.

Сложность алгоритма равна:

$$O(N) * \left(O(N^2) + O(N)\right) = O(N^3) + O(N^2) = O(N^3) \quad (7)$$

Такая сложность является более чем допустимой, так как задачи, имеющие такую сложность, могут обрабатывать довольно большие объёмы данных за допустимое время.

Данные о соответствии времени от размерности выборки отображены в таблице 1.

Таблица 1 – Форматирование страницы в статье

Размерность выборки	Время
100 заявок	До 1 секунды
200 заявок	7 секунд

300 заявок	34 секунд
500 заявок	4 минуты 22 секунды
1000 заявок	71 минута 24 секунды

Результаты тестирования подтверждают расчеты и соответствует выведенной формуле сложности (см. формулу 7).

Выводы

Был изучен и модифицирован метод Кларка-Райта для решения задач развозки. Модификации позволили реализовать данный метод для работы с более точными данными, полученные с помощью сервиса Googl Maps. Описан алгоритм с включенными в него модификациями и проведены тесты, показывающие, что с данными изменениями сложность алгоритма не увеличивается.

Используя данный алгоритм в реальной системе можно добиться уменьшения количества буферных складов и значительно уменьшить расходы на логистику. В перспективе для полноценной программной системы следует разработать сразу набор алгоритмов оптимизации логистики, проработав их взаимосвязь, для получения лучших результатов.

Литература

1. Гери, М. Вычислительные машины и труднорешаемые задачи / М. Гери, Д. Джонсон. – Москва: Мир, 1982 – 416 с.

2. Логистика [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikiversity.org/wiki/Логистика – Загл. с экрана.

3. Навигационная система [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/Навигационная_система – Загл. с экрана.

4. Метод Кларка-Райта. Оптимальное планирование маршрутов грузоперевозок [Электронный ресурс] // Инфостар – Режим доступа: https://infostart.ru/public/443585/ – Загл. с экрана.

5. Оценка сложности алгоритма [Электронный ресурс] // Хабр – Режим доступа: https://habr.com/post/104219/ – Загл. с экрана.

6. Руководство для разработчиков Google Maps Distans API [Электронный ресурс] // Google Maps APIs – Режим доступа: https://developers.google.com/maps/documentation/distance-matrix/intro?hl=ru#unit_systems – Загл. с экрана.

Хубеджев Д.П., Ситникова О.Д. Формализация задачи логистики поставок и разработка ПО для формирования оптимальных маршрутов. *В статье представлен модифицированный метод Кларка-Райта для решения задач развозки. Модификации позволили реализовать данный метод для работы с более точными данными, полученными с помощью сервисов Google Maps.*

Ключевые слова: *логистика, задача развозки, метод Кларка-Райта, маршрут, Google Maps.*

Khubedjev Danil, Sitnikova Olga Formalization of the task of supply logistics and software development for the formation of optimal routes. *The article presents a modified Clark-Wright method for solving transportation tasks. Modifications made it possible to implement this method to work with more accurate data obtained using Google Maps services.*

Key words: *logistics, transportation task, Clark-Wright method, route, Google Maps.*

УДК 004.94:004.89

Проектирование в среде MadKit агентно-ориентированной системы прогнозирования результатов обучения студентов

Янкивский А.А., Павлова Е.М., Федяев О.И.
Донецкий национальный технический университет
lionsasha@gmail.com, olegfedyayev@yahoo.com

Янкивский А.А., Павлова Е.М., Федяев О.И. Проектирование в среде MadKit агентно-ориентированной системы прогнозирования результатов обучения студентов. *В статье рассматривается построение имитационной модели прогнозирования остаточных знаний в зависимости от ментальности студентов. Имитационная модель представлена в виде сообщества искусственных агентов, которые реализуют роли преподавателей и студентов. Логический уровень многоагентной системы представлен совокупностью абстрактных моделей, построенных с помощью методологии Gaia. Физический уровень агентов описан на языке Java, в соответствии со средой MadKit. Объём передаваемых студенту знаний и степень их усвоения определяется агентом Преподаватель с нейросетевой архитектурой.*

Ключевые слова: *агентно-ориентированная система, среда MadKit, прогнозирование, знания и умения, модели Gaia, искусственный агент, нейронная сеть.*

Введение

В современных условиях возрастает потребность в анализе и управлении сложными социально-экономическими и производственными системами, состояние которых в большинстве случаев непредсказуемо и не может быть прогнозируемо изначально аналитически, т.к. оно является результатом динамического взаимодействия множества разнородных активных элементов системы и окружающей среды.

В работе объектом исследования выступает система профессиональной подготовки студентов на выпускающей кафедре университета. По причине неоднородности и интеллектуальности поведения субъектов (студентов и преподавателей) применять классические математические методы не представляется возможным. В этой связи целесообразно применять методы имитационного моделирования для анализа эффективности процесса обучения. Оценка качества обучения является актуальной задачей управления учебным заведением. Сложность её решения обусловлена тем, что образовательные процессы протекают очень медленно, т.е. система подготовки кадров как объект управления является инерционной, поэтому эффективность инновационных изменений можно оценить только через 4-6 лет (цикл обучения студента). Методом моделирования можно за короткое время получить прогноз целесообразности нововведений [1].

Прогнозирование результатов обучения позволит анализировать качество обучения, видеть степень усвоения студентами учебного материала, обнаруживать расхождение по компетенциям между дисциплиной и требованиями фирм, оценивать возможность трудоустройства выпускников.

Анализ литературы показывает, что решение перечисленных задач осуществляется, как правило, не формальными методами, это снижает их практическую ценность. Современный уровень информационных технологий позволяет разрабатывать качественно новые модели, объединяющие достоинства математических методов, статистики, теории нейронных сетей, программирования. С появлением теории многоагентных систем появилась возможность создавать модели сложных распределённых и неоднородных систем, к классу которых относится исследуемый объект.

Данная работа посвящена разработке с помощью многоагентного подхода имитационной модели, которая ориентирована на прогнозирование качества обучения студентов по отдельным дисциплинам. Для этого искусственным агентам имитационной модели необходимо делегировать полномочия субъектов образовательного процесса, т.е. функциональные обязанности студентов и преподавателей. С этой целью в работе было рассмотрено решение следующих задач:

- формализация прогнозирования уровня компетенций студентов после завершения курса обучения;
- составление логических моделей, описывающих полномочия субъектов образовательной деятельности и их взаимодействие;
- реализация способности преподавателя передавать знания студентам с учётом их ментальности;
- программная реализация сообщества искусственных агентов с нейросетевой архитектурой на языке Java в инструментальной среде MadKit.

Постановка задачи прогнозирования уровня остаточных знаний студентов по профессиональным дисциплинам

Рассматривается задача прогнозирования качества профессионального обучения студентов в зависимости от их личностных характеристик и других факторов. Она решается на основе применения интеллектуальных агентов, использующих искусственные нейронные сети. Для этого в архитектуру агентов преподавателей вводятся соответствующие искусственные нейросетевые модели, способные функционально описать зависимость получаемых студентом профессиональных знаний и умений по одной дисциплине от факторов, влияющих на полноту этих знаний. В свою очередь задача прогноза разбивается на две подзадачи. [1]

Подзадача 1. Настройка модели по данным наблюдений. Это обратная задача, связанная с нахождением параметров модели, т.е. с построением функции f по наблюдаемым данным ментальности студента, ментальности преподавателя, среды обучения, а также знаниям и умениям студента по одной изучаемой дисциплине.

$$Pc = f\,(Mc, Mп, C),$$

где M_c – ментальность студента; $M_п$ – ментальность преподавателя; C – среда обучения; P_c – профессионализм студента по одной изучаемой дисциплине.

Подзадача 2. Формирование знаний и умений по ментальности участников образовательного процесса. Данная задача состоит в явном нахождении профессионализма студента (Pc), т. е. его остаточных знаний и умений, после изучения конкретной дисциплины, по замеренным данным о ментальности студента (Mc) и преподавателя (Mп) с помощью построенной модели f :

$$Pc = f\,(Mc, Mп, C).$$

Агентно-ориентированная модель обучения студентов ВУЗа

Объектом анализа выступает система обучения студентов на кафедре, которая является неоднородной и распределённой. Для моделирования таких систем целесообразно применять агентно-ориентированный подход. Агентно-ориентированный анализ объекта, как правило, выполняется по методологии Gaia [2].

Она поддерживает два уровня разработки мультиагентных систем: микроуровень (разработка отдельного агента) и макроуровень (разработка агентства). Также она имеет существенное ограничение: структура каждого агента во время его работы должна оставаться неизменной.

Методология Gaia состоит из двух больших этапов: анализа и проектирования. На этапе анализа реализуются модели ролей и модели взаимодействия, на этапе проектирования – модель агентов (за основу служат модели ролей), модели услуг и модель связей (они формируются с учётом моделей этапа анализа)

Для детализации моделей выделены абстрактные и конкретные понятия поставленной задачи. К абстрактным понятиям отнесены роли, полномочия, обязательства, протоколы, активность, обязательства жизнеспособности, условия безопасности. Эти понятия используются во время анализа для концептуализации системы. Конкретные понятия (например, агентные типы, услуги, связи между агентами) используются на этапе проектирования и имеют непосредственный программный аналог в системе.

В моделе ролей Роль рассматривается как абстрактное описание ожидаемых функций должностного лица. Роли характеризуются полномочиями и обязательствами. Полномочия устанавливают ресурсы, которые могут обоснованно использоваться для выполнения роли, возможность роли генерировать информацию, только чтение информации или возможность ее изменять. Полномочия роли специфицируются ключевыми словами read (доступ к информации без возможности ее изменять), changes (возможность изменять информацию) и generates (возможность генерировать информацию).

Функции роли описываются в виде обязательств, которые делятся на две категории: жизнеспособность и условия безопасности. Обязательства жизнеспособности показывают, что будет сделано агентом и определяются выражениями жизнеспособности, которые определяют "жизненный цикл роли". Выражения жизнеспособности являются регулярными выражениями. Выражения жизнеспособности устанавливают возможность выполнения траектории "жизненного цикла" посредством деятельности и взаимодействий, связанных с ролью. Деятельность – это то, что агент может выполнить, не прибегая к взаимодействию с другими агентами, а взаимодействия с другими ролями определяются протоколами.

Условия безопасности в методологии Gaia специфицируются посредством предикатов. В предикатах используются переменные, обозначающие атрибуты полномочий роли. Схема роли Студент представлена на рисунке 1.

Схема роли Студент

Описание роли

Роль **Студент** предназначена для получения остаточных знаний и умений по каждой дисциплине, изучаемой в текущем семестре.

Активность

Сформировать ментальный портрет
Выслать ментальный портрет
Выбрать дисциплину для изучения
Сохранить усвоенные компетенции

Протоколы

Приём результатов тестирования личности
Запись к лектору на дисциплину
Передача ментальности студента лектору
Приём усвоенных студентом компетенций

Полномочия

reads	данные студента
reads	ментальный портрет студента
reads	список дисциплин в семестре
changes	название изучаемой дисциплины
generates	остаточные знания и умения по дисциплине

Обязательства жизнеспособности

СТУДЕНТ = Формирование личностного портрета студента . Изучение дисциплин в семестре

Формирование личностного портрета студента = ПРИЁМ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ ЛИЧНОСТИ . СФОРМИРОВАТЬ МЕНТАЛЬНЫЙ ПОРТРЕТ

Изучение дисциплин в семестре = (ВЫБРАТЬ ДИСЦИПЛИНУ ДЛЯ ИЗУЧЕНИЯ . ЗАПИСЬ К ЛЕКТОРУ НА ДИСЦИПЛИНУ | ПЕРЕДАЧА МЕНТАЛЬНОСТИ СТУДЕНТА ЛЕКТОРУ | ПРИЁМ УСВОЕННЫХ СТУДЕНТОМ КОМПЕТЕНЦИЙ . СОХРАНИТЬ УСВОЕННЫЕ КОМПЕТЕНЦИИ) +

условия безопасности

- результаты тестирования личности полны и непротиворечивы

Рисунок 1 – Схема роли Студент

В модели взаимодействия динамика МАС определяется взаимодействием между ролями, поэтому на следующем этапе анализа рассматривались основные виды и назначения взаимодействий. Совокупность взаимодействий между ролями образуют модель взаимодействий, которая состоит из набора установленных протоколов. Для каждого вида взаимодействия составляется отдельный протокол. На рисунке 2 представлены примеры протоколов взаимодействий ролей Студент и Лектор.

Рисунок 2 - Примеры протоколов ролей: а) запись на учебную дисциплину; б) запрос ментального портрета студента

К моделям нижнего уровня относятся агентная модель, модель услуг и модель связей агентов.

Модель агентов описывает используемые в системе агентные типы. Модель агентов системы представляется деревом агентных типов, в котором концевые вершины соответствуют ролям, определенным в модели ролей, а другие вершины соответствуют агентным типам. Модель агентов представлена на рисунке 3.

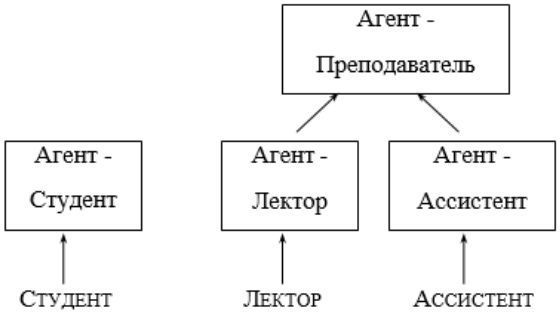


Рисунок 3 – Модель агентов

Модель услуг определяет функции, которые агент должен выполнять в соответствии с обязательствами жизнеспособности роли. Услуги можно сравнить с методами в объектно-ориентированном программировании, однако услуги не доступны для вызова другими агентами в отличии от методов объекта.

Для каждой услуги, выполняемой агентом, в модели услуг определяются входные и выходные данные, пред и пост условия, в зависимости от соблюдения которых агент инициирует выполнение услуги или услуга считается выполненной соответственно. Входные и выходные данные получают из активностей и протоколов взаимодействия роли. Модель услуг для исследуемой предметной области представлена на рисунке 4.

Услуга	Входные данные	Выходные данные	Предусловия	Постусловия
Сформировать ментальный портрет	Данные студента	Ментальный портрет студента	Данные полны	
Выслать ментальный портрет	Ментальный портрет студента	Фамилия студента, Ментальный портрет студента	Метальный портрет для студента получен	сообщение = отправлено
Выбрать дисциплину для изучения	Список дисциплин в семестре	Выбранная дисциплина	Не все дисциплины изучены	
Сохранить усвоенные компетенции	Остаточные знания и умения по дисциплине	Признак успешного сохранения		Признак сохранения сформирован

Рисунок 4 – Модель услуг

Модель связей отражает возможные коммуникативные связи между агентами и представляется в виде ориентированного или неориентированного графа. Узлы соответствуют типам агентов, а дуги – коммуникационным путям. Каналы коммуникации между агентами предполагают передачу сообщений между агентами в обоих направлениях. На рисунке 5 отражена связь агентов.

Рисунок 5 – Модель связей агентов

Нейросетевые средства реализации интеллектуальности агента Лектор

Процесс обучения студентов заключается в передаче знаний и навыков от преподавателей к студентам.

Качество обучения фиксируется в экзаменационной ведомости. Разрабатываемая модель процесса обучения должна формировать на выходе остаточные знания студентов по отдельной дисциплине.

Прогноз остаточных знаний по одной конкретно взятой дисциплине для одного студента осуществляется в два этапа. На первом этапе прогнозируется экзаменационная оценка, а на втором этапе, исходя из прогнозируемой оценки, формируется усреднённый набор остаточных знаний и умений, соответствующий данной оценке. Первая нейронная сеть обучается на основании ментальных портретов студентов и экзаменационной ведомости. Вторая нейронная сеть – на основании критериев оценки и учебной программы дисциплины, в которой содержится перечень знаний и умений. Схема такой двухкаскадной модели представлена на рисунке 6.

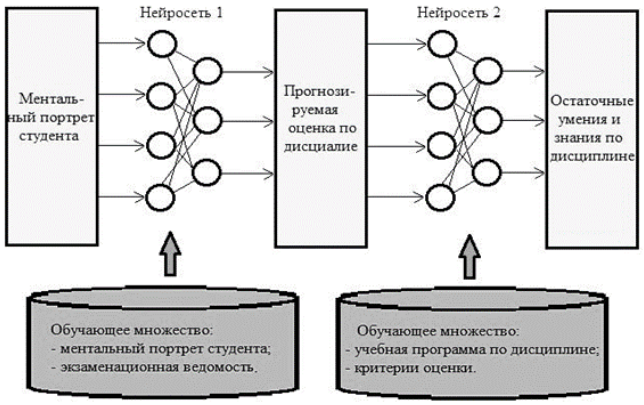


Рисунок 6 — Схема нейромодели, описывающая результаты обучения студента на примере одной дисциплины

Выходные сигналы второй нейросети образуют вектор, компоненты которого определяют уровень соответствующего остаточного знания или умения. Размер вектора определяется суммарным количеством знаний и умений, предусмотренных учебной программой дисциплины. Они обозначены вектором Y=(y1, y2, ..., yn), где n – количество знаний и умений; yi принадлежит интервалу [0,1].

Обучающее множество для второй нейросети составляет преподаватель-профессионал (эксперт) по своей дисциплине, используя утверждённые критерии оценки и учебную программу дисциплины, которая содержит перечень знаний и умений [2].

Спрогнозированная оценка с выхода первой нейросети подавалась на вход второй нейросети, которая формировала результирующий вектор Y остаточных знаний и умений этого студента.

Значения компонент вектора Y можно трактовать как степени уверенности в том, что у данного студента сохраняются в его памяти соответствующие знания и умения (конечно, относительно используемых обучающих множеств) [3].

Описание структуры программных агентов в среде MadKit

Инструментальная система MadKit, которая использовалась при разработке — модульная и масштабируемая мультиагентная платформа, написанная на Java. Она позволяет создавать программных агентов на разных языках: Java, Python, Jess, Scheme, BeanSchell [4]. Наличие визуальных моделей агентов повышает удобство их проектирования.

Архитектура MadKit включает в себя графический хост приложения, прикладных и системных агентов, а также микроядро системы. Компонентами микроядра системы является менеджер групп/ролей, синхронизирующий механизм и механизм обмена сообщениями.

Структура агентов в многоагентной системе прогнозирования состоит из 4 обязательных разделов:

− раздел активации (activate section), содержащий некоторый программный код, который будет выполняться непосредственно после создания агента (см. рис.7а);

− графический раздел (initGUI section), содержащий описание компонента Java, который должен использоваться в качестве графического интерфейса агента, и предназначен для замены графического интерфейса по умолчанию;

− раздел «жизнь» (live section), который содержит основной программный код, описывающий поведение агента (обычно, этот раздел содержит бесконечный цикл) (см. рис.7б);

− раздел «завершения» (end section), содержащий некоторый код, который выполняется, когда агент уничтожается.

```
//Раздел активации
@Override
protected void activate()
{
    //Получение роли
    createGroupIfAbsent(Department.COMMUNITY, Department.STUDENT_GROUP, true, null);
    requestRole(Department.COMMUNITY, Department.STUDENT_GROUP, Department.STUDENT_ROLE,
null);

    System.err.println(getOrganizationSnapShot(true));

    //Заполнение ментальности
    try {
      getMentality();
       }
    catch (IOException ex)
        {
      Logger.getLogger(AgentStudent.class.getName()).log(Level.SEVERE, null, ex);
       }

    getLogger().info(() -> "Я студент " + name);

    //Ожидание нужного преподавателя
    int pause = 2000 + (int) (Math.random() * 1000);
    getLogger().info(() -> "Студент регистрируется у преподавателя " + subject  + pause / 1000 + " секунд !");
    pause(pause);
  }
```

```
//Раздел жизни
@Override
protected void live()
{
    boolean isStudying = false;
    while (!isStudying)
        {
                Message profAnswer = null;
                while (profAnswer == null)
                {
                        profAnswer = sendMessageAndWaitForReply(
                         Department.COMMUNITY,
                         Department.STUDENT_GROUP,
                        subject,
                        new StringMessage(subject));

                        getLogger().info(() -> "Нет регистрации");
                }

      //Профессор найден
      if (getStringContent(profAnswer).equals("ok"))
       {
        getLogger().info(() -> "Запись на курс подтверждена");
        pause(3000 * 5);
        logFindProfessor(profAnswer);
        isStudying = listenCourse(profAnswer);
       }
    }
  }
```

а) б)

Рисунок 7 – Программный код агента Студент: а) для раздела активации; б) для раздела жизни

MadKit не налагает на архитектуру агентов никаких ограничений для достижения максимальной универсальности приложений.

Взаимодействие агентов реализовано с помощью асинхронной передачи сообщений. В имитационной модели агент может отправить сообщение другому агенту, определяемому его адресом или с помощью широковещательного сообщения, которое передают агентам, играющим данную роль в определённой группе.

MadKit обеспечивает несколько видов предопределенных сообщений, таких как StringMessage, XMLMesssage и ActMessage. С помощью последнего вида сообщения можно определить следующие подвиды сообщений: ACLMessages и KQMLMessages. Также есть возможность определить свой собственный класс сообщения, который будет наследоваться от класса сообщения по умолчанию. Эти возможности будут использованы в следующей версии имитационной модели.

У каждого агента есть свой «почтовый ящик», в который доставляются сообщения и который проверяется агентом для анализа сообщений [5].

Заключение

В статье рассмотрены вопросы проектирования имитационной модели прогнозирования остаточных знаний и умений в зависимости от ментальности студента. Имитационная модель представлена в виде сообщества искусственных агентов, которые в соответствии с принципом ограниченной рациональности

реализуют роли преподавателей и студентов. Приведен логический и физический уровень сообщества агентов. Логический уровень системы описан совокупностью моделей по методологии Gaia. Структура программных агентов на физическом уровне описана на языке Java и соответствует требованиям инструментальной системе MadKit. Предложена структура двухкаскадной нейронной сети для прогнозирования остаточных знаний и умений студентов по изучаемой дисциплине.

Литература

1. Лукина Ю.Ю., Федяев О.И. Технология создания мультиагентных систем в инструментальной среде MadKit.
2. Методология проектирования GAIA [Электронный ресурс] – Режим доступа: https://studopedia.ru/9_168381_metodologiya-proektirovaniya-Gaia.html. – Загл. с экрана.
3. Федяев О.И., Грабчук О.П. Формирование зависимости остаточных знаний студентов от их ментальности с помощью нейронных сетей // О.И. Федяев, О.П. Грабчук. – Интеллектуальный анализ информации ИАИ-2015- Донецк, ГОУ ВПО «Донецкий национальный технический университет», 2012.
4. MadKit Development Giude [Электронный ресурс] - Режим доступа: http://www.madkit.net/documentation/devguide/devguide.html. – Загл. с экрана.
5. Платформы для разработки МАС [Электронный ресурс] - Режим доступа: https://www.intuit.ru/studies/courses/13858/1255/lecture/23977. – Загл. с экрана.

Янкивский А.А., Павлова Е.М., Федяев О.И. Проектирование в среде MadKit агентно-ориентированной системы прогнозирования результатов обучения студентов. *В статье рассматривается построение имитационной модели прогнозирования остаточных знаний в зависимости от ментальности студентов. Имитационная модель представлена в виде сообщества искусственных агентов, которые реализуют роли преподавателей и студентов. Логический уровень многоагентной системы представлен совокупностью абстрактных моделей, построенных с помощью методологии Gaia. Физический уровень агентов описан на языке Java, в соответствии со средой MadKit. Объём передаваемых студенту знаний и степень их усвоения определяется агентом Преподаватель с нейросетевой архитектурой.*

Ключевые слова: *агентно-ориентированная система, среда MadKit, прогнозирование, знания и умения, модели Gaia, искусственный агент, нейронная сеть.*

Yankivsky Alexander, Pavlova Catherine, Fedyaev Oleg. Designing MadKit in an agent-based system for predicting student learning outcomes. *The article discusses the construction of a simulation model for predicting residual knowledge depending on the mentality of students. The simulation model is presented in the form of a community of artificial agents that implement the roles of teachers and students. The logical level of a multi-agent system is represented by a set of abstract models built using the Gaia methodology. The physical level of agents is described in Java, in accordance with the MadKit environment. The volume of knowledge transferred to the student and the degree of their learning is determined by the agent Teacher with neural network architecture.*

Key words: *agent-oriented system, MadKit environment, forecasting, knowledge and skills, Gaia models, artificial agent, neural network.*

СЕКЦИЯ «СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ»

УДК 004.94

Анализ функциональных возможностей и перспектива развития САПР кораблей

Бондаренко Е.С., Григорьев А.В.
Донецкий национальный технический университет
katusha.bondarencko@yandex.ua, grigorievalvl@gmail.com

Бондаренко Е.С., Григорьев А.В. Анализ функциональных возможностей и перспектива развития САПР кораблей. *В статье описаны основные этапы проектирования современных кораблей. Выделены стандартные требования, предъявляемые ко всем судам. Представлены популярные программные средства для проектирования и выделены их основные недостатки и преимущества. Проанализированы основные перспективы развития САПР кораблей и их функциональные возможности в настоящее время.*

Ключевые слова: *статья, САПР кораблей, проектирование, корабль, деталь, изделие, развитие.*

Введение

Цель: исследование функциональных возможностей САПР кораблей и перспективы их развития.

Актуальность работы: в настоящее время стремительно развиваются торгово-рыночные отношения между странами, как соседними, так и находящимися на разных континентах. Экономика государств не стоит на месте, развиваются экономические отношения между разными государствами. В связи с этим возрастает необходимость развития и совершенствования путей сообщения по всему миру.

Одним из наиболее безопасных, надежных и вместительных средств транспортировки является корабль. Еще несколько столетий назад им пользовались народы, проживающие вблизи морей и океанов. Сейчас этот транспорт не утратил своей актуальности, однако, технические характеристики многих кораблей давно устарели в связи со стремительным развитием техники в современном мире. Многие корабли требуют усовершенствования, а некоторые типы судна еще, возможно, не изобретены. В связи с этим возникает необходимость поиска оптимальной модели корабля для создания новых и усовершенствованных кораблей.

Такая возможность представляется в САПР кораблей. В этой среде суда можно создавать относительно быстро и с учетом всех деталей и характеристик. Такое судно будет прочным и надежным. Создание кораблей в САПР позволит вывести кораблестроение на новый уровень.

Корабль как объект проектирования.

Корабль — крупное морское судно, в том числе [1]:

1.Боевая и организационная единица Вооружённых Сил государства, входящая в состав военно-морского флота Вооружённых Сил государства, способна решать определённые боевые или специальные задачи в мирное и военное время; имеет государственный флаг и вооружение.

2.Многомачтовое парусное судно с прямыми парусами.

Представление о геометрической форме корпуса судна можно получить, изобразив его в трех проекциях соответственно трем основным взаимно-перпендикулярным плоскостям.

Существует много разновидностей кораблей. В первую очередь, все суда можно разделить на гражданские и военные. Их, в свою очередь, можно классифицировать по разным характеристикам. Однако выделим виды гражданских кораблей по назначению:

- грузовые суда;
- промысловые суда;
- служебно-вспомогательные суда;
- суда технического флота;

- научно-исследовательские суда;
- пассажирские суда.

Рисунок 1 – Грузовое судно

Рисунок 2 – Промысловое судно

Рисунок 3 - Служебно-вспомогательное судно

Рисунок 4 – Техническое судно

Рисунок 5 - Научно-исследовательское судно

Рисунок 6 – Пассажирское судно

Одной из определяющих характеристик кораблей является их размер. Для каждого типа судна он свой. Существует классификация судов по размерам, как общая, так и по каждому типу судна отдельно.

Интересно отметить, что самыми большими известными построенными судами в мире являются танкеры для добычи нефти и природного газа. Prelude в настоящее время – первое крупнейшее плавучее сооружение в мире. Супертанкер Seawise Giant известен как самый длинный корабль в истории. Его длина была соизмерима лишь с самыми высокими зданиями в мире. Судно Allure of the Seas недавно являлось крупнейшим пассажирским судном. Максимальное число пассажиров, какое этот лайнер способен нести, составляет 6296 человек, а экипажа – 2384.

Определившись с типом и размером судна, создается изображение судна и чертежи его проекций.

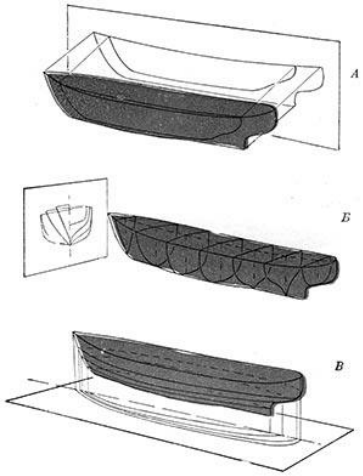


Рисунок 7 – изображение судна в проекциях: А — Профиль корпуса в диаметральной плоскости. Б — Проекция корпуса в плоскости миделъшпангоута. На левой половине чертежа — носовая часть корпуса, а на правой — кормовая. В— Плоскость ватерлинии как бы делит корпус судна на две части — подводную и надводную.

Такой вид имеет корпус современного судна. Геометрия подводной части имеет огромное значение для ходовых качеств судна. Изображение корпуса во всех трех плоскостях совместно образует теоретический

чертеж судна.

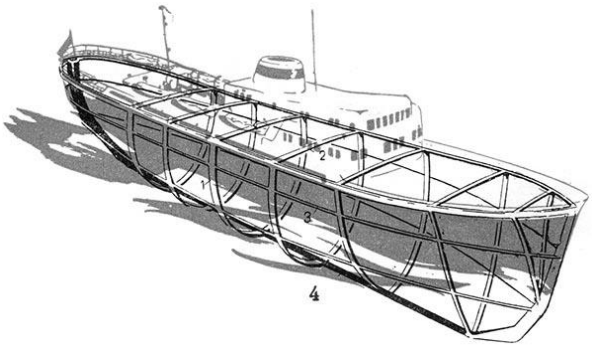


Рисунок 8 — Изображение основных частей набора корпуса всякого судна—его скелет: 1—шпангоуты; 2—бимсы; 3—стрингера; 4—киль

Многие современные суда собираются на конвейере из крупных блоков. Каждый блок — уже полностью собранная часть корпуса будущего судна.

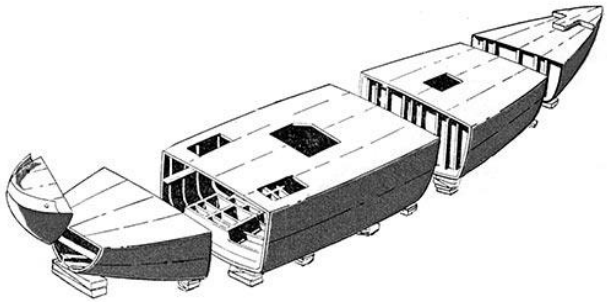


Рисунок 9 — Части корпуса судна

Основные этапы проектирования корабля

В период конструкторской подготовки производства разрабатывают проект судна. Разработка проекта корабля обычно проводится в четыре этапа [3]:

1. Техническое предложение

Техническое предложение содержит: схемы общего расположения судна, мидель-шпангоута, расположения механизмов в машинном отделении, расположения специальных устройств и эксплуатационно-экономический расчет.

2. Эскизный проект

Эскизный проект содержит чертежи общего расположения судна, теоретический чертеж, конструктивный мидель-шпангоут, расчеты весовой нагрузки.

3. Технический проект

Технический проект содержит: договорную документацию (чертежи общего расположения судна, спецификации по общесудовой, корпусной и механической части, системам и электрооборудованию), проектную документацию (чертежи по общесудовой и корпусной части), чертежи по механической части (установка главных двигателей и валопровода и схемы трубопроводов), чертежи общесудовых систем, чертежи электрооборудования, расчеты прочности по теории корабля, весовой нагрузки.

4. Рабочий проект

Рабочий проект содержит: рабочие чертежи и всю технологическую документацию, объем которой устанавливает предприятие-судостроитель в зависимости от степени подготовленности производства, от типа и размерений судна, размера серии и прочих данных. В состав рабочего проекта входят разрабатываемые вновь, а также типовые, обезличенные и нормализованные чертежи.

При конструкторской подготовке производства осуществляется унификация оборудования и материалов, решаются вопросы о технологичности конструкций и ремонтопригодности судна, обосновывается метод его постройки и производится разбивка корпуса на секции. Существует несколько способов сборки корпуса судна на стапеле: подетальный, секционный и блочный.

Требования к кораблю

К прочности судна предъявляются высокие требования. Набор судна, его обшивка, различные дополнительные узлы корпуса — вся конструкция целиком должна обладать необходимой жесткостью. Поэтому на больших судах шпангоуты, бимсы, стрингера изготавливают из стальных балок, а затем

обшивают их листовой сталью. Над килем обычно настилают водонепроницаемое второе дно. На бимсах располагают палубы, их может быть несколько. Верхняя палуба также делается водонепроницаемой.

Среди всех требований, предъявляемых к любому кораблю, можно выделить основные, которые являются определяющими. Рассмотрим функциональные и специальные требования [4].

Функциональные требования:

- грузоподъемность и пассажировместимость
- тип и назначение судна
- скорость хода
- тип энергетической установки
- район и дальность плавания
- автономность по запасам
- эксплуатационные ограничения
- класс классификационного общества
- численность и состав экипажа
- требования к технике безопасности
- прочность
- материалы для изготовления судна

Специальные требования:

- ходовые качества,
- судовые устройства и системы,
- навигационное оборудование,
- экологическая безопасность

Анализ функциональных возможностей САПР кораблей

Наиболее известными САПР кораблей в настоящее время являются K3-Ship, Foran, и Aveva Marine.

K3-Ship — это комплекс компьютерных программ для производства кораблей. Его преимущество по сравнению с остальными САПР – удобный и интуитивно-понятный интерфейс, легкость в освоении.

FORAN имеет преимущество среди CAD/CAM/CAE технологий в судостроении, в интеграции решений для полного проектирования судна, включая определение формы, определение структуры корпуса, насыщения, электрики.

Aveva Marine - комплекс программ для проектирования и постройки судов, в составе имеющий технологии для управления жизненным циклом судна в целом. Преимущество Aveva Marine состоит в том, что в ней возможна интеграция и синхронизация с другими системами, экспорт и импорт информации из них происходит быстро и без каких-либо осложнений [5].

Однако все эти САПР имеют недостатки: слабая расчетная база, создание разных типов судов по одному шаблону, отсутствие заготовленных моделей для дальнейшего проектирования.

Основными задачами совершенствования современных систем проектирования являются:

- создание БД кораблей по типам и видам;
- возможность выбора типа и вида судна из общей БД со своими особенными техническими характеристиками;
- применение современных более точных прямых расчетных методик на ранних стадиях проектирования.

Одной из перспективных отраслей развития САПР судостроения является 3D-моделирование. Судостроение – отрасль, которая считается консервативной, способна развиваться в мире 3D. Трехмерное сканирование и печать обладают уникальным потенциалом, так как позволяют воспроизводить сложнейшие пространственные формы и объекты.

В последние годы САПР кораблей значительно усовершенствовались, однако они все еще имеют большие недостатки. САПР должна быть масштабируемой – иметь возможность создавать как самые простые и малозатратные корабли, так и суда с большими техническими возможностями.

Выводы

В статье приведены классификации современных кораблей и основные этапы их проектирования. Выделены стандартные требования, предъявляемые ко всем судам. Рассмотрены САПР кораблей, их преимущества и недостатки, функциональные возможности и проанализированы их перспективы развития.

Литература

1. Белавин Н. И. Экранопланы. Л., «Судостроение», 1968.
2. Войтницкий Я. П., Першиц Р. Я., Титов И. А. Справочник по теории корабля. М., Судпромгиз, 1960.
3. Проектирование судов. В.В. Ашик, Ленинград 1975 г. 7670
4. Проектирование конструкций судового корпуса и основы прочности судов. В.Н. Лазарев, Н.В.

Юношева, Ленинград 1989 г. 8115
5. Минченко Л. В., Кандратова Т. А. Системы автоматического проектирования в судостроении [Текст] // Современные тенденции технических наук: материалы V Междунар. науч. конф. (г. Казань, май 2017 г.). — Казань: Бук, 2017. — С. 73-76. — URL https://moluch.ru/conf/tech/archive/230/12335/ (дата обращения: 06.11.2018).

Бондаренко Е.С., Григорьев А.В. Анализ функциональных возможностей и перспектива развития САПР кораблей*. В статье описаны основные этапы проектирования современных кораблей. Выделены стандартные требования, предъявляемые ко всем судам. Представлены популярные программные средства для проектирования и выделены их основные недостатки и преимущества. Проанализированы основные перспективы развития САПР кораблей и их функциональные возможности в настоящее время.*

Ключевые слова: *статья, САПР кораблей, проектирование, корабль, деталь, изделие, развитие.*

Bondarenko E.S., Grigoryev A.V. Analysis of the functional capabilities and the future development of spacecraft CAD. *The article describes the main stages of the design of modern ships. The standard requirements for all ships are highlighted. Presents popular software tools for design and highlights their main advantages and disadvantages. Analyzed the main prospects for the development of spacecraft CAD and their functional capabilities at the present time.*
Key words: *article, ship CAD, design, ship, detail, product, development.*

УДК 004.356.2

Обзор современных технологий, принципов и процессов 3D-печати

Горбань В.В., Григорьев А.В.
Донецкий национальный технический университет

Горбань В.В., Григорьев А.В. Обзор современных технологий и языков программирования для 3D-принтеров. *В статье дан обзор основ 3D печати, рассмотрены преимущества и недостатки, рассмотрен процесс создания изделия на 3D принтере. Актуальность данной темы обусловлена все возрастающей популярностью и распространенностью технологий 3D-печати.*

Ключевые слова: *3D-печать, 3D-принтер, 3D-модель, слой, сопло, изделие, наплавление, стереолитография, лазерное спекание, g-code.*

Введение

D печать набирает свои обороты каждый день. Уже сейчас 3D-печать заняла свое место в различных отраслях, от автомобильной промышленности до домашнего использования. На 3D-принтерах печатаются шедевры кулинарии, воссоздаются детали автомобилей, производство которых было прекращено десятки лет назад, и даже создаются живые ткани. По статистике, рынок 3D-принтеров с 2015 года вырос по размерам в 2,5 раза по сравнению с 2015 годом, с 5.2 до 12,8 млрд. [1]. В связи с этим, растет уровень интереса к этому стеку технологий.

Этот доклад имеет цель рассмотреть процесс 3D-печати, технологии создания и печати изделий на 3D принтерах, а также изучить известные достоинства и ограничения 3D-печати.

Процесс 3D-печати

D печать является разновидностью аддитивного производства [2] и обычно относится к инструментам быстрого прототипирования. 3D принтер представляет собой станок с числовым программным управлением (ЧПУ), использующий метод послойного создания детали.

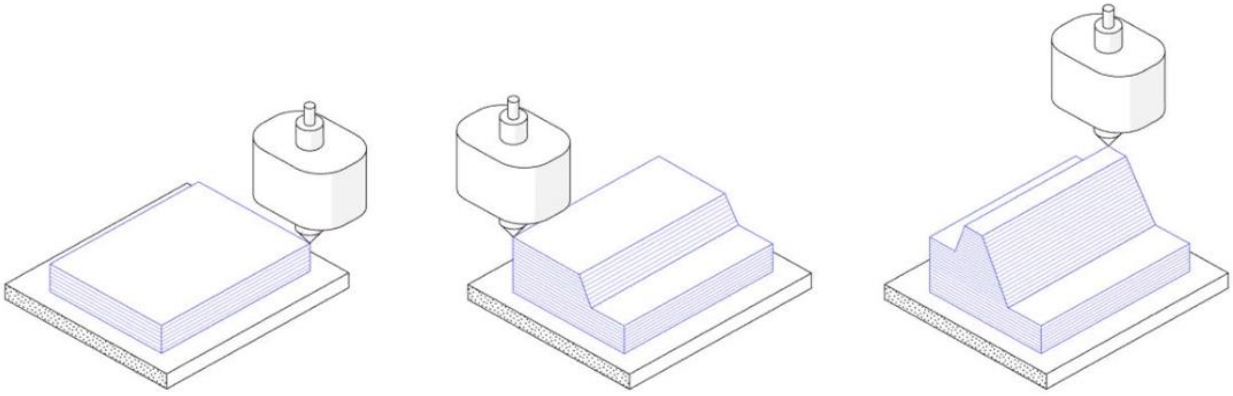


Рисунок 1 – Иллюстрация аддитивного процесса

Как создать и напечатать изделие на 3D-принтере?

В первую очередь нужно получить 3D-модель печатаемого изделия. Ее можно сделать с помощью CAD, получить из 3D-сканнера или скачать из онлайн-репозитория [3]. Наиболее распространенные форматы йлов 3D моделей, используемые в печати - STL, Obj.

- STL – самый популярный формат 3D-модели, хранит информацию только о геометрии объекта.
- OBJ – также популярный формат. В отличие от STL, может также хранить информацию о цвете и текстуре модели. Предположительно, станет популярнее при появлении полноценной многоцветной печати.

После получения 3D-модели, ее нужно преобразовать в код, готовый к выполнению на принтере. Это делается с помощью слайсера [4] – программы, которая преобразует 3D-модель в набор 2D-слоев толщиной

обычно в 0,5 мм. Слайсер также добавляет структуры поддержки, которые необходимы для печати нависающих деталей. Слайсер позволяет положение изделия на печатном столе, изменить размер изделия, имеет настройки заполнения и принтера (например, скорость печати и температура сопел), а для принтеров с несколькими экструдерами позволяет настроить отдельно каждый экструдер. В том случае, когда конфигурация принтера, для которого нужно сгенерировать инструкции, неизвестна, можно настроить свою, выбрав соответствующую опцию при настройке и задав все необходимые характеристики. Готовые к исполнению на принтере инструкции сохраняются в формате g-code. После загрузки файла g-code в принтер и завершения печати, нужно выполнить постобработку: удалить поддерживающие структуры, отшлифовать, отполировать и покрасить изделие.

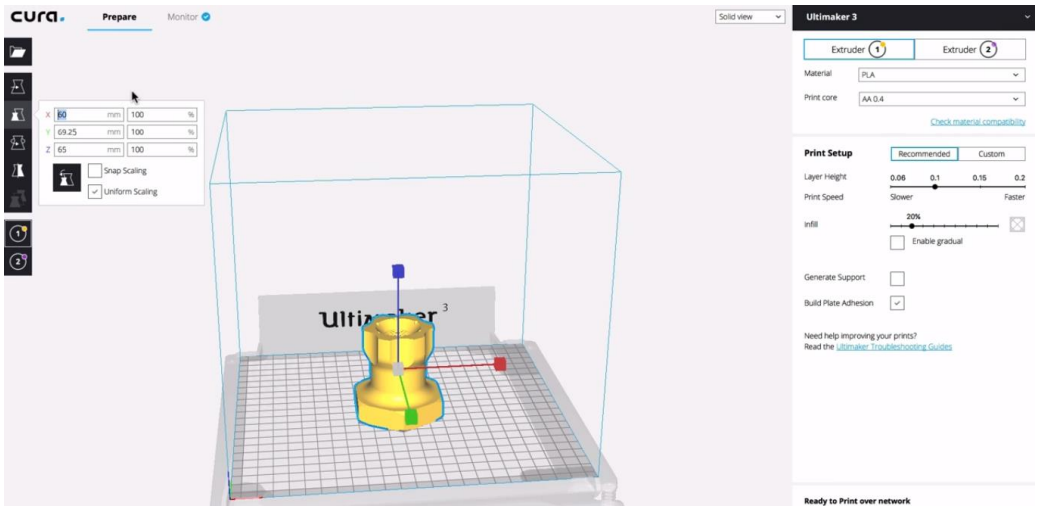


Рисунок 2 – Работа в слайсере Cura

Структура G-code

Программу g-code условно можно разделить на несколько блоков-этапов:

1. Комментарии с различной информацией
2. Подготовительные операции
3. Рабочая часть – изготовление детали
4. Заключительные операции

В блоке подготовительных операций перед печатью задается температура стола, система измерений и начальная точка сопла. Рабочая часть в основном состоит из перемещений по координатам и выдавливания расплава, а в блоке заключительных операций отключается нагрев и двигатели.

```
M190 S70.000000
M109 S200.000000
;Sliced at: Mon 30-11-2015 19:10:26
;Basic settings: Layer height: 0.25 Walls: 1.2 Fill: 100
;Print time: 1 hour 21 minutes
;Filament used: 5.61m 44.0g
;Filament cost: None
;M190 S100 ;Uncomment to add your own bed temperature line
;M109 S200 ;Uncomment to add your own temperature line
G21         ;metric values
G90         ;absolute positioning
M82         ;set extruder to absolute mode
M107        ;start with the fan off
G28 X0 Y0   ;move X/Y to min endstops
G28 Z0      ;move Z to min endstops
G1 X100 Y100 Z50 F4200 ;move the platform down 15mm
G92 E0                 ;zero the extruded length
G1 F200 E4             ;extrude 3mm of feed stock
G92 E0                 ;zero the extruded length again
G1 F4200
;Put printing message on LCD screen
M117 Printing...
```

code

```
2269 G1 F1800 E84.81274
2270 G0 F4200 X97.710 Y111.425 Z0.550
2271 ;TYPE:WALL-INNER
2272 G1 F1800 E87.81274
2273 G1 F2100 X97.124 Y111.144 E87.82802
2274 G1 X96.911 Y110.936 E87.83502
2275 G1 X96.906 Y110.912 E87.83560
2276 G1 X96.629 Y110.328 E87.85079
2277 G1 X96.555 Y110.114 E87.85612
2278 G1 X96.452 Y109.491 E87.87097
2279 G1 X96.450 Y109.196 E87.87790
2280 G1 X96.410 Y108.975 E87.88318
2281 G1 X96.151 Y108.160 E87.90329
2282 G1 X96.057 Y107.828 E87.91140
2283 G1 X95.114 Y106.537 E87.94900
```

Рисунок 4 – Рабочая часть g-code

```
109640 ;End GCode
109641 M104 S0                     ;extruder heater off
109642 M140 S0                     ;heated bed heater off (if you have it)
109643 G91                                 ;relative positioning
109644 G1 E-1 F300                         ;retract the filament a bit before lift
109645 G1 Z+0.5 E-5 F4200 ;move Z up a bit and retract filament even more
109646 G28 X100 Y0 Z0                              ;move X/Y to min endstops, so the h
109647 ;G28 Z0
109648 M84                         ;steppers off
109649 G90                         ;absolute positioning
```

Рисунок 5 – Заключительные операции g-code

Основные технологии 3D печати

Технологий 3D-печати на данный момент существует достаточно много [5]. Эта статья охватывает только основные из них.

– FDM (послойное наплавление). Самая доступная и самая распространенная в мире. Пластик как расходный материал нагревается до 200 градусов и выталкивается через сопло. Сопло выталкивает расплавленный материал слой за слоем. Каждый слой прилипает к тому, что находится под ним.STL – самый популярный формат 3D-модели, хранит информацию только о геометрии объекта.

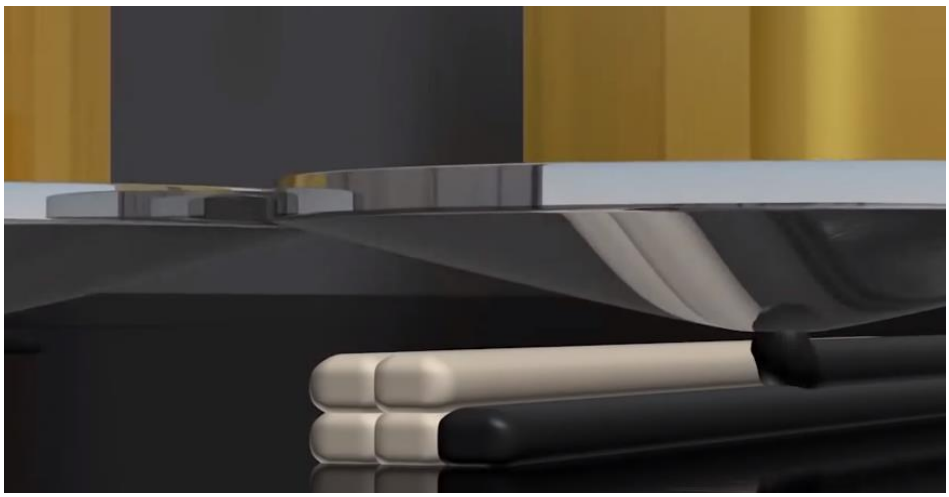


Рисунок 6 – Принцип работы FDA

– SLA (стереолитография). В емкость с жидким фотополимером помещается сетчатая платформа, на которой будет происходить выращивание прототипа. Изначально платформа устанавливается на такой глубине, чтобы ее покрывал тончайший слой вещества, толщиной всего 0.05-0.13 мм — по сути это и есть толщина слоя в лазерной стереолитографии. Далее включается лазер, воздействующий на те участки полимера, которые соответствуют стенкам заданного объекта, вызывая их затвердевание. После этого вся платформа снова погружается на глубину одного слоя.

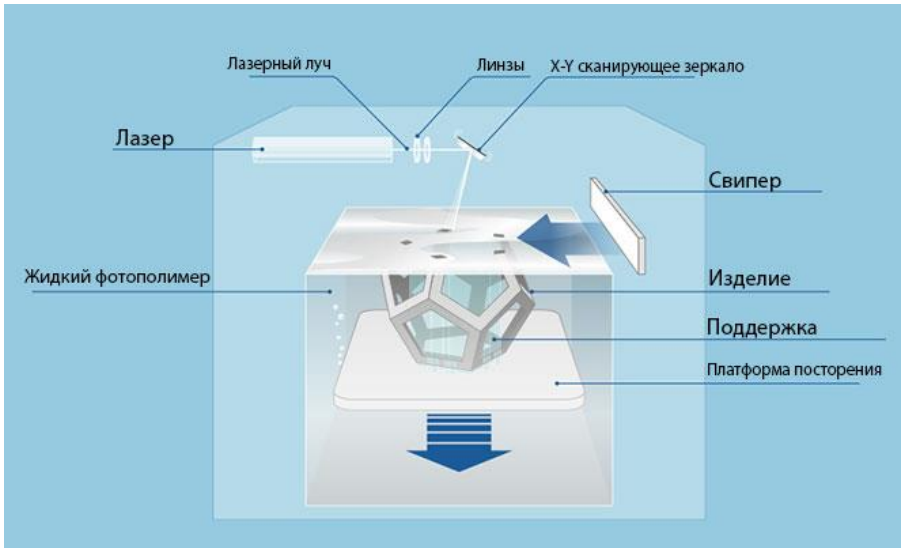


Рисунок 7 – Принцип работы SLA

– SLS (селективное лазерное спекание). Технология заключается в послойном спекании частиц порошкообразного материала для образования объекта по заданной CAD-модели. Выращивание изделия происходит снизу вверх. Каждый слой наращиваемого изделия спекается, повторяя контуры слоев цифровой модели.


Рисунок 8 – Селективное лазерное спекание

Достоинства и ограничения 3D печати

Достоинства 3D-печати:

1. Сложность формы без дополнительных затрат. 3D печать позволяет с легкостью производить сложные формы, многие из которых не могут быть произведены любым другим методом. Геометрическая сложность не означает более высокую себестоимость изделия. Части со сложной или ограничесской геометрией в производстве стоят столько же, сколько более простые детали. Разница лишь в количестве и стоимости необходимого материала.
2. Очень низкие затраты на запуск производства. В отличие от традиционного подхода, где для каждой отдельной детали нужна отдельная форма, 3D печать не требует никаких специализированных инструментов, здесь вообще нет затрат на запуск производства. Стоимость изделия, напечатанного на 3D принтере, зависит только от стоимости и количества материала, который был использован при печати, а также от времени, необходимого на печать и от стоимости пост-обработки (если это необходимо).
3. Легкая изменяемость каждой части. Чтобы напечатать измененную модель, нужно всего лишь изменить 3D-модель изделия и запустить процесс печати. Таким образом, каждый элемент в целях удовлетворения нужд пользователя может быть изменен без увеличения стоимости производства.
4. Быстрое и дешевое прототипирование. Одна из основных сфер применения 3D-печати – прототипирование, как формы изделия, так и его функциональности. Части, напечатанные на 3D-принтере, обычно готовы уже за ночь. Скорость прототипирования уменьшает время и затраты на процесс проектирования. Как следствие, продукты, на разработку которых раньше требовалось 8-10 месяцев, теперь могут быть выполнены за 8-10 недель.
5. Большое количество материалов для печати. Самый популярный материал – пластик. Тем не менее, использоваться может любой материал, который способен менять свою форму. В промышленности используется печать металлом; из других популярных материалов – смола, нейлон, резина. Также

используются смеси. Таким образом можно создать изделие с уникальными свойствами для необычного применения. Напечатанные из современных материалов части могут быть устойчивы к высоким температурам, достаточно жесткими и даже быть биосовместимыми, в зависимости от задач, для которых они печатаются.

Ограничения 3D печати:

1. Долгое время печати. 3D печать занимает немало времени. Единственное решение – закупка дополнительных принтеров.
2. Ограниченные размеры рабочей области 3D-принтера. 3D принтер работает на рабочей поверхности, или печатном столе. Это накладывает ограничение на максимальный размер модели. Проблема решается печатью отдельных частей модели и последующим склеиванием или сплавлением деталей.
3. Меньшая прочность материала. Из-за того, что изделие изготовлено слой за слоем, оно не такое крепкое, как аналог из цельного куска материала. (Обычно разница составляет от 10 до 50%). Поэтому части, которые напечатаны на 3D принтере, обычно используются в некритических местах.
4. Необходимость удаления поддерживающих конструкций [6]. При производстве сложных деталей, которые «нависают» над воздухом, т.е. имеют угол более 45 градусов, приходится использовать вспомогательные конструкции. После завершения процесса печати эти структуры нужно убрать, а места соединения нужно дополнительно отшлифовать.

Заключение

Список технологий и процессов 3D печати продолжает увеличиваться, потому что 3D печать постоянно меняется и развивается. Индустрия 3D печати продолжает совершенствовать оборудование, материалы и технологии. В зависимости от многих факторов, таких как размер бюджета, дизайн и функциональность, важно выбрать подходящую технологию и материал для 3D печати. 3D печать позволяет печатать множество различных объектов, которые ранее могли быть созданы только с использованием сложных технологий производства. Популярность и применение 3D печати продолжает расти.

Литература

1. Изменение размера рынка 3D-печати в 2018 году по сравнению с 2015. тема автоматического
2. Аддитивное производство // Wikipedia.org [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Аддитивное_производство
3. Репозиторий // Wikipedia.org [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Репозиторий
5. Типы 3D печати //3dhubs.com [Электронный ресурс] – Режим доступа:
6. П
7. What is 3D printing? The definitive guide //3dhubs.com [Электронный ресурс] – Режим доступа: https://www.3dhubs.com/guides/3d-printing

Горбань В.В., Григорьев А.В. Обзор современных технологий и языков программирования для 3D-принтеров. *В статье дан обзор основ 3D печати, рассмотрены преимущества и недостатки, рассмотрен процесс создания изделия на 3D принтере. Актуальность данной темы обусловлена все возрастающей популярностью и распространенностью технологий 3D-печати.*

Ключевые слова: *3D-печать, 3D-принтер, 3D-модель, слой, сопло, изделие, наплавление, стереолитография, лазерное спекание, g-code.*

Gorban V.V., Grigoriev A.V. Overview of modern technologies and programming languages for 3D printers. *The article gives an overview of the basics of 3D printing, the advantages and disadvantages, the process of creating a product on a 3D printer. The relevance of this topic is due to the increasing popularity and prevalence of 3D printing technologies.*

Keywords: *3D printing, 3D printer, 3D model, layer, nozzle, product, fusion, stereolithography, laser sintering, g-code.*

УДК 004.032

Анализ функциональных возможностей современных систем управления «умными» теплицами и перспективы развития

Дубянская А.О., Григорьев А.В.
Донецкий национальный технический университет
grigorievalvl@gmail.com, dubjnskaj@yandex.ru

Дубянская А.О., Григорьев А.В. Анализ функциональных возможностей современных систем управления «умными» теплицами и перспективы развития. *В статье описаны ключевые компоненты, с помощью которых создаются «умные» теплицы, функции, над которыми должна осуществлять контроль система управления, описана специфика задачи. Продемонстрирован анализ функциональных возможностей подобного рода систем, описаны перспективы развития.*

Ключевые слова: *«умная» теплица, управление, система, анализ, требования, функциональность.*

Введение

Традиционно эпитетом «умные» принято величать всевозможные предметы или объекты, наделенные теми или иными саморегулирующимися функциями. Например, «умный пол» умеет при помощи датчиков и ИК-камер отслеживать месторасположение людей и предметов в помещении. И даже, при необходимости, способен вызывать «скорую» или полицию. А «умный дом», благодаря все тем же высоким технологиям, обеспечивает для своих жильцов комфорт, безопасность и ресурсосбережение. Попросту говоря, умные вещи – это бытовые «помощники», настроенные на решение простых, рутинных ежедневных задач – включать и выключать свет, управлять сплит-системой, поднимать и опускать шторы и жалюзи и т.д. И теплицы в этом списке – не исключение.

Как правило, в любой теплице, претендующей на статус «умной», соблюдаются три золотых правила:

— температурный режим внутри сооружения поддерживается автоматически;

— полив растений осуществляется своевременно и в нужном количестве без участия человека (капельное орошение);

— почва в грядках или контейнерах восстанавливается самостоятельно и не требует частой замены.

Актуальность работы: создание «умной» теплицы значительно упрощает процесс выращивания овощей, иногда, теплица настолько «умная», что пользователю остаётся лишь изредка наблюдать и контролировать показатели микроклимата в теплице. Что даёт использование «умных» теплиц:

— экономия времени, затрачиваемого на «ручное» выращивание и контроль растений в теплице (это достигается тем, что система управления сама контролирует все показатели в теплице);

— экономия энергоресурсов;

— экономия трудовых ресурсов, уже не нужно будет нанимать большое количество людей, которые будут контролировать процессы, происходящие в теплицах.

Цель работы: исследование и анализ функциональных возможностей «умных» теплиц, рассмотрение методов реализации подобных систем, их перспективы развития.

Постановка задачи создания «умной» теплицы

Работая над этим материалом, было решено воспользоваться описанием одной из самых удачных моделей (теплица Малышевского) для того, чтобы проиллюстрировать весь процесс правильной организации умной теплицы [1].

Правило №1: Для строительства умной теплицы необходимо выбрать умное расположение.

При выборе места под будущую садовую «умную» теплицу необходимо учитывать три основных фактора:

— назначение теплицы;

— географическую широту вашего участка;

— схему ветровой активности (розу ветров) вашего района.

В основе всех моделей умных теплиц лежит принцип максимального использования солнечной энергии. Однако в разных широтах солнце ведет себя по-разному. Кроме того, далеко не все тепличные культуры солнцелюбивы. К примеру, для выращивания огурцов, салатов и болгарского перца, то следует располагать теплицу с юга на север. Таким образом, она будет получать оптимальное количество солнца утром и вечером и в тоже время, не перегреваться в дневную жару. Для возделывания томатов и баклажан лучше подойдет теплица, ориентированная с востока на запад.

Сильный ветер — это закадычный враг всех тепличников. Он может резко изменить внешний температурный фон или же попросту сорвать с места недостаточно укрепленную конструкцию. Поэтому, при выборе места под теплицу нужно учесть направление основных ветров. А при необходимости – организовать дополнительную ветровую защиту. Например, устроить живую изгородь в 10-15 метрах от теплицы.

Правило №2: При проектировании или выборе каркаса для умной теплицы фрамуги для проветривания находились как можно выше.

Чем воздух холоднее, тем он «тяжелее». То есть, холодный воздух в помещениях всегда стелется по полу, а перепад температур для растений может обернуться не только сильным стрессом, но и гибелью. Выход только один – устроить в «умной» теплице систему проветривания без сквозняков. Во-первых, форточки для проветривания должны находиться как можно выше, в идеале – на крыше теплицы. А во-вторых, двери в теплицу должны быть без щелей и плотно закрываться. При проникновении холодного воздуха через «высокие» форточки, он смешивается с горячим «верхним» воздухом. Таким образом, происходит обмен влагой и теплом и, в результате, растения чувствуют себя замечательно. При правильном расположении форточек средний температурный фон в теплице летом будет держаться на уровне от +32 до +35°С.

Правило №3: В «умной» теплице почва тоже должна быть «умной».

Суть метода заключается в следующем: в качестве почвы используется биогумус «домашнего» изготовления; сверху плодородный слой покрывается органической или минеральной мульчей, толщиной до 10 сантиметров; почва обязательно «заселяется» дождевыми червями, которые в процессе жизнедеятельности обеспечивают растения всеми необходимыми удобрениями, вот такой получается умный симбиоз.

Правило №4: Обеспечение «умной» теплицы надежной автоматикой для проветривания и полива.

Пожалуй, что главными атрибутами «умной» теплицы являются автоматизированный полив и обеспечение оптимальной температуры для роста и вызревания культур. Сегодня существует множество автоматических устройств, отличающихся как по производительности, так и по функциональности. Но самые удобные и надежные – это автономные автоматические «открыватели», работающие по принципу гидроцилиндров. Принцип работы этих устройств прост: при нагревании жидкость внутри цилиндра расширяется и приводит в действие шток-толкатель, который, в свою очередь, открывает форточку или кран, подающий воду в систему орошения. И наоборот – при понижении температуры жидкость в цилиндре охлаждается и шток возвращается на исходную (см. рис. 1).

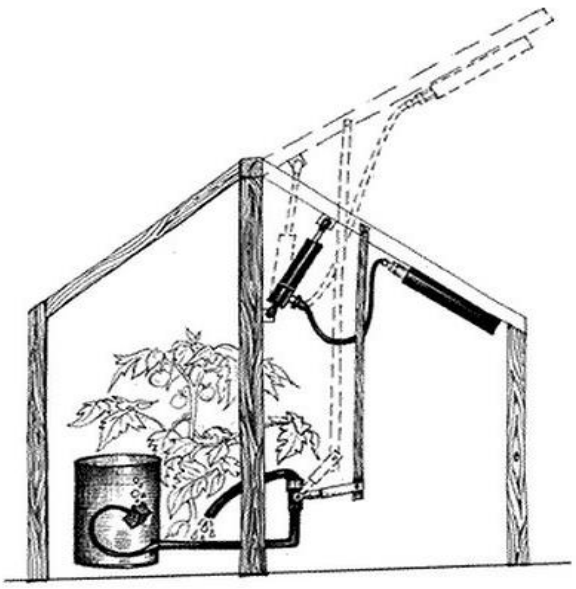


Рисунок 1 - Схема работы автомата по проветриванию теплиц.

Правило № 5: Чтобы сберечь теплоту после захода солнца следует использовать аккумуляторы тепла.

Большая емкость с водой, которую используют для капельного орошения растений, послужит отличным накопителем тепла, и будет обогревать теплицу в темное время суток. Также, можно организовать бетонную дорожку между грядками. Однако, солнечное тепло может не быть не только полезным, но и очень вредным. Деликатность проблемы заключается в том, что разница между идеальной температурой взращивания и границей, после которой растения погибают, весьма невелика. При температуре окружающей среды +40°С растения начинают болеть и погибать. Для того, чтобы оградить тепличные растения от летнего перегрева можно «затонировать» окна водным раствором мела. После того как жара спадет, теплицу можно будет легко очистить обыкновенной водой. А для того, чтобы эта тонировка не влияла на аккумулирование тепла емкость с водой следует покрасить черной матовой краской.

Методы реализации современных систем управления «умными» теплицами

Разные компании, фирмы и просто группы людей предоставляют и проектируют совершенно разное ПО для автоматизации системы управления теплицей, но преследуют совершенно одинаковые цели – улучшить качество продукции на выходе. К одной из таких групп относится небольшая команда профессионалов, разрабатывающих электронное оборудование и программное обеспечение для автоматизации работ в теплицах, на дачах и приусадебных участках. Их программное обеспечение автоматизированной системы управления предназначено для управления сервером автоматизированной системы управления и организации взаимодействия периферийных устройств с сервером автоматизированной системы управления [2].

Для обеспечения взаимодействия ПО АСУ с ПУ должна быть организована локальная сеть: сервер АСУ и все ПУ должны находиться в одной подсети. ПУ подключаются в локальную сеть через Wi-Fi. В связи с этим, для организации локальной сети нужно применять Wi-Fi роутер (или устройство, заменяющее его). При этом применение смартфона или планшета в режиме точки доступа пока не поддерживается в ПО АСУ.

Для обеспечения гибкости управления процессами автоматизации пользователь в ПО АСУ создает набор программ управления процессами автоматизации. Программы запускаются по системным событиям, по событиям от датчиков или непосредственно пользователем и обеспечивает требуемую гибкость управления процессами автоматизации.

Основной областью применения АСУ является автоматизация процессов управления выращиванием овощей, фруктов, грибов, управление микроклиматом теплиц. Возможно применение на приусадебных участках (см. рис. 2).

Рисунок 2 – Структурная схема СУМТ

Анализ функциональных возможностей современных «умных» теплиц

К числу основных функциональных возможностей действительно «умной» теплицы относятся:

— контроль и управление светом;
— контроль полива;
— контроль температуры и влажности;
— удаленный мониторинг текущего состояния теплицы;
— автоматическое управление теплицей;
— автоматический процесс фотографирования роста растений;
— низкая потребляемая мощность;
— защита от коротких замыканий и отключения электричества.

Таким образом, подобная «умная» теплица представляет собой полнофункциональную автоматизированную систему, которая позволяет выращивать разного рода растения, не беспокоясь об их состоянии каждый момент времени. Такая теплица позволяет значительно освободить своё время, не посвящая его полностью работе в теплице, контролируя все процессы, которые там происходят, ведь это за вас делает система, а вы лишь отслеживаете правильность её выполнения удаленно.

Заключение

В данное время разработка «умных» теплиц переживает стадию бурного развития. Развитие «умных» теплиц имеет существенные преимущества по сравнению с обычными теплицами, управляемыми людьми. Использование новых передовых технологий в деле теплицеведения позволит значительно расширить функционал теплицы, а значит, как следствие, увеличить качество выращиваемого продукта и ускорить его процесс выращивания.

Сейчас на рынке популярны разного рода платы с собственной памятью и процессором. Такие платы можно оснастить любыми функциями, например, можно использовать шаговый двигатель, который сможет открывать и закрывать окна в теплице, или воду для полива или орошения. Платы можно оснастить видеокамерой с постоянным выходом в интернет, и в режиме реального времени наблюдать за происходящими процессами в теплице из любого места нахождения. Используя несколько подобных плат можно создать настолько «умную» теплицу, что она будет не только следить за всеми процессами, происходящими в теплице, но и будет осуществлять сбор урожая. Например, для сбора урожая можно создать устройство, которое автоматически сможет полоть растения в определенные промежутки времени, включая в себя серво датчики, к которым будет прикреплен инструмент.

Для подобного проекта «умной» теплицы существует много идей и доработок, подобный проект можно совершенствовать и совершенствовать, привнося в функционал системы всё новые и новые решения. Данная технология заинтересует, в первую очередь, теплицеводов, имеющих огромные тепличные участки, такой вид теплиц значительно сэкономит время, затрачиваемое на выращивание растений и их, возможно, продажу.

Литература

1. Курдюмов Н., Мальшевский К. Умная теплица, 2014, с. 8.
2. Умная теплица. Программное обеспечение // gh-smart.ru [Электронный ресурс] – Режим доступа: https://gh-smart.ru/index.php/programmnoe-obespechenie.

Дубянская А.О., Григорьев А.В. Анализ функциональных возможностей современных систем управления «умными» теплицами и перспективы развития*. В статье описаны ключевые компоненты, с помощью которых создаются «умные» теплицы, функции, над которыми должна осуществлять контроль система управления, описана специфика задачи. Продемонстрирован анализ функциональных возможностей подобного рода систем, описаны перспективы развития.*

Ключевые слова: *«умная» теплица, управление, система, анализ, требования, функциональность.*

Dubyanskaya, A.O., Grigoriev, A.V. Analysis of the functionality of modern control systems "smart" greenhouses and development prospects. *The article describes the key components with the help of which "smart" greenhouses are created, the functions over which the control system should exercise control, describes the specificity of the task. The analysis of the functional capabilities of such systems is demonstrated, and the development prospects are described.*

Keywords: *«smart» greenhouse, management, system, analysis, requirements, functionality.*

УДК 004.023

Анализ функциональных возможностей и перспектива развития системы автоматического составления рациона правильного питания

Капков Ю.Д., Григорьев А.В.
Донецкий национальный технический университет
kapkov555@gmail.com, grigorievalvl@gmail.com

Капков Ю.Д., Григорьев А.В. Анализ функциональных возможностей и перспектива развития САПР по составлению рациона правильного питания. *В статье описан термин «Правильное питание», основные этапы составления рациона в практике врачей диетологов, профессиональных спортсменов и в практике САПР.*

Ключевые слова: *САПР правильного питания, САПР «Здоровое питание и диета», рацион, здоровье, энергетическая ценность, БЖУ.*

Введение

Одним из основных факторов, влияющих на здоровье человека, является питание. В современном мире,

где на счету каждая секунда, большой процент населения питается полуфабрикатами, что приводит к таким заболеваниям [1] как ожирение, сахарные диабет, гипертония, атеросклероз, заболевания сердца и даже рак. Также правильное питание необходимо и людям, которые ведут активный образ жизни, чтобы достичь новых высот. Решить данные проблемы поможет САПР по составлению правильного питания.

Наиболее развитые системы по составлению правильного питания включают:

1) программы, которые специализируются на составлении баз данных всевозможных продуктов и их энергетической ценности.
2) программы, которые специализируются на расчете потребностей организма человека, основываясь на таких показателях как рост, вес, возраст, активность и другие.
3) программы, которые содержат всевозможные рецепты всех кухонь мира.
4) программы анализаторы, которые получают данные от различных фитнес браслетов и определяют количество необходимой энергии, для полноценного функционирования организма.

Современные САПР должны удовлетворять следующим требованиям:

1) полная функциональность, способная решать весь спектр задач от расчета необходимого количества БЖУ, исходя от целей пользователя, до составления рациона на указанный период.
2) простота и мобильность
3) динамичность, т. е. развитие и совершенствование продукта, который будет постоянно наращивать свой функционал.

Актуальность работы: в последнее время большой акцент делается на развитие информационных технологий, что приводит к значительным изменениям в области автоматизации проектирования различных отраслей. Во многих странах активно разрабатываются современные системы проектирования, которые повышают качество выполняемой работы, вследствие чего эффективность труда возрастает, а затраченное время падает. Это наблюдается также и в сфере, затрагивающей здоровье человека.

Цель: исследование функциональных возможностей САПР правильного питания в общем, а также на примере конкретной программы; рассмотрение методов проектирования правильного питания в реальной жизни, и в системе САПР.

Правильное питание как объект проектирования

Что же такое правильно питание? Это питание, которое обеспечивает рост, нормальное развитие и жизнедеятельность человека, укрепляющее его здоровье и способствует профилактике различных заболеваний. Если соблюдать правила здорового питания и заниматься физическими нагрузками, то есть высокая вероятность сократить риск возникновения хронических заболеваний. Сейчас проводится большое количество научных исследований, чтобы составить оптимальный рацион для предупреждения многих заболеваний. Руководители многих развитых стран регулярно вкладывают средства на популяризацию здорового образа жизни, а в частности и на правильное питание.

Разнообразие питания очень велико и включает в себя такие виды: всеядное питание, раздельное питание, дробное питание, вегетарианство, веганство, сыроедение, фруторианство, жидкое питание и другие.

Существуют определенные правила и нормы, которые применяются для составления рациона. Для среднестатистического человека существуют такие правила:

1) питание должно быть разнообразным и при этом должна преобладать растительная пища, а не животная.
2) основой питания является хлеб, макаронные изделия, зерно, рис и картофель.
3) в течение дня необходимо употреблять фрукты и овощи.
4) ежедневно употреблять молочные и кисломолочные продукты.
5) необходимо употреблять не менее 90 г нежирного мяса в сутки.
6) необходимо исключить алкоголь.
7) нормой употребления пищевой соли - 5 г в сутки.

Методы составления правильного питания врачом диетологом

У каждого врача диетолога есть свой наработанный план, по которому он составляет рацион для конкретного человека. Приведем пример такого плана [2]:

1) проведение осмотра.
2) общение с пациентом и сбор информации.
3) тщательный анализ образа жизни, а также выявления цели обращения пациента.
4) анализ текущего питания и образа жизни.
5) проведение необходимых обследований (измерение роста, веса, объем талии, бедер и другие).
6) составление нового рациона.

Рисунок 1 – Процесс составления рациона

Данный процесс можно поделить на три условных этапа:

— сбор информации
— анализ информации
— создание рациона

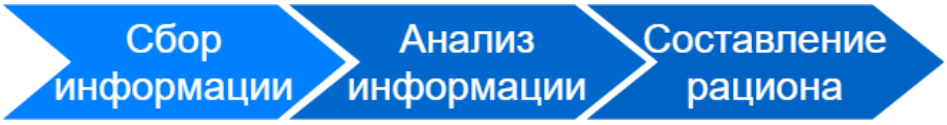


Рисунок 2 – Условные этапы формирования рациона

Общие методики составления правильного питания в САПР

Существуют типичные этапы составления правильного питания в САПР:

1. Сбор входных данных о пользователе: программа принимает на вход все необходимые параметры пользователя, такие как:

— рост
— вес
— возраст
— обхват талии
— обхват шеи
— цель подбора питания
— уровень активности

2. Анализ полученных данных: благодаря специальным алгоритмам программа на основе исходных данных производит расчет необходимого количества энергии, жидкости, БЖУ для данного пользователя.

3. Генерация рациона правильного питания: программа анализирует базу данных, которая в себе хранит всевозможные продукты питания, а затем начинает подбор рациона с учетом личных предпочтений пациента, чтобы удовлетворить все его энергетические потребности.

Анализ функциональных возможностей современных САПР по питанию на примере САПР «Правильное питание и диета» — мобильное приложение по контролю и расчету питания. Назначение, общие функции и структура САПР «Здоровое питание и диета»

САПР «Здоровое питание и диета» [3] – одна из популярных систем по автоматизации процессов составления рациона правильного питания. При помощи данной системы можно выполнять следующую работу: сбор базовой информации о пользователе; анализ входных данных и расчет необходимого количества энергии на день; примерный подбор необходимых продуктов, которые необходимо съесть в течение дня; рекомендовать самые популярные блюда и отображать их рецепты приготовления; предоставляет возможность вноски коррективов при расчете необходимой энергии.

На рис. 3 показан интерфейс САПР «Здоровое питание и диета», а – в таб.1 – структура САПР «Здоровое питание и диета».

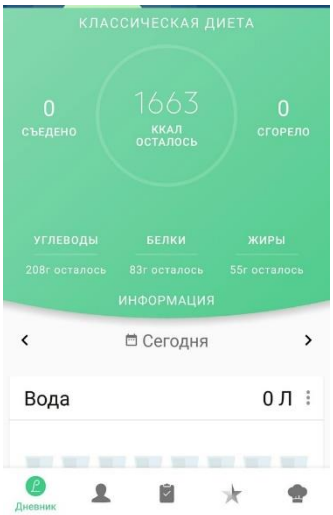


Рисунок 3 – Интерфейс САПР «Здоровое питание и диета»

Таблица 1 – Пункты меню САПР «Здоровое питание и диета»

Пункт меню	Назначение
Дневник	В данном пункте меню происходит расчет и отображение количества необходимой энергии, а также количества белка, жиров и углеводов. Также отображение и подсчет съеденных продуктов.
Профиль	В данном пункте меню отображаются все физические характеристики пользователя.
Задачи	Отображается статистика с начала использования приложения, такая как: изменение веса, изменение количества употребляемых продуктов и жидкости, активность человека за указанный промежуток времени.
Рекомендации	Отображаются всевозможные наборы готов диет, а также их описание.
Рецепты	Отображаются рекомендованные блюда, а также множество рецептов для их приготовления.

Генерация рациона в САПР «Здоровое питание и диета»

Порядок генерации рациона правильного питания в САПР «Здоровое питание и диета» выглядит следующим образом:

Шаг 1: Выбор пола, возраста, роста и веса пользователя на момент запуска приложения.

Рисунок 4 – Первый шаг генерации рациона

Шаг 2: Выбор вида активности, а также установка необходимого соотношения БЖУ.

Рисунок 5 – Второй шаг генерации рациона

Шаг 3: Вывод результата генерации рациона

Рисунок 6 – Пример составленного рациона

Обзор перспектив развития САПР по составлению рациона правильного питания

Основными задачами совершенствования современных систем автоматизированного проектирования рациона правильного питания являются:

1) популяризация правильного питания, за счет упрощения составления рациона правильного питания с помощью САПР.
2) разработка и организация САПР на основе единой информационной системы, управляющей всеми этапами составления рациона.
3) применение сторонних приборов, чтобы более детально диагностировать пользователя, с целью более корректного и эффективного рациона.
4) применение интеллектуальных технологий, в том числе использование искусственного интеллекта.

Выводы

В статье приведён анализ проектирования рациона правильного питания во врачебной практике и – практике САПР. Рассмотренная на примере система предлагает ряд функций, обеспечивающая быструю, эффективную работу. Пользователями данной системой могут являться все пользователи, которые обладают навыками работы с компьютером или смартфоном. Выполнен анализ перспектив развития САПР составления рациона правильного питания.

Литература

1. Болезни от неправильного питания [Электронный ресурс] – Режим доступа: http://www.menslife.com/health/bolezni–ot–nepravilnogo–pitaniya.html

2. Статья по практчиеской диетологии [Электронный ресурс] – Режим доступа: https://praktik–dietolog.ru/article/167.html
3. Сайт САПР «правильное питание и диета» [Электронный ресурс] – Режим доступа: http://deta.by/

Капков Ю.Д., Григорьев А.В. Анализ функциональных возможностей и перспектива развития САПР по составлению рациона правильного питания. *В статье описан термин «Правильное питание», основные этапы составления рациона в практике врачей диетологов, профессиональных спортсменов и в практике САПР.*

Ключевые слова: *САПР правильного питания, САПР «Здоровое питание и диета», рацион, здоровье, энергетическая ценность, БЖУ.*

Kapkov Y.D Grigoriev A.V. Analysis of the functional possibilities and prospects of development of CAD software for drafting the diet of proper nutrition. *The article describes the term "Proper nutrition", the main stages of the preparation of the diet in the practice of dietitians, professional athletes and in the practice of CAD.*

Keywords: *CAD of healthy nutrition, CAD "Healthy nutrition and diet", diet, health, energy value, PFC.*

УДК 004.4'27

Анализ методов и инструментальных средств разработки игр

Макорин С.А., Григорьев А.В.
Донецкий национальный технический университет

Макорин С.А., Григорьев А.В. *Анализ методов и инструментальных средств разработки игр. Основы и структура. Рассмотрение компонентов. Навигация по средствам разработки.*

Ключевые слова: *компьютерные игры, движок, 3д модели, интерактивное развлечение, средства разработки.*

Введение

Игры живут столько же времени, сколько живет человечество.

Набор для настольной игры из города Ур. 3е тыс. до н. э. (рис. 1) [1].

Вариацию этой игры, под названием "сенет" любили в др. Египте. Играли все слои населения. Люди, для которых даже выживание, было нетривиальной задачей, тратили массу времени на развлечения.

Это говорит о том, что с незапамятных времён, игры - неотъемлемая часть человеческого быта.

На сегодня производство интерактивных развлечений, превратилось в многомиллиардную индустрию, став важной частью экономики.

Рисунок 1 – Набор для настольной игры из города Ур

Наша область, реализация интерактивных развлечений, при помощи компьютеров. В данной статье мы акцентируем внимание на потребностях разработчиков, а не конечных пользователей продукта.

Актуальной задачей, на сегодняшний день, является не только создание новых программных средств, но навигация в огромном количестве существующих, выбор между аналогами на каждом из этапов производства.

Целью данной статьи не является проанализировать их все, мы постарались выделить только те моменты, которые, по нашему мнению, освещаются менее всего.

Компьютерная игра – как объект разработки

Компьютерную игру можно рассматривать как произведение искусства, что несомненно налагает свои особенности на процесс её производства, но это также одна из самых сложных компьютерных программ. Чтобы лучше понять средства её разработки, рассмотрим несколько вариантов описания её структуры [2, 3].

По динамике взаимодействия с пользователем, можно выделить три уровня – оперативный, тактический и стратегический.

Оперативный — представляет собой совокупность действий внутри программы между двумя последовательными действиями игрока. Результатом действия оперативного уровня является отображение всех перемещений и изменений на экране дисплея.

Тактический — определяется как совокупность игровых действий, ведущих к достижению какой-либо локальной цели. В результате действия тактического плана играющий достигает улучшения (или ухудшения) положения в игре.

Стратегический — предполагает планирование всей игры, которая должна строиться так, чтобы достичь цели и добиться выигрыша.

Логическая структура компьютерной игры

Игровая среда – это совокупность всех объектов и связей в игре и законов их изменения. Другими словами, ИС – это основа, "мир", в котором развивается игровое действие. Так, в шахматах игровой средой будет совокупность, в которую входят: доска, два набора фигур, правила перемещения фигур по доске, а также правила взятия (и превращения) фигур. В популярной КИ "Посадка на Луну" игровой средой будет вектор, описывающий параметры "сажаемого на Луну корабля" (массу, запас топлива, высоту, скорость), а также

некоторое уравнение, связывающее эти параметры. Для распространенной КИ "Pacman" ИС – это лабиринт, в котором передвигаются персонажи: один, управляемый играющим, и несколько "врагов", управляемых программой.

Взаимодействие с играющим – это совокупность средств, предоставляемых играющему для изменений игровой среды, т. е. для действий и изменений в ИС, которые происходят, когда играющий нажимает определенные клавиши дисплея. Заметим, что игры реального времени от игр-головоломок отличаются именно этой компонентой: в играх реального Бремени период времени между нажатиями существенно влияет на ход игры.

Оценка игровой ситуации – это соотношения и условия, которые определяют задачу для играющего в данной игре. Сюда включается система очков и штрафов за игровые действия, описание начальной и конечной игровой ситуации.

Компоненты компьютерной игры. С точки зрения производства

Физический движок — базовая составляющая игровой программы, используемая для расчета столкновений, отрисовки изображения и других основных вещей низкого уровня, но без правил и логики, жестко фиксированных данных конкретной игры, игрового мира, графических ресурсов.

Трехмерные модели — персонажей, декораций, предметов.

Двумерная графика — сюда входит внешний вид интерфейсов, персонажи, декорации, предметы в двумерных играх, вид предметов для инвентаря и др. для трехмерных. Также концепт-арты, не входящие в готовый продукт, но использующиеся в процессе производства визуального контента.

Текстуры

Шрифты

Анимации трехмерных моделей

Игровой мир — в узком смысле цельная или разделенная на уровни либо локации площадь, заполненная объектами, на которой и происходит игровой процесс.

Сценарий, предыстория

Геймплей

Звуковые эффекты и музыка

Стандартный набор инструментария разработчика игр

На рисунке 2 [4] мы видим схему, объединяющую стадии разработки игры, и набор популярных программ, использующихся на соответствующих стадиях. На данном этапе развития индустрии, объединение всех или даже большей части из этих разно этапных средств в одной системе, не рационально.

Не только из-за сложившихся привычных практик, но и по технологическим и организационным причинам. Всё больше распространяются практик, когда игроделы приобретают часть контента, программных модулей у сторонних исполнителей.

Существуют интернет - площадки по продаже готовых 3д моделей и анимации. Наряду с магазинами для пользователей интегрированных сред движков Unreal и Unity и др. они сами являются ценными инструментами, позволяющими не изобретать «велосипед». Т. е. не тратить время на маловажный контент, играющий фоновую роль.

В своё время, повторное использование кода, сыграло большую роль в развитии всей индустрии ПО, в том числе — и игровых программах. Позволив выйти на новые масштабы.

Мы считаем, что повторное использование контента (хотя это явление не новое), может принять и новые формы, и будет и дальше служить развитию индустрии интерактивных развлечений.

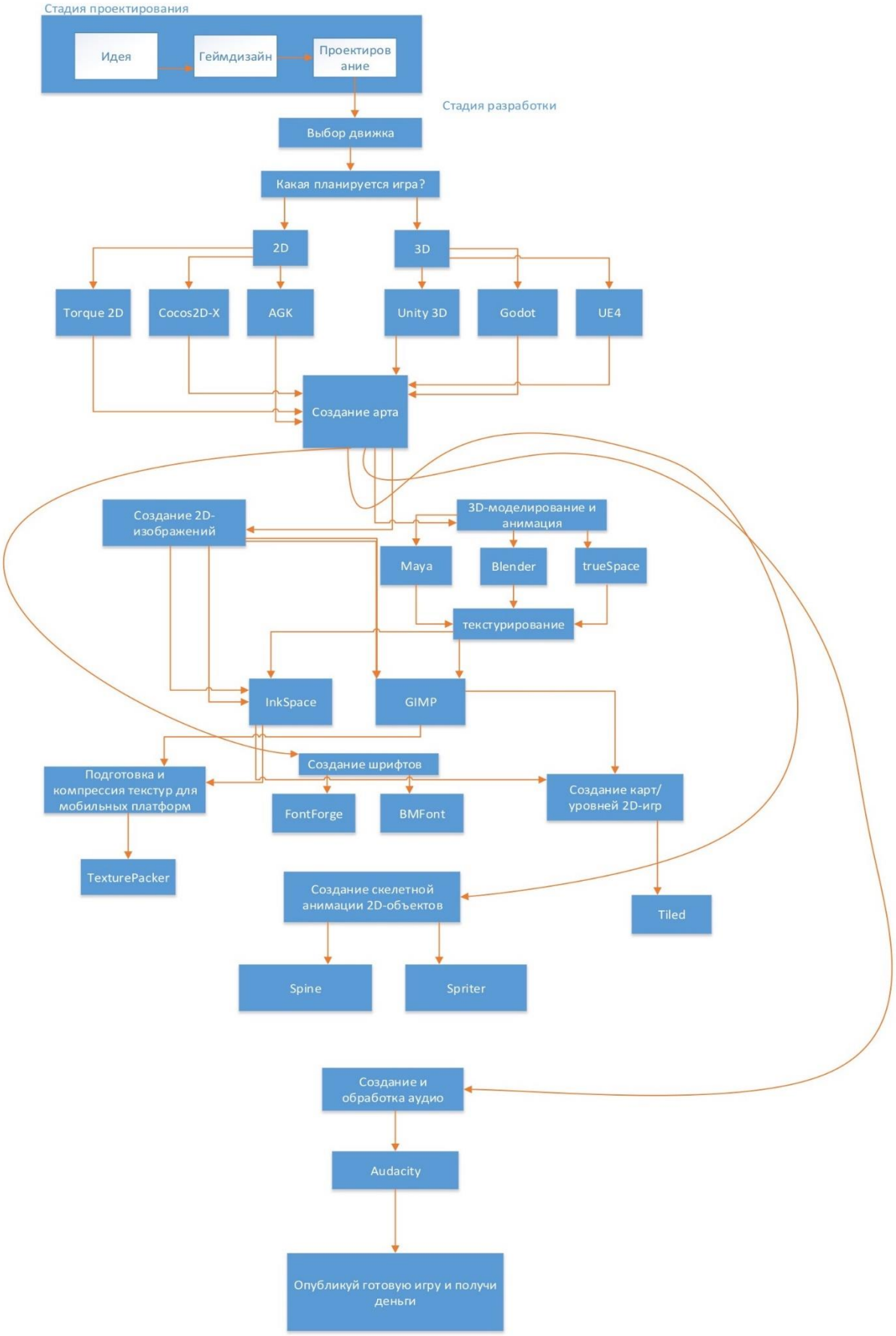


Рисунок 2 – Стандартный набор инструментов разработчиков игр

Человек, как средство производства компьютерной игры

Кроме программных инструментов, не менее важной частью производственного процесса являются соответствующие специалисты. Поскольку большое количество работы производится в головах, на облик готовой игры, или даже, на существование, сильное влияние оказывают конкретные личности. Один из таких примеров, композитор написавший музыку к серии игр Heroes Might and Magic, Пол Ромеро, его музыка – один из существенных элементов внёсший свой вклад в уникальный облик серии.

В таблице 1 [5] мы можем увидеть пример состава команды разработки, небольшой студии. Эта студия на данный момент готовится выпустить трехмерный шутер от первого лица, с элементами квеста и хоррора, в органическом дизайне (Scorn). Здесь можно отметить немаловажную деталь, отдельный человек – концептуальный художник. Его художественный замысел, воплощенный в созданных им двухмерных рисунках, используется 3д художниками и дизайнером уровней для создания моделей существ, декораций, предметов, которые будет видеть игрок. Работы же концептуального художника останутся «за кадром», но именно он служат фундаментом (см. рис. 4-5) [6]. Кроме своей основной производственной задачи, его работы помогают всей команде придерживаться одного общего вектора, чтобы проект оставался цельным произведением искусства, а не набором разнородных элементов.

Таблица 1— Команда студии Ebb Software. Разработчик игры Scorn.

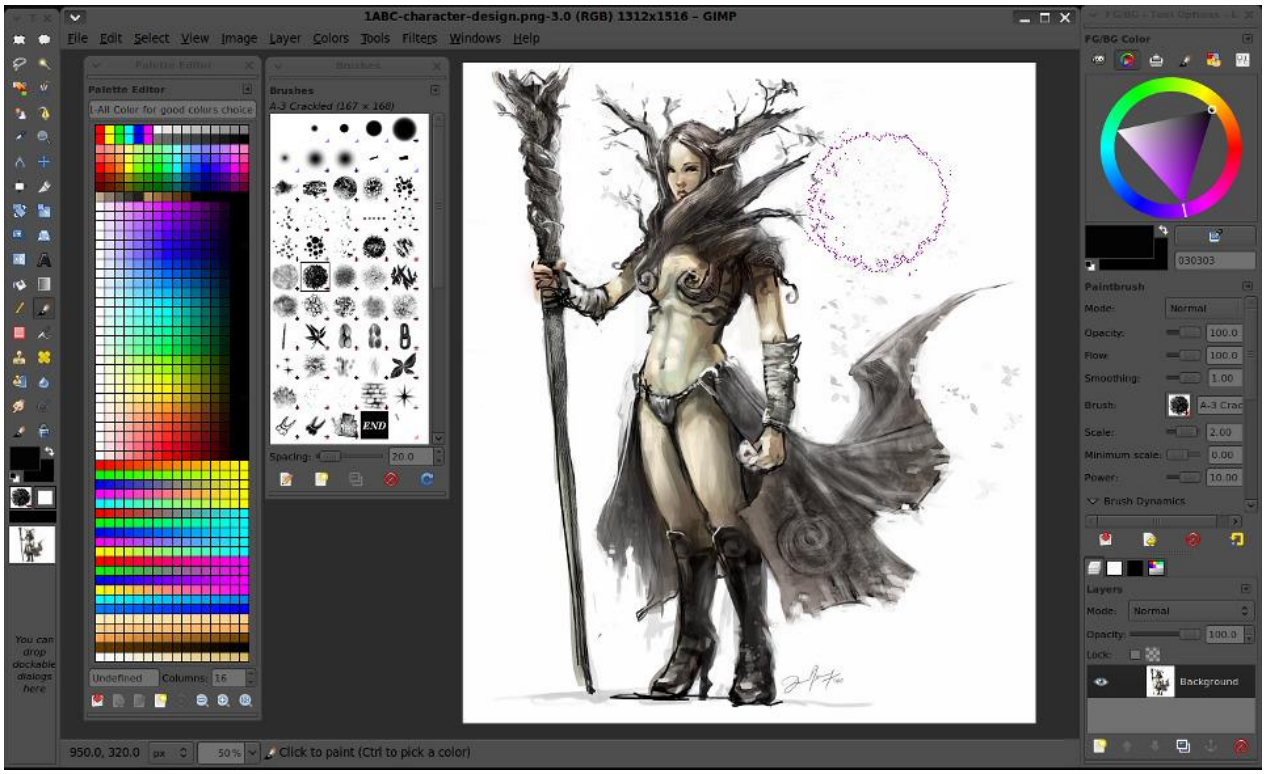


Рисунок 3 - Процесс создания концепт-арта в редакторе Gimp

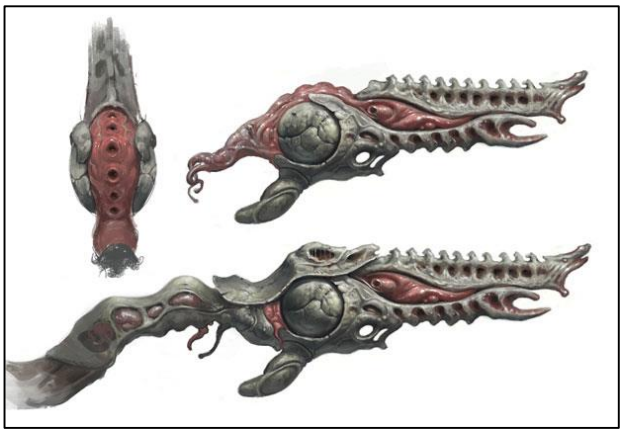


Рисунок 4 — Концепт-арт оружия для игры Scorn

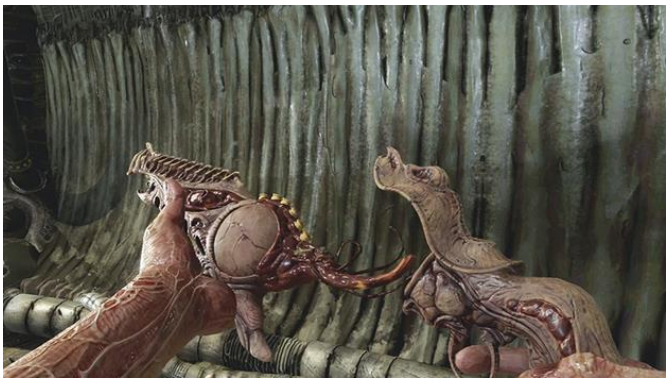


Рисунок 5 -— Реализация концепта оружия в готовой игре, идёт процесс перезарядки

Вывод

В данной статье был проведён обзор и анализ компонентов игры с точки зрения потребностей разработчика, рассмотрены варианты набора инструментария: как программного, так и человеческого. Приведены примеры отдельных, ключевых этапов разработки и популярных инструментов.

Литература

1. Т
2. Ламот, А. Программирование трехмерных игр для Windows. Советы профессионала по трехмерной
3. Роллингс, Эндрю. Моррис, Дэйв. Проектирование и архитектура игр. / Пер. с англ. - М.: Вильямс, 2006. — 1040 с.
4. Журнал Хакер / Инструментарий игродела [Электронный ресурс] – Режим доступа:
5. Геймпедия [Электронный ресурс]– Режим доступа: https://scorn.gamepedia.com/Ebb_Software
6. Официальный сайт игры Scorn [Электронный ресурс]– Режим доступа: https://scorn-game.com

Макорин С.А., Григорьев А.В. Анализ методов и инструментальных средств разработки игр. *Основы и структура. Рассмотрение компонентов. Навигация по средствам разработки.*

Ключевые слова: *компьютерные игры, движок, 3д модели, интерактивное развлечение, средства разработки.*

Makorin S.A., Grigoriev A.V. Analysis of methods and tools for game development. *Basics and structure. Consideration of components. Navigation by means of development.*

Keywords: *computer games, engine, 3D models, interactive entertainment, development tools.*

УДК 004.896

Проблематика создания трёхмерных моделей городов и анализ рынка приложений для 3D-моделирования

Мамутова В.А., Григорьев А.В.
Донецкий национальный технический университет
vlada.mamutova@yandex.ua, grigorievalvl@gmail.com

Мамутова В.А., Григорьев А.В. Проблематика создания трёхмерных моделей городов и анализ рынка приложений для 3D-моделирования. *В статье описаны ключевые моменты по созданию виртуального города, доступного через Интернет. Выполнен обзор приложений для трёхмерного моделирования и планирования городской среды. Проанализированы проблемы, возникающие при трёхмерном моделировании объектов городской застройки, определены возможные пути их решения.*

Ключевые слова: *виртуальный город, трёхмерное моделирование, 3D-модель городской среды, текстурирование, аэрокосмические снимки, хранение пространственных данных.*

Введение

3D-макет города, доступный через Интернет, – уникальная технология, позволяющая получать наглядное представление об архитектуре города, объектах инфраструктуры, быстро перемещаться по виртуальной сцене и изучать обширные территории [1], погрузиться в городскую среду и воспринять её так, как она выглядит в реальности.

Потребность в реалистичном отображении окружающего мира увеличивает значимость трехмерного (3D) моделирования. 3D-модели облегчают планирование, контроль и принятие решений во многих отраслях [2]. Трехмерная фотореалистичная визуализация реальных городов может использоваться во множестве приложений, включая планирование, транспорт, окружающую среду и туризм.

Существующие технологии трёхмерного моделирования и хранения пространственных данных, применяемые в современном программном обеспечении, обладают рядом существенных недостатков, которые не позволяют развиваться порталам виртуальных городов и геоинформационным системам высокими темпами. Высокоэффективная программа для трехмерного моделирования объектов городской среды призвана вывести разработку макета города на новый уровень и обеспечить всеми необходимыми инструментами для быстрого создания, редактирования и хранения 3D-моделей городской застройки.

Актуальность обусловлена практической потребностью создания полнофункциональных высокоэффективных программ для 3D-моделирования макетов городов и необходимостью применения их в геоинформационных системах и геопорталах для решения многих практических, научных и учебных задач.

Цель – определение основных аспектов при разработке программного обеспечения для 3D-моделирования городов. Для достижения поставленной цели необходимо выполнить следующие исследовательские задачи:

– описать основные бизнес-процессы при создании виртуального города;

– проанализировать функциональные возможности программ для трёхмерного моделирования;

– выявить основные проблемы при создании моделей объектов города и определить пути их решения.

Описание основных бизнес-процессов при создании виртуального города

В программной системе для создания виртуального города можно выделить следующие необходимые модули в соответствии с их функциональными возможностями:

– центр моделирования (3D-редактор) — мощная специализированная графическая станция, позволяющая с помощью с помощью простых в использовании инструментов быстро создавать эскизы и текстурировать 3D-модели зданий;

– набор программных модулей для создания интерактивного взаимодействия с виртуальным макетом;

– файловая база геоданных, предоставляющая возможности хранения созданных 3D-моделей в различных форматах;

– WEB-приложение для публикации 3D-сцен в Интернет и обмена ими с общественностью и компаниями, работающими с 3D-моделями.

Центр моделирования представляет собой наиболее сложный многофункциональный комплекс и требует к себе повышенного внимания.

Трехмерная сцена города состоит из модели местности (земной поверхности) и моделей наземных объектов. При создании моделей необходимо решить две важнейшие задачи: конструирование геометрии и текстурирование модели [2].

Распространённую стратегию моделирования 3D-макета города можно представить в виде следующей схемы (рис. 1 [2]).

Рисунок 1 - Стратегия и компоненты 3D-моделирования территории города

Основные этапы моделирования 3D-сцены города [3]:

1. Проектирование карты местности. Для моделирования поверхности используются цифровые модели рельефа, созданные по спутниковым снимкам.

2. Подготовка материала. Определение местоположений оснований зданий, их размеры происходит по ГИС-данным и информации с 2D-карт (фотоснимкам).

3. Моделирование объектов. Происходит по специально разработанной технологии.

4. Текстурирование. На основе собранных фотоматериалов и библиотеки текстур создается UVM – развертка модели здания с помощью 3D- и графического редакторов.

5. Экспортирование моделей в среду графического процессора. Разработанная 3D-модель конкретного архитектурного сооружения преобразуется в файл определённого формата для экспорта на 3D-карту.

6. Сверка и корректура расположения моделей зданий на карте. Требуется как сверка наличия объектов в определенном квадрате карты, так и сверка по масштабу и точному расположению зданий в 3D-пространстве.

7. Тестирование. Производится визуальная оценка 3D-сцены и измеряется производительность карты.

Анализ функциональных возможностей программ для трёхмерного моделирования

Существует множество технологий, применяемых для моделирования объектов городской среды. Охватить все предлагаемые методики не представляется возможным, однако их можно попробовать сгруппировать по степени автоматизации основных процессов:

– ручное создание моделей в программах трехмерного моделирования;

– автоматическая генерация 3D-моделей;

– полуавтоматическое создание 3D-моделей. [4]

1. Рассмотрим принцип построения моделей по первой из вышеперечисленных технологий. В программах подобного типа моделирование геометрии и текстурирование моделей проводятся вручную. Модели создаются для каждого типа строений и затем множатся нужное количество раз при размещении на карте. Текстурирование обычно выполняется наземными фотоснимками и изображениями из библиотек текстур [4].

Одной из программ подобного типа является Google SketchUp. 3D-модель здания, разработанная в данном продукте, представлена на рис. 2.

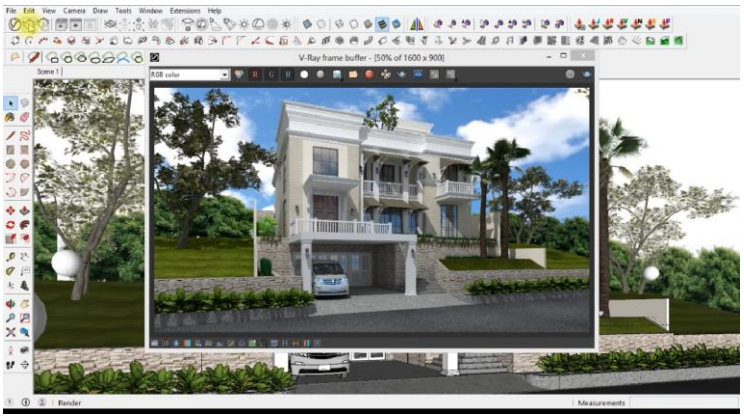


Рисунок 2 – 3D-модель здания в Google SketchUP

Модели, разрабатываемые в Google SketchUP, можно создавать с высокой детализацией, однако недостаточно реалистично текстурированными и метрически точными. Также значительным недостатком является высокая трудоёмкость создания модели.

2. Рассмотрим наиболее молодую и перспективную автоматическую технологию создания 3D-моделей. Она использует алгоритмы восстановления геометрической формы объектов по их стереоизображениям. Стереоизображения получают с самолета, для этого используют наклонные цифровые камеры. Эти же изображения используются как источник текстур фасадов зданий. Для уточнения геометрии зданий и получения модели рельефа может использоваться воздушный лазерный сканер [4].

Примером данной технологии является цифровая модель Берлина, которую можно скачивать по кусочкам для локальной работы. Около 550 000 смоделированных зданий представлены на карте, однако центр и отдельные знаковые модели проработаны хорошо, а остальные объекты, созданные автоматически, являются недостаточно фотореалистичными или отсутствуют вовсе. Модель квартала Берлина представлена на рис. 3.

Рисунок 3 – 3D-модель Берлина

Процесс автоматического построения модели имеет ряд недостатков. При моделировании объектов таким способом возникают многочисленные погрешности в геометрической точности модели и её текстуре. Фигуры зданий не точно воспроизводятся, стены могут быть наклоненными, а размеры объектов - искажёнными. При текстурировании модели посторонние объекты часто остаются на стенах зданий. Таким образом, виртуальная сцена становится недостаточно детализированной: архитектурные сооружения, деревья и рельеф местности сливаются в одну сплошную поверхность.

3. Третья методика моделирования - полуавтоматическое создание 3D-моделей. Оно исключает некоторые слабые стороны автоматического процесса генерации моделей городов. Геометрические модели зданий здесь создаются операторами в основном по аэроснимкам. Этот подход применяется в CyberCity-Modeler.

Моделирование в данном продукте происходит на полуавтоматических фотограмметрических станциях. Общая концепция технологии, применяемой в CyberCity-Modeler, представлена на рис. 4 [5].

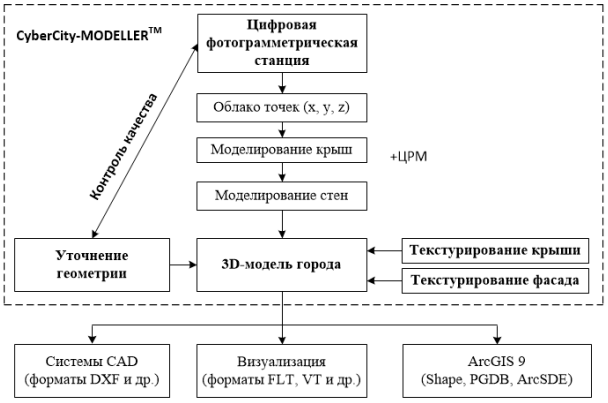


Рисунок 4 - Технология обработки данных в программе CyberCity-Modeller

Исходные данные вытягиваются из данных аэрофото- и космических снимков, а также данных лазерного сканирования. Специальный алгоритм получает из данных координаты основных точек поверхности крыши, стен. Затем станция импортирует их в виде облака точек и автоматически вносит в модель. Чем проще тип элемента здания, тем меньшее количество точек необходимо для его моделирования, что увеличивает производительность. Неточности, возникающие в процессе оцифровки, можно редактировать с помощью специальных инструментов. Текстурирование производится автоматически одним

из трёх способов (унифицированным, автоматическим или наземным) в зависимости от качества исходных данных и требуемого уровня результирующего изображения. При текстурировании также используются аэрофото-, орто- или космические снимки [5].

Модели CyberCity 3D высокого разрешения раздаются во многих распространённых 3D-форматах для последующей работы с моделью в прочем программном обеспечении (см. рис. 5).

Рисунок 5 – 3D-модель города Майами-Бич, штат Флорида, построенная в CyberCity

Слабое место в подходе, применяемом в CyberCity, - это низкое качество текстур. По сравнению с наземной фотосъёмкой аэроснимки, из которых извлекаются текстуры, имеют низкое разрешение. Если изображений недостаточно, то некоторые стороны объектов могут оказаться и вовсе без текстурирования.

4. Отдельно стоит рассмотреть уникальное приложение для трёхмерного моделирования - Esri CityEngine.

CityEngine применяет процедурный подход моделирования, который позволяет эффективно создавать детальные крупномасштабные 3D-модели городов несколькими щелчками мышки, вместо того чтобы вручную создавать каждый объект отдельно. В процедурном моделировании 3D-объекты и текстур создаются при помощи правил пространственного моделирования вместо трудоемкого ручного моделирования. Простое процедурное правило может применяться для создания множества 3D-моделей одновременно. Например, правило может использовать атрибутивную информацию ГИС-объекта (тип крыши, количество этажей, материал стен и т.д.) для создания серии 3D-моделей, максимально точно передающих характеристики каждого объекта. Чем больше будет количество атрибутов, тем точнее будет созданная модель.

Демонстрационная 3D веб-сцена городского квартала Филадельфии, разработанная с помощью CityEngine и просмотренная в веб-вьюере CityEngine, представлен на рис. 6.

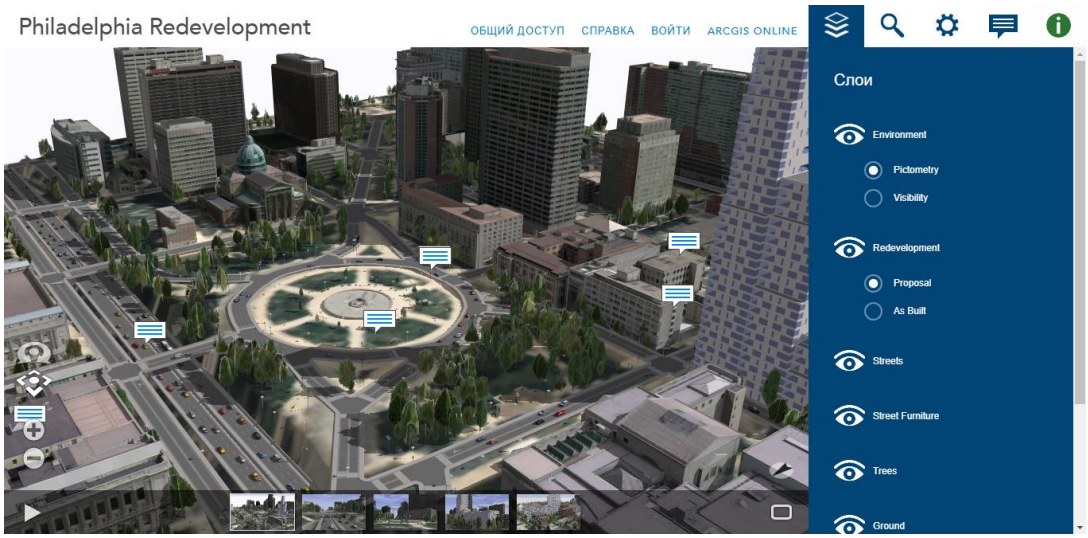


Рисунок 6 – Виртуальная модель квартала Филадельфии в CityEngine

Основным недостатком CityEngine является то, что данное ПО не предлагает решений для автоматического построения 3D-городов по перспективным снимкам или лидарным данным (облаку точек), что ускорило бы моделирование объектов.

Проблематика создания трёхмерной модели города и пути решения

После анализа возможностей и технологий существующих программ для трёхмерного моделирования было выявлено ряд проблем.

1. Наиболее существенной проблемой создания 3D-макета города является низкая скорость его моделирования. Ручное моделирование объектов городской среды повышает трудоёмкость и стоимость ПО.

Решение состоит в том, чтобы при создании зданий использовать типовые шаблоны. В программе должна быть предусмотрена обширная база типовых крыш, стен, окон и других элементов фасада зданий и объектов городского окружения. Это обеспечит создание массивной комплексной настройки архитектурного стиля и особенностей города. В итоге на одно типовое здание оператор будет тратить значительно меньше времени.

2. Следующей проблемой, затрагивающей удобство пользователей, является геометрическая точность и фотореалистичность 3D-моделей. Восстановление формы объектов по их фотоснимкам или данным лазерного сканирования часто происходит с ошибками: искажается форма и размеры зданий, посторонние объекты остаются на текстурах зданий либо объекты не текстурированы вовсе.

Успешное решение заключается в проведении измерений стереоскопическим методом, при котором точность измерений зависит от геометрической точности исходных аэроснимков. Автоматическое текстурирование будет извлекать текстуры из наземных и аэрофото, орто- и космических снимков. Для удаления посторонних элементов на фасадах зданий необходимо добавить специальный модуль обработки, редактирования и ретуширования изображений.

3. В ходе исследования выявлена проблема хранения пространственных данных. Ограничение авторских прав на использование уникальных алгоритмов и форматов хранения данных приводит к возникновению массы проблем с переносом данных в другой формат при обмене ими с другими системами.

Для преодоления данной проблемы в системе необходимо предусмотреть возможность работы без конвертации в основных ведущих форматах графических (ArcView, Shape File, ESRI ArcSDE), табличных (Access, Excel, xBASE) и данных в растровых форматах (GIF, JPEG, TIFF, GEO, PCX). Это облегчит интеграцию с другими ГИС и САПР. При использовании собственного формата данных необходимо добавить возможности подключения данных по стандартам WMS (Web Map Service), WFS (Web Feature Service), OGC (Open GeoSpatial Consortium), а также импорт и экспорт данных в формате GML (Geography Markup Language).

4. Ещё одной проблемой является внутренняя организация хранения данных. Сервер данных собственной разработки часто имеет ограниченный функционал и недостаточно быструю скорость работы с базой данных.

Решением является построение системы по схеме клиент-сервер, которая использует одну из распространённых СУБД (Postgre SQL, Oracle) и их расширения (Spatial PostGIS, Oracle Locator/Spatial), которые увеличат возможности по работе с пространственными данными и повысят быстродействие системы в целом.

5. Следующая проблема состоит в публикации созданных карт в виде WEB. В последнее время особую популярность приобрели облачные технологии. Это влечёт за собой необходимость встраивания набора скриптов для работы с макетом города в Интернет-порталы.

Предлагается следующее решение. Весь функционал по работе с системой необходимо реализовать в виде WEB-приложения, работающего через WEB-браузер. Наиболее преимущественным является использование связки MapServer OpenLayers в качестве модуля для создания WEB-приложений.

Выводы

В статье описаны и проанализированы основные бизнес-процессы создания виртуального города. Выполнен обзор и анализ функциональных возможностей существующих программ для трёхмерного моделирования. Выявлены достоинства и недостатки технологий моделирования в каждом продукте. В ходе исследования выявлены и сформулированы проблемы, возникающие при создании виртуального города, доступного через Интернет, в целом. Предложены решения по устранению проблем скорости создания моделей, их качества, хранения пространственных данных, публикации разработанного 3D-макета.

Литература

1. Коростылёв Р.И., Еремин И.Е. Электронная карта с использованием реалистичных 3D-моделей зданий // Ученые заметки ТОГУ. – 2013. – Т. 4, № 3. – с. 67 – 71.

2. 1. Гречищев, А. Трехмерное моделирование и фотореалистичная визуализация городских территорий / А. Гречищев, В. Бараниченко, С. Монастырев, А. Шпильман // ArcReview. – 2003. – №2 – С.12-13.

3. Дубинин М.В., Мишаченко К.Г., Еремин И.Е. Реалистичная модель городского пространства // Ученые заметки ТОГУ. – 2014. – Т. 5, № 4. – с. 1379-1384.

4. Основные стратегии создания 3D-моделей городов [Электронный ресурс]/ Александр Бондарец. – 2010. – Режим доступа: http://gis-lab.info/qa/3dcities.html.

5. Ulm K., 2003 Reality-based 3D city models with CyberCity-Modeler (CCmodeler) and laserscanner data. VI Conference on Optical 3D Measurement Techniques – Gruen/Kahmen (Eds), 2003. – p.32-39.

Мамутова В.А., Григорьев А.В. Проблематика создания трёхмерных моделей городов и анализ рынка приложений для 3D-моделирования. *В статье описаны ключевые моменты по созданию виртуального города, доступного через Интернет. Выполнен обзор приложений для трёхмерного моделирования и планирования городской среды. Проанализированы проблемы, возникающие при трёхмерном моделировании объектов городской застройки, определены возможные пути их решения.*

Ключевые слова: *виртуальный город, трёхмерное моделирование, 3D-модель городской среды, текстурирование, аэрокосмические снимки, хранение пространственных данных.*

Mamutova V.A., Grigoriev A.V. The problem of creating three-dimensional models of cities and analysis of the market of software programs for 3D modeling. *The article describes the key points arising from creating a virtual city, which is accessible via the Internet. The article provides a brief overview of applications for three-dimensional modeling and planning of the urban environment. The author analyses problems arising from the three-dimensional modeling of objects and proposes possible ways to solve it.*

Keywords: *virtual city, three-dimensional modeling, 3D-model of the urban environment, texturing, aerospace images, spatial data storage.*

УДК 004.92

Обзор и сравнение программного обеспечения трехмерного моделирования

Назарко А.В., Григорьев А.В.
Донецкий национальный технический университет
nazarko.anna@list.ru , grigorievalvl@gmail.com

Назарко А.В., Григорьев А.В. Обзор и сравнение программного обеспечения трехмерного моделирования. *В статье описаны распространенные программы, с помощью которых можно создать трехмерное изображение. Продемонстрированы основные сходства и различия между ними, проанализированы их характеристики и выделено наилучшее ПО.*

Ключевые слова: *трехмерное моделирование, компьютерная графика, программное обеспечение, визуализатор, функциональность, разработка.*

Введение

Актуальность работы: Высокоскоростное становление технологий за настолько краткое время привело к чрезвычайному прогрессу в области программного обеспечения и компьютерной техники. Обыденными стали и зрительные эффекты, которые на нынешний день доступны всякому из-за широкого распространения программ для компьютерной графики и, в частности, трехмерного моделирования [1].

Цель работы: выполнить обзор и сравнение программного обеспечения трехмерного моделирования, определить перспективы направления развития данного класса программного обеспечения.

Обзор программного обеспечения

Выполним обзор таких наиболее популярных программных решений как: Autodesk 3D Max, Blender 3D, Autodesk Maya и FaceGen Modeller.

1. Autodesk 3D Max - эта программа очень популярна, в большинстве это благодаря тому, что программа нацелена на визуализацию архитектуры. В данном ПО есть модули, которые важны при разработке всевозможных строительных планов – от обычных прототипов дверей и окошек, до природных явлений, а также строительных инструментов.

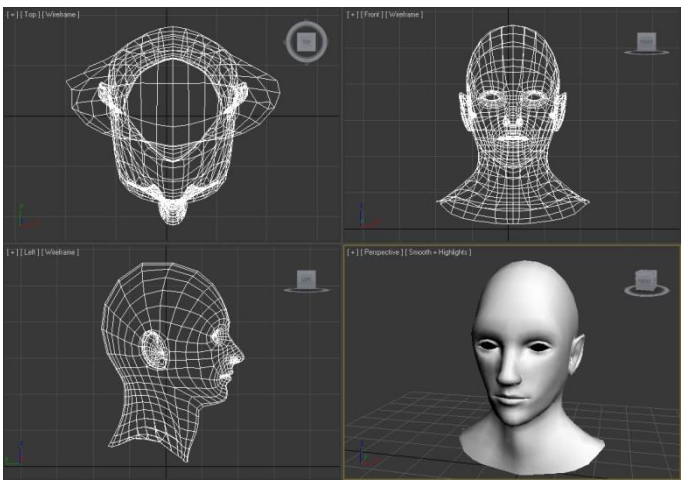


Рисунок 1 – Интерфейс пакета 3D Max

Помимо всего, в данном ПО есть опции для настройки освещения 3D сцены. Ко всему прочему в программу включен фотореалистичный визуализатор, позволяющий достичь очень реалистичного результата. Недостатки компенсируется наличием внушающего числа плагинов, наращивающих интегрированные способности пакета. К примеру, модуль Afterburn используется для моделирования близких к реальности взрывов [2].

2. Blender 3D – умелое ПО в свободном доступе для работы с трёхмерными изображениями, содержащее анимационные способы, а также такие инструменты как: рендеринг, постобработка и монтаж видео со звуком, сборки с поддержкой «узлов» и поддерживает разработку трехмерных игр. В реальное время

очень известен между бесплатными 3D-редакторами благодаря его скорым и умеренным развитием, которому содействует компетентный состав создателей. Не смотря на свободный доступ ПО не уступает иным платным пакетам.

Рисунок 2 – Интерфейс пакета Blender 3D

Одной из причин для конкурентоспособности является то, что модифицировать редактор может обычный пользователь. Многие из модулей и параметров, появившиеся в Blender, были добавлены абсолютно различными пользователями, работавшими над разнообразными функциями для решения определенных задач.

3. Autodesk Maya – в течение длительного периода времени данное программное обеспечение противопоставляется главному сопернику на рынке пакетов для работы с трехмерной графикой – 3D Max. Искусными 3D-модельерами этот продукт применяется почаще других. Данная программа нередко применяется популярными во всем мире мультипликационными студиями Dreamworks, Walt Disney и др.

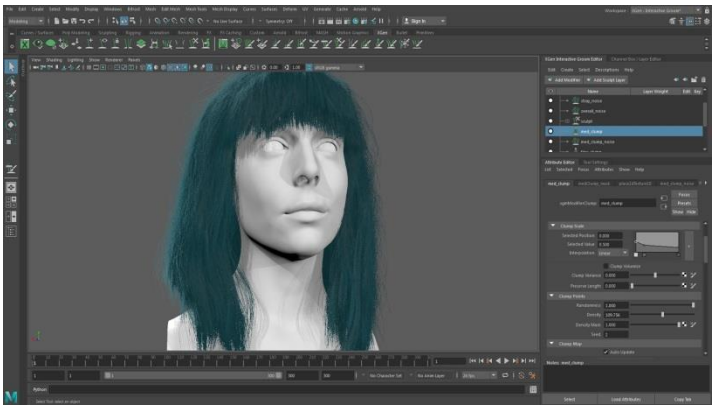


Рисунок 3 – Интерфейс пакета Autodesk Maya

4. FaceGen Modeller – считается бесплатным, быстроразвивающимся ПО, не уступает иным платным пакетам. Базирующаяся в 1998 году, продолжающая исследовать и разрабатывать технологии для автоматического сотворения 3D внешности. Фирма лицензировала FaceGen для множества фирм и институтов, охватывая Electronic Arts (Game Face), Sony, Microsoft и Sega. Различные компании применяют ПО для различных нужд, а именно таких как: трехмерные игры, VR миры и VR реальность, online-продажа одежды, чаты и пакеты для видеосвязи, моделирования в области психологии и графики, 3D-печать, 3D-моделирование и образовательное ПО, обучение и оценка алгоритмов распознавания лиц, комбинированное ПО для моделирования внешности [3].

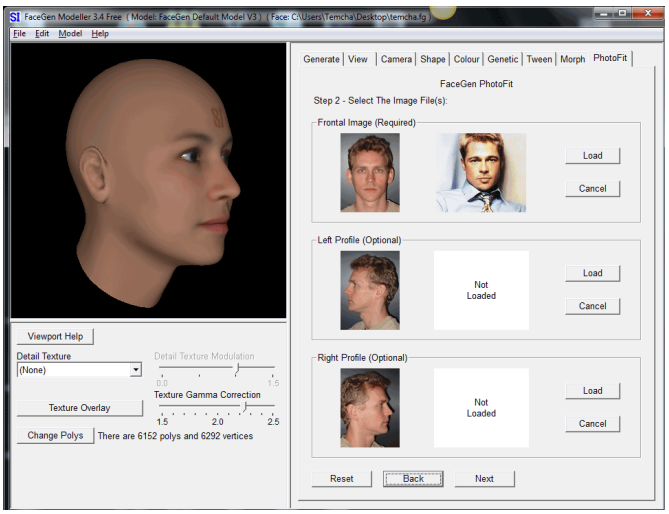


Рисунок 4 – Интерфейс пакета FaceGen Modeller

Сравнение программного обеспечения

Изучив информацию о вышеуказанных 4 пакетах трехмерного моделирования и взяв за основу методику сравнения из указанной статьи[1] и дополнив её можно составить сводную таблицу их основных характеристик, а также достоинств и недостатков каждого программного обеспечения.

Таблица 1 – Характеристики пакетов 3D моделирования

	Autodesk 3Ds Max	Blender 3D	Autodesk Maya	FaceGen Modeller
Отрасль использования	Визуализация -игры	Визуализация – режим реального времени	Игры - кино	Визуализация - игры
Цена полного пакета продукта	5000 €	Бесплатно	2500 €	1395$
Операционная система	Windows	Windows, Mac OS, Linux	Windows, Mac OS, Linux	Windows, Mac OS, Linux
Руководство пользователя	-	-	+	+
Популярность производителя	+	-	+	-
Документация	+	+	+	-
Видео обучение	+	+	+	+
Поддержка популярных форматов импорта / экспорта				
3DS	+	+	+	+
COLLADA	+	+	+	+
FBX	+	+	+	+
STL	+	+	+	+
Рендеринг				
Рендеринг	Internal, mental ray	Internal	Internal, mental ray	Internal, mental ray
Качество	+	+	+	+
Сферы применения				
Визуализация – дизайн	+	+	+	+
Фильмы	+	-	+	+
Визуальные эффекты – motion эффекты	+	+	+	+

Игры	+	-	+	+
Web-дизайн	-	+	-	-
3D в реальном времени / виртуальная реальность	+	+	+	+
Отзывы пользователей и известность пакета				
Америка	+	+	+	+
Европа	+	+	+	+
Азия	+	-	+	+

Заключение

В данной работе выполнены обзор и сравнение различных пакетов программного обеспечения трехмерного моделирования. Сравнив эти пакеты, можно сделать вывод, что, лучшим из четырех редакторов, несмотря на то, что она не находится в свободном доступе, является Autodesk Maya. Благодаря своим уникальным возможностям и легкости в обучении этот пакет на сегодняшний день имеет наибольшее количество положительных отзывов, как среди новичков, так и среди уверенных пользователей. Но не смотря на ее лидирующую позицию, со стремительным прогрессом различных характеристик 3D-изображений, в скором времени понадобятся уже новые средства и модификации для их обработки, а значит развитие данного класса программ очень актуально. Остается все меньше сфер деятельности человека, в которых не используется трехмерная графика. Она активно применяется при разработке игр и графики в фильмах, в проектировании и строительстве, в различных науках и всевозможных других отраслях.

Литература

1. Зенг В.А. ОБЗОР ПРОГРАММ 3D-МОДЕЛИРОВАНИЯ // Научное сообщество студентов XXI столетия. ТЕХНИЧЕСКИЕ НАУКИ: сб. ст. по мат. XXXVII междунар. студ. науч.-практ. конф. № 10(36). URL: http://sibac.info/archive/technic/10(36).pdf

2. Бондаренко С.В., Бондаренко М. Ю. 3ds max. Легкий старт. – СПб.: Питер, 2013. – 128 с.: ил.

3. Онлайн обучение для программы FaceGen Modeller // facegen.com [Электронный ресурс] – Режим доступа: https://facegen.com/support.htm.

Назарко А.В., Григорьев А.В. Обзор и сравнение программного обеспечения трехмерного моделирования. *В статье описаны распространенные программы, с помощью которых можно создать трехмерное изображение. Продемонстрированы основные сходства и различия между ними, проанализированы их характеристики и выделено наилучшее ПО.*

Ключевые слова: *трехмерное моделирование, компьютерная графика, программное обеспечение, визуализатор, функциональность, разработка.*

Nazarko Anna, Grigoriev Alexander. Review and comparison of 3D modeling software. *The article describes common programs with which you can create a three-dimensional image. The main similarities and differences between them are demonstrated, their characteristics are analyzed and the best software is highlighted..*

Key words: *three-dimensional modeling, computer graphics, software, visualizer, functionality, development.*

УДК 004.9

Образовательная социальная сеть как современный подход к обучению

Пыльцов Д.А., Григорьев А.В.
Донецкий национальный технический университет
dmitrypyltsov@yandex.ru, grigorievalvl@gmail.com

Пыльцов Д.А., Григорьев А.В. Образовательная социальная сеть как современный подход к обучению. *В статье рассмотрены основные этапы, методы образовательного процесса и подходы к нему. Изучены существующие системы, использующие классические и современные методики обучения, рассмотрены их преимущества и недостатки, проведено сравнение, а также проанализированы основные особенности проектирования образовательной социальной сети, способной используя лучшие практики предоставить удобную платформу для обучения.*

Ключевые слова: *образовательная социальная сеть, проектирование, обучение, современные подходы, анализ особенностей.*

Введение

Получение новых знаний и актуализация имеющихся являются важными для человека в любом возрасте. Учитывая специфику существующей очной системы образования, заключающейся в необходимости присутствовать на занятиях с фиксированным графиком, а также в изучении дисциплин, которые развиваются и требуют обновления полученных знаний, проблема использования новых подходов является одной из самых важных.

На данный момент в сфере образования присутствует тенденция использования методов дистанционного обучения. С учетом этого одним из возможных решений проблемы можно представить использование онлайн-ресурса, содержащего актуальную информацию по различным отраслям человеческой деятельности и позволяющего получать эти знания в любое удобное время в единой системе с возможностью коммуникации с экспертами для решения возникающих вопросов, а также с другими обучающимися для обмена опытом, который затем может быть повторно применим на практике.

Целью написания данной статьи является анализ методов и тенденций в образовательной сфере, изучение основных особенностей существующих систем, а также формирование требований к проектируемой системе.

Анализ предметной области

Для учета в проектируемой системе хороших и исключения неудачных практик, необходимо рассмотреть особенности существующих решений. Данный подход позволяет не допустить критичных ошибок на раннем этапе и, как следствие, в дальнейшем создать более качественный продукт, представляющий из себя достойную конкуренцию существующим системам и методикам.

Перед обзором решений необходимо изучить основные требования к образовательному процессу и его социальной составляющей, что позволит сделать анализ выбранных систем более объективным, а также улучшит видение требований к проектируемой системе.

В любой образовательной среде существуют определенные подходы к получению знаний и их контролю. В общем виде, обучение является процессом передачи опыта. Это значит, что процесс обучения в разных образовательных учреждениях можно считать процессом передачи обществом накопленного опыта подрастающему поколению. Такой опыт включает знания об окружающем нас мире, которые все время совершенствуются, а также способы применения таких знаний в повседневной деятельности человека. В соответствии с этим общество все время изучает мир для развития собственной практической деятельности и окружающей нас действительности [1].

С учетом этого важным фактором в образовательном процессе важной составляющей является получение практического опыта. Также важно и получение теоретических знаний, без которых практика будет невозможной.

Помимо получения знаний, образование несет в себе также и социальную значимость, которая при этом является важной составляющей обучения.

В процессе получения новых знаний обучающийся нуждается в обсуждении новых знаний с другими, а также в дальнейшем обмене опытом, при расширенном изучении интересуемой тематики.

Для более продуктивного изучения любой дисциплины обучающийся также должен поддерживать связь с преподавателем, чтобы тот мог ответить на появляющиеся вопросы, которые могут усложнить понимание более сложного материала.

Процесс обучения не ограничивается только получением знаний и опыта, к основным этапам также относятся следующие:

– определение потребностей к обучению;
– формирование и распределение ресурсов;
– выбор методов;
– составление учебных планов и программ;
– реализация учебных планов и программ.

Рисунок 1 – Процесс обучения

Рассмотрим выделенные этапы на примере классического подходе к обучению, как одного из существующих методов.

На данный момент студенты получают теоретический материал во время прослушивания лекций или изучения методических пособий. После этого, полученные знания закрепляются на практике в процессе выполнения практических работ, решения дополнительных заданий или при обсуждениях на семинарах.

Потребность к обучению в данном случае определяется в интересе студента к изучаемой специальности и желании дальнейшего развития в выбранной сфере, а также в получении соответствующего документа, предоставляющего больше возможностей при трудоустройстве.

Формирование и распределение ресурсов существенно с обеих сторон, как со стороны студента, так и со стороны образовательного учреждения. В данном случае ресурсы можно рассматривать не только как деньги, но и как время, место, базовые знания, опыт, а также прочие материальные и нематериальные составляющие.

Составление учебных планов происходит в данном случае без участия обучающихся.

Существуют также более современные, альтернативные подходы к обучению. К ним относятся образовательные порталы, онлайн институты, а также отдельные обучающие курсы.

Обзор существующих решений

В первую очередь рассмотрим обучение в образовательных учреждениях, как устоявшийся подход к получению знаний. Дальнейший обзор будет проводиться для очной формы обучения в общем случае.

Среди особенностей, которые вероятнее относятся к недостаткам, можно выделить фиксированный график посещения занятий, строго установленные сроки сдачи контрольных заданий и проверочных работ, как правило редкое обновление изучаемой информации, все большее введение систем контроля в виде экзаменов, слабая ориентация на дальнейшее использование получаемых знаний, ограничение знаниями преподавателя по определенной дисциплине.

Из положительных особенностей могут быть выделены материальное поощрение в виде стипендии, общение с другими обучающимися, что развивает навыки коллективной работы, возможность участия в олимпиадах для проверки уровня знаний и развития конкурентоспособности.

Вероятнее, данный метод активно используется по причине своей устоявшейся системы и общей ориентации на такой подход. Также, во время приема сотрудников на работу, пройденное обучение в образовательном учреждении иногда является одним из важных критериев отбора.

На смену или в поддержку такому подходу появляются другие методы, ориентированные на онлайн обучения без обязательства присутствия на занятиях.

Такие системы имеют ряд преимуществ перед классическим подходом [2]:

– повышает посещаемость мероприятий за счет онлайн-участников и просмотров записей;

– привлекает тех участников, кто не может присутствовать на занятиях в силу нехватки времени или географической удаленности;

– предоставляет возможность доступа к электронным материалам и видеозаписям после лекции;

– привлекает новых слушателей, предоставляя им возможность в любое свободное время присоединиться к онлайн-занятию;

– обеспечивает доступность и экономичность образования для всех категорий граждан, в том числе социально незащищенных и маломобильных;

– обеспечивает возможность выбора индивидуального содержания обучения, а также его эффективность и результативность;

– дает возможность выбора индивидуального темпа освоения знаний;

– стимулирует самостоятельную познавательную деятельность учащегося.

В данный момент можно выделить несколько лидирующих систем, используемых для онлайн обучения по различным дисциплинам.

Среди основных стоит рассмотреть проект «ИНТУИТ» [3].

«Национальный Открытый Университет «ИНТУИТ» - является образовательным проектом, а также негосударственным образовательным частным учреждением дополнительного профессионального образования, основными целями которого являются предоставление услуг удаленного обучения и свободное распространение различных знаний в сети интернет.

«ИНТУИТ» организует запись видеокурсов и лекций в крупнейших вузах, а также телестудии. База видеозаписей проекта насчитывает несколько тысяч часов лекций известных профессоров и докладов ученых.

Проект сотрудничает с учебными заведениями, многие учебные материалы «ИНТУИТ» используются в учебном процессе большого числа вузов в разных странах, а также является одним из самых популярных образовательных ресурсов и имеет большой потенциал роста.

Данный проект соответствует многим требованиям к системам онлайн-обучения, однако имеет некоторые недостатки. Самыми существенными можно выделить отсутствие полноценной коммуникации с другими обучающимися и преподавателями, а также слабая поддержка представленных курсов и устаревшая информация в некоторых из них. Сам проект появился раньше более современных аналогов и на данный момент практически не развивается и лишь наполняется теоретическим материалом.

Существует более современный аналог – проект «Универсариум».

Обучение в системе построено по принципу изучения последовательных модулей курса.
Длительность курса составляет от 7 до 10 недель в зависимости от его объема и сложности. Каждый модуль курса включает в себя лекцию в формате видео, самостоятельную проверочную работу, домашнее задание и тестирование [4].

Курсы «Универсариума» оцениваются как элементы образовательных дисциплин в областях знаний.

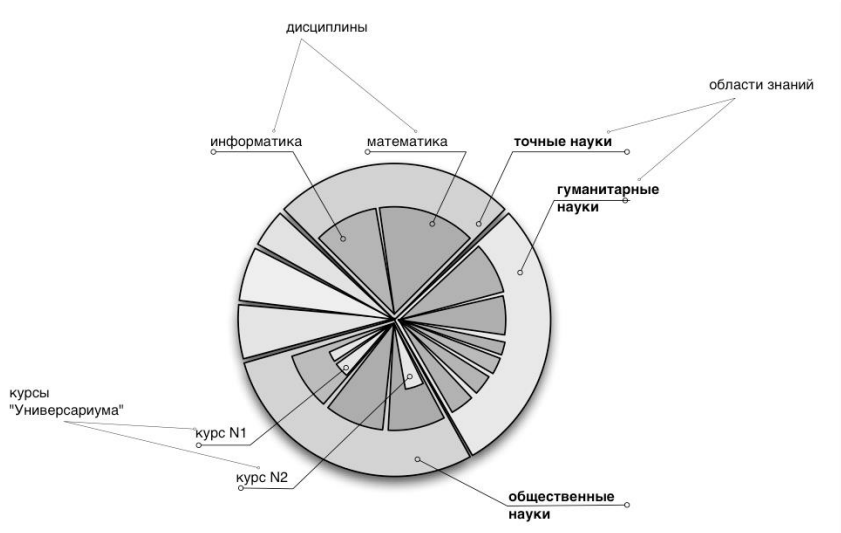


Рисунок 2 – Области знаний, изучаемые в проекте «Универсариум»

«Универсариум» ориентируясь на различные целевые аудитории реализует следующие социальные функции:

− возможность получения дополнительного образования для людей, которые проживают на труднодоступных или далеко расположенных территориях;

− возможности для получения доступного и качественного образования всеми желающими;

− популяризация получения образования;

– возможность получения образования гражданами с ограниченными физическими возможностями.

Из основных недостатков проекта можно выделить лекции в видеоформате, что усложняет актуализацию или корректировку представленного учебного материала. Коммуникация с другими пользователями доступна только на странице отдельного модуля, что неудобно для длинного общения на смежные темы.

Существуют и другие похожие проекты, в том числе и зарубежные, но основные преимущества и недостатки у них в целом похожи.

Также можно рассмотреть узкоспециализированные системы, которые тоже обладают своими особенностями, делающими процесс обучения проще и интереснее.

Как пример такой системы можно рассмотреть портал GeekBrains. На нем представлены учебные материалы по программированию в формате видеолекций, а также вебинаров. Из основных преимуществ можно отметить наличие уникальны заданий для каждого модуля курса, возможность обсуждений на форуме, а также потенциальное трудоустройство от компаний-сотрудников. Из основных недостатков стоит выделить платность курсов и недостаточно развитую систему коммуникации между обучающимися. Помимо этого, все курсы создаются и редактируются основателями портала, а значит их поддержка и корректирование не являются возможными для других пользователей.

Формирование требований к проектируемой системе

Проанализировав положительные и отрицательные стороны существующих решений, а также изучив основные проблемы предметной области, можно сформировать ряд требований, позволяющих составить общее представление о системе, которая позволит улучшить возможности людей в получении и актуализации знаний.

Так как обучение требует также коммуникации и конкуренции с другими обучающимися, в проектируемой системе должны иметься удобные возможности для общения с другими пользователями, как для обмена опытом, так и для обсуждения существующих или разрабатываемых образовательных материалов, а также средства для поддержания стимула на основе конкуренции. Рассмотрим, какие из них должны присутствовать в системе:

– личные и групповые сообщения между пользователями;

– обсуждения, связанные с отдельным образовательным материалом;

– обсуждения в рамках разработки проекта;

– публикации собственных работ для конкуренции с работами других пользователями.

С учетом важности актуализации существующей информации, система должна поддерживать удобные инструменты для обновления существующих материалов, при том не только своих, но созданных другими пользователями. Среди основных инструментов для актуализации можно выделить следующие:

– редактирование собственного образовательного материала;

– редактирование образовательного материала другого пользователя с дальнейшим подтверждением от него или, спустя определенное время, от модератора;

– возможность оставлять отзыв к работе другого пользователя для его мотивации к дополнению;

– средства для обозначения ошибок (как смысловых, так и грамматических) в существующих образовательных материалах.

Для качественного запоминания изученной информации в системе должны присутствовать практические задания для проверки знаний и получения дополнительного опыта. К средствам закрепления материала и получения опыта можно отнести следующие:

– отдельные задания для образовательного материала в виде практической работы, тестирования или решения определенных задач;

– глобальные задания по отдельным разделам для проверки пользователями общего уровня знаний и определения дальнейших планов по изучению;

– индивидуальные задания для особо активных пользователей;

– возможность объединения пользователей в проекты, для реализации поставленной задачи и изучения дополнительного материала с, возможно, его публикацией в системе для изучения другими пользователями.

Из рассмотренных средств отдельно стоит рассмотреть глобальные задания, как один из более важных способов получения практического опыта и проверки уровня знаний. Пользователи, публикующие большое число образовательных материалов в определенном разделе и имеющие достаточно высокий рейтинг в системе, смогут разрабатывать собственные задания, после выполнения которых можно будет оценить уровень отдельных обучающихся и порекомендовать следующий образовательный материал для изучения. Такие задания могут проводиться совместно с компаниями-сотрудниками, с ориентацией задания на поставленную компанией задачу, с возможностью материального вознаграждения лучших работ, что будет являться дополнительным стимулом для обучающихся и возможностью поиска потенциальных сотрудников компаниями.

В системе должны присутствовать страницы компаний-сотрудников, на которых будут отображены основные должности с указанием знаний и навыков, необходимых для трудоустройства. В лучшем случае знания и навыки будут адресованы на соответствующие образовательные материалы в системе. Помимо

перечня навыков и знаний, компании также смогут публиковать собственный набор заданий. Это позволит сделать получение опыта обучающихся более продуктивным.

Для контроля процесса обучения, в системе должны присутствовать инструменты для планирования собственного образовательного процесса и для его контроля. Необходимыми также будут списки выполняемых заданий, с возможностью внесения в каждое из них дополнительной информации в виде ссылок, этапов, схем и прочих данных, позволяющих больше сфокусироваться на самом выполнении и не забыть о деталях.

Из архитектурных решений на данный момент можно выделить только то, что основная версия образовательной социальной сети будет разрабатываться в виде веб-сайта, для возможности использования на любой операционной системе и тем самым предоставляя доступ для большего числа людей. Также среди основных возможностей проектируемой системы необходимо отметить важность наличия мобильного приложения, в котором будет присутствовать возможность локального сохранения выбранных материалов, для дальнейшего изучения без доступа к сети.

Из последних важных свойств системы стоит упомянуть поддержку мультиязычного пользовательского интерфейса и возможность перевода существующих образовательных материалов на разные языки для охвата большей аудитории и популяризации публикуемых обучающих ресурсов.

Выводы

При проектировании системы с учетом сформированных требований, процесс обучения может стать проще, а возможность актуализации имеющихся знаний будет доступнее и экономнее по времени. Все публикуемые материалы смогут поддерживаться сообществом, а значит они практически всегда будут актуальны и корректно оформлены, без лишнего содержания и различных логических ошибок. С учетом возможности коммуникации с другими обучающимися и преподавателями, получение опыта и обмен им с другими будет проще. Также появится дополнительная мотивация для создания коллективных проектов и написания совместных работ.

В данной работе были рассмотрены основные этапы, проходимые в процессе обучения, а также сформированы основные требования к проектируемой системе на основании изученных положительных и отрицательных особенностей существующих систем.

Дальнейшими работами над системой станут проектирование, на основе сформированных требований, а затем разработка и популяризация, которая будет проходить совместно с наполнением системы информацией.

Литература

1. Образование человека. Структура системы образования [Электронный ресурс] – Режим доступа: http://www.grandars.ru/college/psihologiya/obrazovanie.html. - Загл. с экрана.

2. Марчук Н.Ю. Психолого-педагогические особенности дистанционного обучения // Педагогическое образование в России. 2013. № 4.

3. НОУ ИНТУИТ | О проекте [Электронный ресурс] – Режим доступа: https://www.intuit.ru/content/about-project. - Загл. с экрана.

4. О проекте | Универсариум [Электронный ресурс] – Режим доступа: https://universarium.org/project- Загл. с экрана.

Пыльцов Д.А., Григорьев А.В. Образовательная социальная сеть как современный подход к обучению. *В статье рассмотрены основные этапы, методы образовательного процесса и подходы к нему. Изучены существующие системы, использующие классические и современные методики обучения, рассмотрены их преимущества и недостатки, проведено сравнение, а также проанализированы основные особенности проектирования образовательной социальной сети, способной используя лучшие практики предоставить удобную платформу для обучения.*

Ключевые слова: *образовательная социальная сеть, проектирование, обучение, современные подходы, анализ особенностей.*

Pyltsov D.A., Grigoriev A.V. Educational social network as a modern approach to learning. *The article describes the main stages, methods of the educational process and approaches to it. Existing systems using classical and modern teaching methods were studied, their advantages and disadvantages were reviewed, a comparison was made, and the main features of the design of an educational social network were analyzed that could provide a convenient learning platform using best practices.*

Keywords: *educational social network, design, training, modern approaches, analysis of features.*

УДК 004.02, 004.04, 004.82

Анализ инструментальных средств создания музыкальных композиций и перспективы их развития

Семик А.О., Филипишин Д.А., Григорьев А.В.
Донецкий национальный технический университет
domaco@mail.ru, alexander.semik.99@gmail.com, grigorievalvl@gmail.com

Семик А.О., Филипишин Д.А., Григорьев А.В. Анализ инструментальных средств создания музыкальных композиций и перспективы их развития. *В работе предлагается решение задачи автоматизации создания музыкальных композиций. Приведён пример создания музыкальной композиции по всем этапам творческого процесса. Данная работа нацелена на исследование современных технологий и методов, используемых для создания песен и мелодий, а также освящение современных программных продуктов, реализующих эти технологии.*

Ключевые слова: *генератор композиций, нотный редактор, секвенсор, аудио редактор, midi аудио формат данных, vst-инструменты, аудиокодек, аудио фрейм, аудио проигрыватель.*

Введение

Актуальность работы: в последнее время музыкальная индустрия всё больше переносится на персональный компьютер, это относится и как к составлению нотных партитур, так и к созданию песен с нуля, включая даже электронный вокал.

Создание любой качественной музыкальной композиции требует выполнения нескольких этапов, которые в последнее время почти полностью выполняются современными аудио-китами (midi-секвенсорами):

- генерация мотива композиции (черновой вариант);
- создание базовой структуры и синхронизация с текстом, если имеется (можно и без текста);
- дополнение вспомогательных и основных инструментов и запись их партий (сведение);
- доработка аранжировки (наложение вокала);
- мастеринг;
- экспорт в аудио формат и распространение в сети интернет или по друзьям и знакомым;
- монтаж и финальный просчёт с видео сопровождением в качестве одной из аудиодорожек к фильму или фоновой музыки в слайдшоу [1].

Также стоит отметить не вошедший пункт написания аппликатуры для реального исполнения каждым, либо одним из инструментов с учётом возможностей человека, как например, невозможно слишком широко растянуть пальцы на грифе гитары или ударить в три тарелки на барабанах имея всего две руки.

Секвенсоры в наше время могут практически всё, начиная с генерации аккордов налету и заканчивая синтезом новых звуков используя специальные синтезаторы. Такие программы легко переваливают за тысячу долларов в цене, что конечно же оправдывает вложенный труд программистов, но также существуют бесплатные аналоги, уступающие в незначительных частях того или иного этапа создания композиции.

Обратно возможностям любой из программ секвенсоров можно сказать, что ни одна из этих программ не имеет полной автоматизации процесса. Каждый из этапов так или иначе требует вмешательства звукорежиссёра и квалифицированных навыков.

Целью данной работы является выявление необходимости автоматизации музыкального творчества в одном программном комплексе с помощью набора команд.

Проектирование мелодии с помощью MIDI-аранжировщика Band-in-a-Box.

Band-in-a-Box («группа в коробке») — MIDI-аранжировщик, программа, созданная компанией PG Music. По заданным аккордам она автоматически генерирует музыкальное сопровождение и сложные инструментальные соло, имитируя стиль игры известных музыкантов.

Начиная с обновления 2008 года, программа используется ещё и как музыкальный редактор в связи с появлением в программе клавишного редактора, который позволяет пользователю редактировать мелодии проигрывая их на MIDI-инструментах и сохраняя пользовательские изменения. Одно из главных достоинств программы Band-in-a-Box - проста в использовании. Интуитивно понятный интерфейс позволяет легко проектировать музыку любой сложности и любого стиля (см. рис. 1).

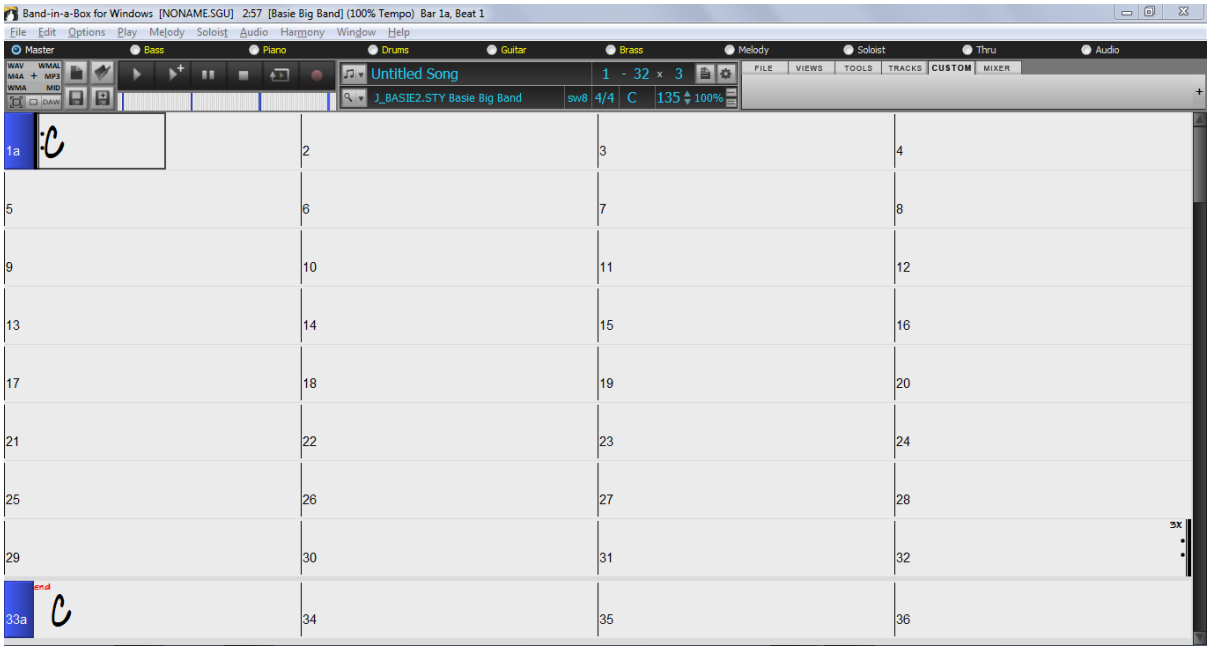


Рисунок 1 – Главное окно программы Band-in-a-Box

Для того, чтобы спроектировать мелодию в программе Band-in-a-Box, нужно определить стиль, в котором будет исполняться композиция. В Band-in-a-Box есть большая база стилей, которую можно использовать при проектировании мелодии. В случае отсутствия нужной мелодии, можно импортировать пользовательские стили (см. рис. 2).

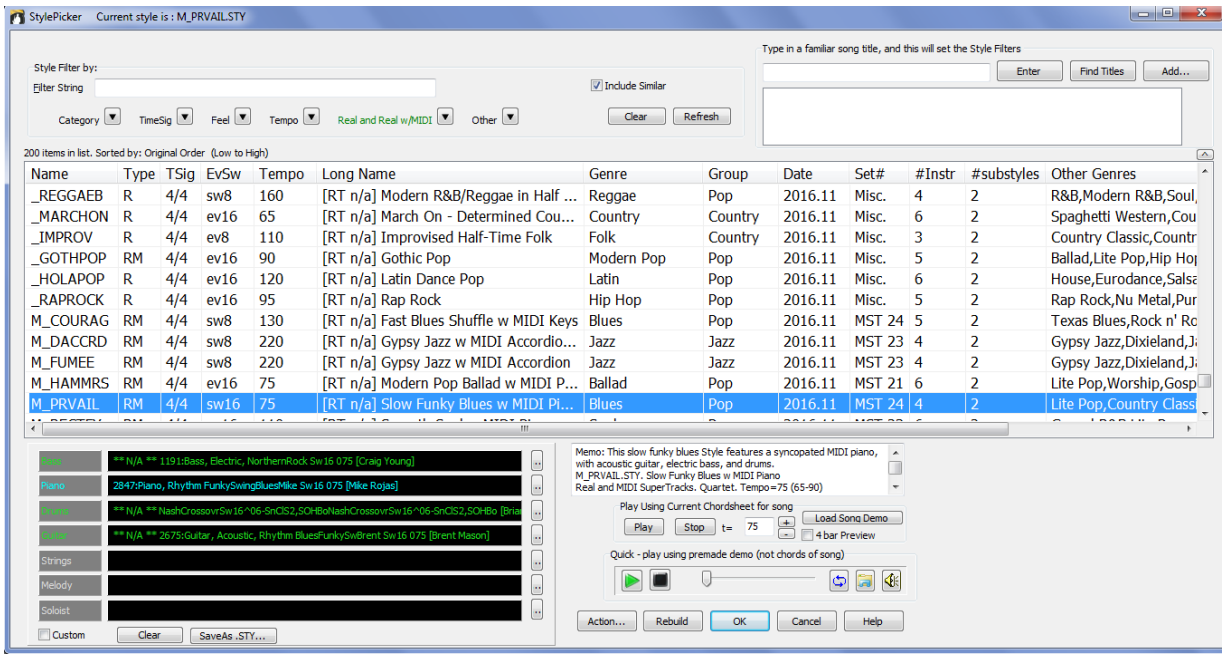


Рисунок 2 – Модуль выбора стиля

После подбора стиля, нужно определить тональность в котором будет исполняться произведение.

После определения тональности, нужно подобрать аккорды. Это можно сделать вручную (ввод с клавиатуры названия аккордов), либо можно использовать модуль «Chord Builder», который предоставляет возможность пользователю предварительно прослушать звучание выбранного аккорда (см. рис. 3).

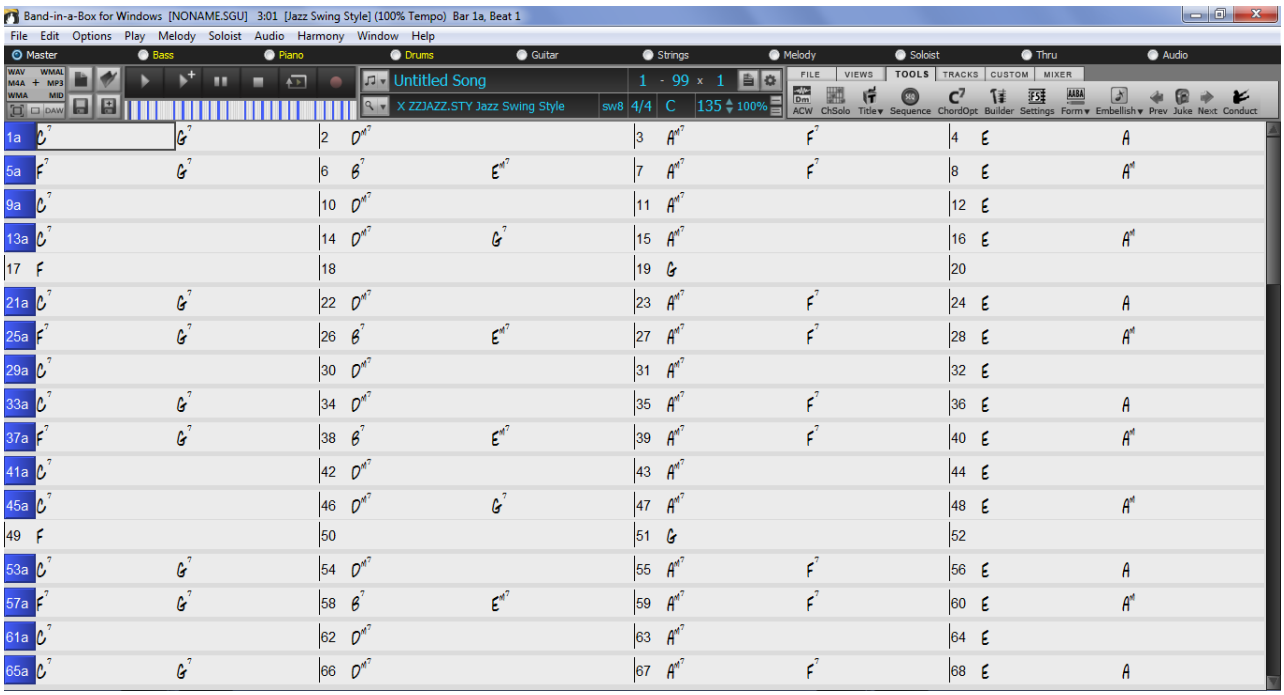


Рисунок 3 – Проектирование композиции в Band-in-a-Box

После проектирования композиции можно экспортировать в один из популярных аудио форматов mp3, wav или экспортировать проект в MIDI-файл, для дальнейшей обработки в других программах. В данной статье проект экспортирован в midi-файл [2].

Обработка инструментов композиции, сведение.

Для дальнейшей обработки инструментов используется midi-секвенсор, в нашем случае это FL Studio. FL Studio - цифровая звуковая рабочая станция (DAW) и секвенсор для написания музыки разработанная компанией Image-Line. Музыка создаётся путём записи и сведения аудио- или MIDI-материала. Готовая композиция может быть записана в формате WAV, FLAC, MP3 или OGG.

Для начала работы с MIDI-файлом, необходимо выполнить его импорт в FL Studio с помощью операции File -> import -> midi file. Программа прочитает файл и вызовет мастер импорта midi-файлов (см. рис. 4).

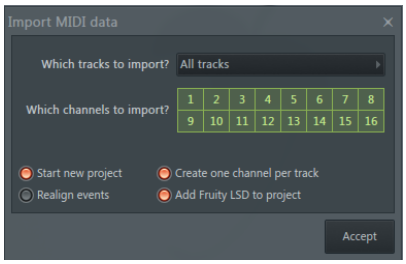


Рисунок 4 – Импорт MIDI-файла

После чего программа запустит ранее созданный проект в первом паттерне, который будет отображён в виде набора автоматически подобранных инструментов (см. рис. 5).

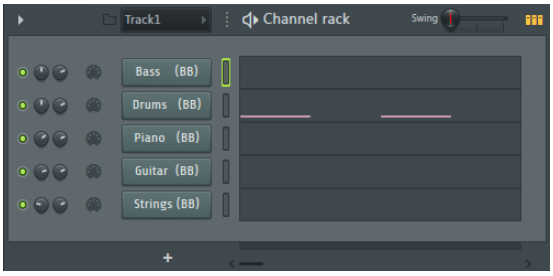


Рисунок 5 – Отображение 1-го канала композиции

Каждый из инструментов представлен программным инструментом и инструкций игры в виде набора нот, которые отображены в midi-редакторе (см. рис. 6) [3].

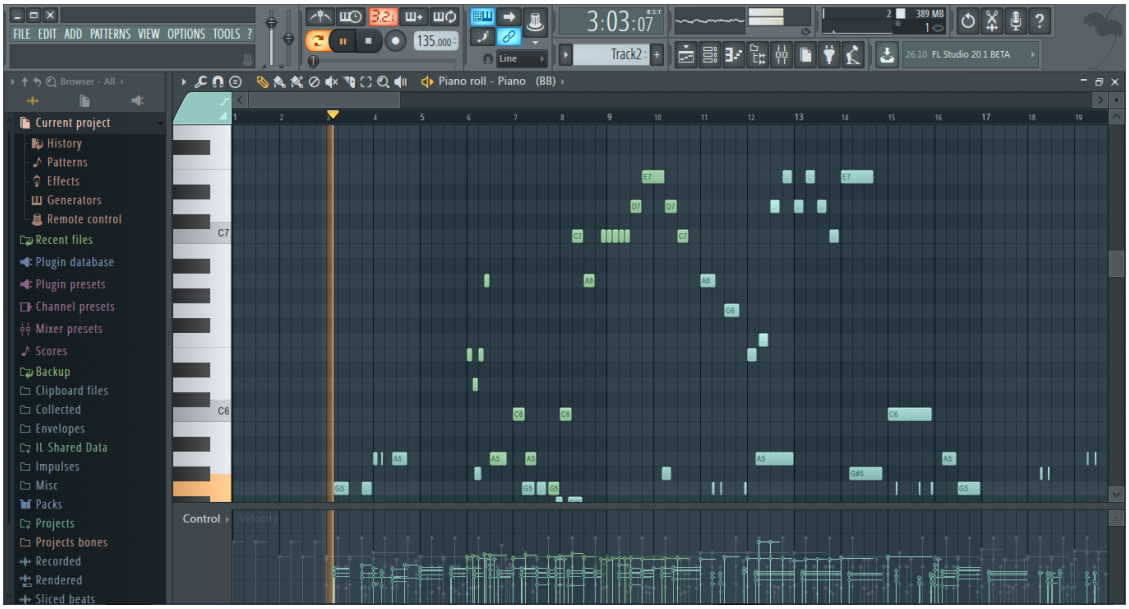


Рисунок 6 – Пример нотной партии инструмента «пианино»

Программа секвенсор позволяет заменять инструменты на собственные (поставляемые с программой) или сторонние (ранее предустановленные в форматах AUX, VST).

В рамках данной программы мы использовались только midi инструменты (рис. 7).

Рисунок 7 – Изменение midi-инструмента

Далее проект дорабатывается в плейлисте, в котором компонуются паттерны. Каждый паттерн можно по-своему настроить, назначить свои эффекты, динамику и фильтры, предварительно распределив по каналам каждую из дорожек. Таким образом можно одновременно один и тот же инструмент с одной и той же партией воспроизводить с разными эффектами, например, гитара и флейта, пианино и синтезированный бас и пр. (см. рис. 8).

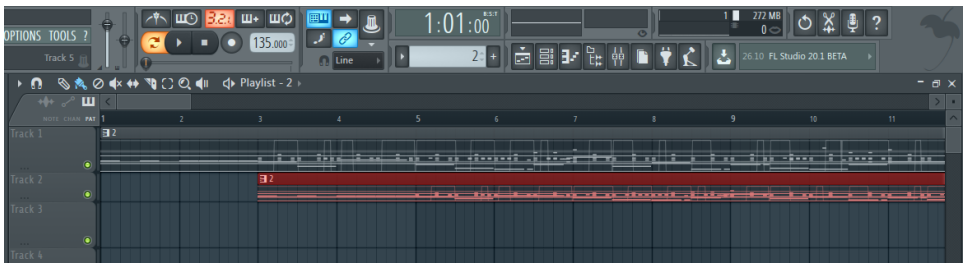


Рисунок 8 – Пример отображения плейлиста для выбранной композиции

Далее проект экспортируется в одном из аудио форматов: wav, mp3, ogg, flac, midi.

Мастеринг композиции

Завершает процесс создания музыкальной композиции мастеринг (пост обработка) полученной композиции, песни или просто мелодии в звуковом редакторе [5, 6].

Мастеринг описывает весь процесс оптимизации звукового файла для конкретной среды, таких как радио, видео, CD или интернет. Перед мастерингом звука следует рассмотреть требования среды, в которой он будет звучать. Если предназначен для интернета, то вероятнее всего файл будет проигрываться через компьютерные колонки, которые плохо воспроизводят низкие звуки. Чтобы это компенсировать, можно поднять низкие частоты во время стадии эквализации [8].

В данной статье была использована программа Adobe Audition [7], в которой с помощью специальных инструментов было почищено звучание от шумов и отрегулирована динамика композиции (см. рис. 9).

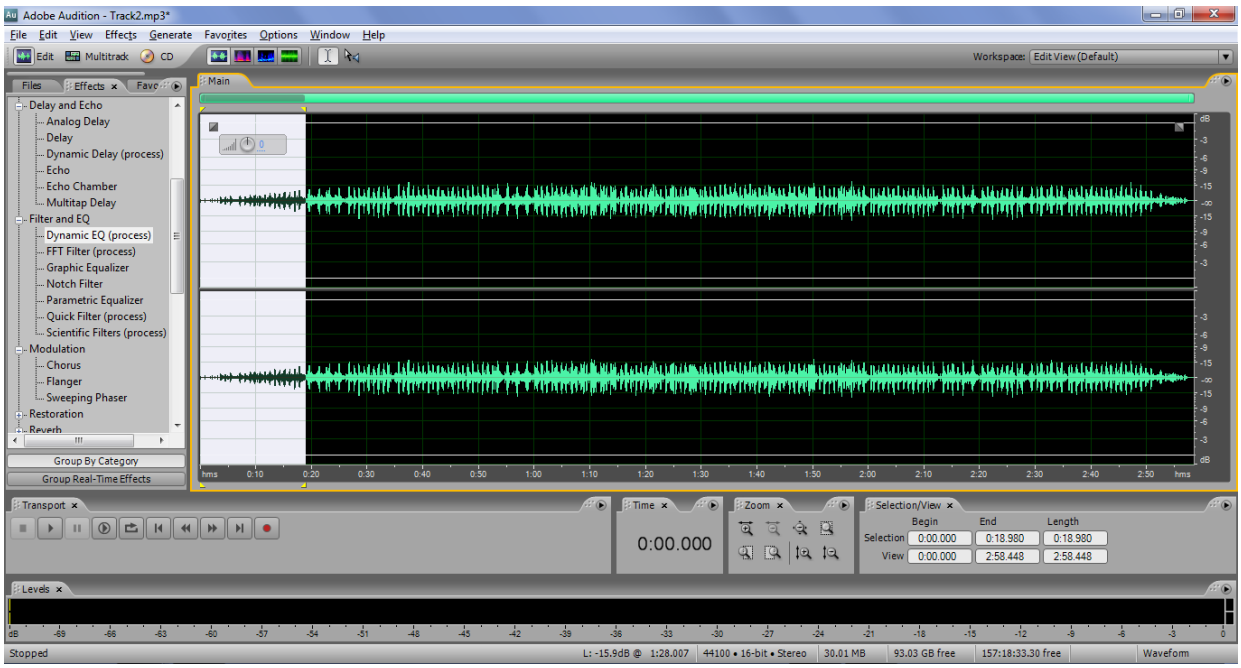


Рисунок 9 − Обработка звука в Adobe Audition

Просмотр и редактирование нотной аппликатуры.

MIDI файлы можно просматривать в виде нотных аппликатур, как например в таких программах, как Finale, Lily Pond, Sibelius, Guitar Pro. Работа в профессиональном редакторе нотных партитур Avid Sibelius будет полезна для пианистов и композиторов, которые должны расписать партии для нескольких инструментов оркестра или группы (см. рис. 10).

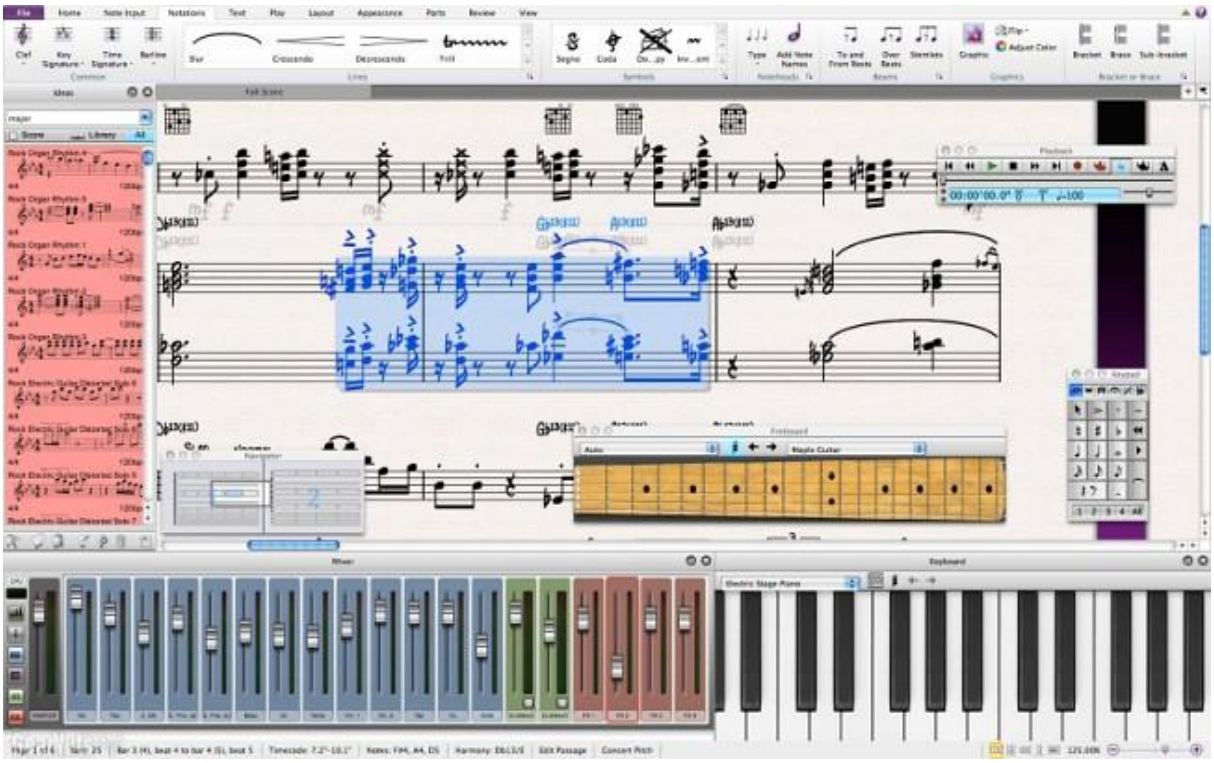


Рисунок 10 – Пример работы в программе Avid Sibelius

Аппликатура для гитаристов чаще всего редактируется в очень удобной программе Guitar Pro. Программа также позволяет редактировать барабаны, синтезаторы, а также настраивать звуки гитар во время исполнения полноценно имитируя реальное исполнение. Можно расставить корректную расстановку пальцев, а также указать точную аппликатуру использованных аккордов (рис. 11).

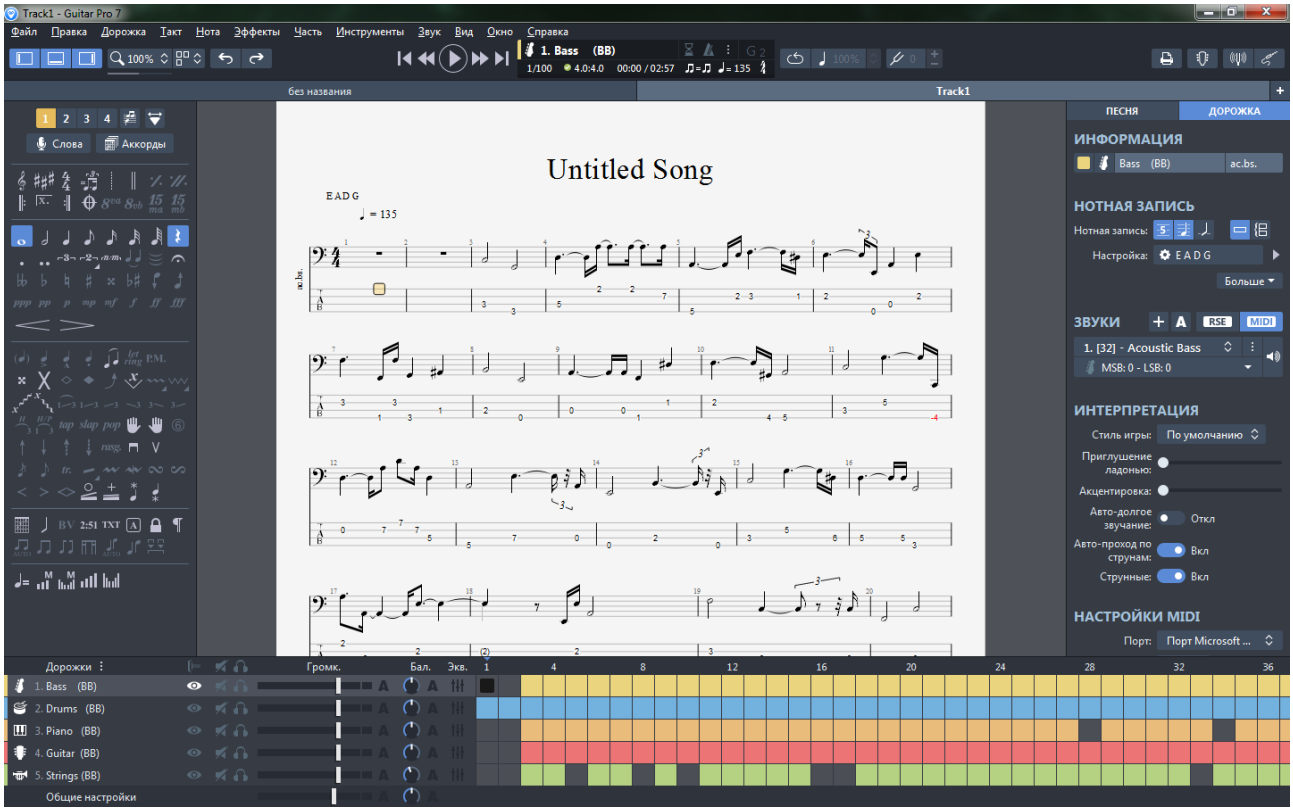


Рисунок 11 – Пример работы в Guitar Pro 7

Выводы

В данной работе проанализирован процесс реализации создания музыкальной композиции с описанием каждого творческого этапа с целью показать, что использование описанных программ на каждом из этапов требует наличия навыков, понимания и умений для квалифицированного выполнения требуемой работы. Следует отметить, что современные программы секвенсоры, такие как Steinberg Cubase совмещают в себе несколько этапов в виде модулей программного комплекса, что также требует наличия знаний использования этой программы и её модулей.

В результате работы был рассмотрен вопрос потребности создания программного комплекса, который позволяет единым набором команд сочинять, сводить, проводить мастеринг и прослушивать музыкальные композиции, то есть полностью автоматизировать весь творческий процесс.

Литература

1. Создание песен от а до я [электронный ресурс] // Энциклопедия курсов для авторов песен в любом стиле: [сайт]. URL: http://songwriting.ru.
2. Форум пользователей Band-in-a-Box на русском языке [электронный ресурс] − http://www.pgmusic.com/forums/ubbthreads.php?ubb=postlist&Board=23.
3. Музыкальное оборудование [электронный ресурс] // MIDI в деталях [сайт]. URL: http://www.muzoborudovanie.ru/articles/midi/midi1.php
4. Статьи, описания и обучение – FL Studio [электронный ресурс] // Русифицированный мануал по FL Studio]. URL: http://fl-studio.ru/publ/ .
5. Основы сведения и мастеринга [информационный портал] // статья для habr. [сайт]. URL: https://habr.com/post/54101/
6. Мастеринг [электронный ресурс] // Свободная энциклопедия Википедия: [сайт]. URL: https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%BD%D0%B3.
7. Руководство по эксплуатации Adobe AUDITION 1.5, 2014, 308 с.
8. Мастеринг [электронный ресурс] // Свободная энциклопедия Википедия: [сайт]. URL: Мастеринг [электронный ресурс] // Свободная энциклопедия Википедия: [сайт]. URL: https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%BD%D0%B3.

Семик А.О., Филипишин Д.А., Григорьев А.В. Анализ инструментальных средств создания музыкальных композиций и перспективы их развития. *В работе предлагается решение задачи автоматизации создания музыкальных композиций. Приведён пример создания музыкальной композиции по всем этапам творческого процесса. Данная работа нацелена на исследование современных технологий и методов, используемых для создания песен и мелодий, а также освящение современных программных продуктов, реализующих эти технологии.*

Ключевые слова: *генератор композиций, нотный редактор, секвенсор, аудио редактор, midi аудио формат данных, vst-инструменты, аудиокодек, аудио фрейм, аудио проигрыватель.*

Semik Alexander, Filipishin Dmitriy, Grigoriev Alexander. Instrumental tools for creating a musical composition and prospects for their development. *The paper proposes a solution to the problem of automating the creation of music. An example of creating a musical composition at all stages of the creative process is given. This work is aimed at the study of modern technologies and methods used to create songs and melodies, as well as the consecration of modern software products that implement these technologies.*

Key words: *composition generator, music editor, sequencer, audio editor, midi audio data format, vst instruments, audio codec, audio frame, audio player.*

УДК 004.94

Анализ функций и перспективы развития САПР трубопроводов

Чернышов Д.Н., Григорьев А.В.
Донецкий национальный технический университет
dima.ch2000@mail.ru, grigorievalvl@gmail.com

Чернышов Д.Н., Григорьев А.В. Анализ функций и перспективы развития САПР. *Рассмотрены понятие и классификация трубопроводов, а также лучшие практики. Выделены несколько требований совладельцев к САПР.*

Ключевые слова: *САПР, трубопровод, классификация, Aveva PDMS, AutoCAD PLANT-4d, требования.*

Введение

Проектирование трубопроводов — важная часть процесса разработки изделий во многих отраслях промышленности, таких как промышленное и нефтегазовое машиностроение, производство оборудования для пищевой промышленности и т.д. При этом решение ключевых задач, в число которых входят пространственная компоновка трубопроводов, подбор необходимой трубопроводной арматуры и создание конструкторской документации, вызывает определенные трудности.

Плотную компоновку трубопроводов сложно представить на листе чертежа. На нем трудно отследить ошибки расположения труб, а если такие ошибки обнаруживаются лишь на стадии производства, то цена их исправления может увеличиться на порядки. Это делает целесообразным использование средств трехмерного проектирования или систем автоматического проектирования.

Кроме того, несовершенным является аппарат функционального моделирования и – комплекс средств прочностных расчетов.

Т.о., по-прежнему актуальной является задача построения новых САПР трубопроводов, имеющих более совершенный инструментарий проектировщика, способный в более высокой степени автоматизировать процесс проектирования.

Целью предлагаемой работы является:

- анализ функциональных возможностей современных САПР трубопроводов, выявление их достоинств и недостатков;
- выявление перспективных направлений развития САПР трубопроводов;
- формирование комплекса требований на новый САПР трубопроводов, лишенный недостатков, но – сохраняющий достоинства современных САПР трубопроводов.

Классификация типов трубопроводов

Трубопровод — искусственное сооружение, предназначенное для транспортировки газообразных и жидких веществ, а также твёрдого топлива и иных твёрдых веществ в виде взвеси под воздействием разницы давлений в поперечных сечениях трубы. Трубопроводы могут защищаться от разрушения из-за превышения давления предохранительными клапанами. С целью защиты от коррозии могут быть покрыты эмалями.

В зависимости от транспортируемой среды, трубопроводы подразделяют:

- аммиакопровод - предназначается для транспортировки аммиака.
- водопровод - предназначен для обеспечения водой населения и промышленности. При этом вода для бытовых и промышленных нужд может различаться по органолептическим свойствам; пригодности для питья, бытовых и промышленных нужд.
- воздухопровод - часто создается в рамках промышленного предприятия для обеспечения производства сжатым воздухом
- газопровод - предназначается для транспортировки попутного нефтяного и природного газа. Стратегические газопроводы предназначаются для передачи на дальние расстояния больших объемов газа - на экспорт и предприятиям, осуществляющим газовый синтез.
- канализация — отвод промышленных и бытовых стоков через систему трубопроводов. Имеет важное санитарное значение. Включает в себя системы очистки стоков, предполагающих возвращение очищенных вод в хозяйственный оборот.
- нефтепровод — предназначается для транспортировки сырой нефти. Нефть при этом подвергается подогреву, препятствующему затвердеванию входящих в ее состав парафинов.

− нефтепродуктопровод (нефтепродуктовод) — транспортировка нефтепродуктов - в том числе бензина и керосина, полученных в результате крекинга. Осуществляется до предприятий, предназначенных для производства нефтепродуктов более высокого передела.

− мазутопровод - трубопровод, осуществляющий транспортировку тяжелых нефтепродуктов, отходов крекинга. Такие продукты могут использоваться в качестве топочного мазута, а также для переработки в дизельное топливо или даже для дальнейшего отделения легких углеводородов.

− паропровод - технологический трубопровод, предназначенный для передачи пара под давлением, используемого для отопления или работы сторонних механизмов.

− пневматическая почта - использование воздуха под давлением для перемещения по трубам физических объектов - чаще всего, стандартизированных капсул с объектами небольшой массы и объема. Используется в рамках одного или близко расположенных зданий, использует механические способы маршрутизации.

− продуктопровод - в общем смысле, трубопровод, предназначенный для транспортировки искусственно синтезированных веществ (в том числе, перечисленных выше), чаще всего - продуктов нефтяного синтеза. В частном случае может означать систему, предназначенную для доставки по трубам любых пригодных для этого объектов.

− теплопровод - трубопровод, предназначенный для передачи тепла (например - в виде нагретого водяного пара).

− этиленопровод - инфраструктура, предназначенная для транспортировки по трубам специфического синтезированного промышленного сырья – этилена [1].

В зависимости от вида прокладки и/или перехода (типа опирания):

− наземный — укладывается выше уровня земли на отдельных опорах;
− надземный;
− арочный;
− висячий;
− балочный;
− подземный — укладывается непосредственно на грунт в траншеях, канавах, насыпях, штольнях, на опорах в тоннелях и дюкерах;
− подводный — укладывается по дну водоёмов, рек или в траншеях, прорытых на дне;
− плавающий — укладывается на поверхности болот, а также озёр, рек и др. водоёмов с креплениями к поплавкам (чаще пластмассовым).

Лучшие практики

Число САПР трубопроводов достаточно велико. Однако, на данный момент лучшими САПР трубопроводов являются пакеты Aveva PDMS и AutoCAD PLANT-4d.

Можно сказать, что именно они определяют уровень развития современных САПР трубопроводов.

Дадим им краткую характеристику, выделим их достоинства и недостатки.

Цель такого анализа – выявление перспективных направлений развития САПР трубопроводов.

Анализ САПР Aveva PDMS

Система PDMS компании AVEVA разработана для проектирования различных промышленных объектов — от небольших проектов, например газокомпрессорных станций, обустройства месторождений до морских нефтяных платформ, предприятий нефтехимии, нефтепереработки, тепловых и атомных электростанций.

В PDMS использована технология graphics from data — это оригинальная и пока непревзойденная технология, которую конкуренты переняли лишь в последнее время и пока еще осваивают. Таким образом, принципиальным отличием системы трехмерного проектирования PDMS от других систем является тот факт, что ядром системы являются не чертежи, а данные.

Проектная документация выпускается автоматически по данным 3D-модели, таким образом, первостепенное значение имеют непосредственно сами проектные решения, реализованные в модели проекта, в то время как их плоскостные проекции — чертежи — являются лишь производными от 3D-модели. Все изменения в проекте вносятся в 3D-модель, при этом автоматически отслеживается увязка вновь принятого решения со всем остальным проектом, а также автоматически вносятся изменения в чертежи (виды, размеры, спецификации, выноски и значки). Благодаря автоматическому ведению статистики рабочих сеансов изменения могут отслеживаться и выделяться на чертежах. Все это позволяет командам проектировщиков и технического персонала выполнять работу с полной уверенностью в полноте и согласованности всех данных. Результатом является эффективная по времени и затратам разработка проекта без коллизий с последующим автоматизированным выпуском проектной документации [2].

В числе основных преимуществ использования PDMS можно отметить:

− интеграцию проектных данных;
− стандартизацию процесса проектирования;

− модульное проектирование, то есть использование данных предыдущих проектов;
− высокое качество и наглядность проектных решений;
− сокращение ошибок в проекте и отзывов со строительной площадки менее чем до 1% за счет функций проверки на коллизии;
− простота внесения изменений в проект, взаимная согласованность данных всех проектных дисциплин;
− инвариантное проектирование;
− сокращение трудозатрат за счет автоматизированного выпуска чертежей;
− сокращение расходов на строительство и эксплуатацию [2].

Основные недостатки PDMS:
− отсутствие прочностных расчетов;
− отсутствие расчетов термодинамики.

Анализ САПР AutoCAD PLANT-4D

AutoCAD PLANT-4D — это система трехмерного проектирования, полностью отвечающая современным требованиям и решающая широкий круг задач: проектирование технологических схем, моделирование нестандартного оборудования, расстановка стандартного и нестандартного оборудования в пространстве трехмерной модели, трехмерная трассировка трубопроводов, выпуск рабочих монтажно-технологических чертежей, автоматическая генерация изометрических чертежей с размерами и спецификациями, автоматическое составление ведомостей, отчетов, спецификаций, заданий смежным отделам и многое-многое другое [3].

Преимущества системы PLANT-4d:
− возможность организации коллективной работы над проектом;
− наглядность;
− ранняя диагностика ошибок и коллизий;
− модульная архитектура системы;
− обширная элементная база;
− удобство в формировании отчетов и документаций;
− по ходу проектирования система позволяет формировать задания смежникам, в том числе и как запросы к базе проекта;

Недостатком системы PLANT-4d является то, что система не может предсказать, как поведет себя среда в трубе и чем это может обернуться.

Вывод из анализа

В качестве общих недостатков перечисленных САПР трубопроводов можно назвать:
− отсутствие прочностных расчетов;
− отсутствие расчетов термодинамики.

Для компенсации данных недостатков САПР трубопроводов на практике используют поход использования независимых пакетов прочностных расчетов.

Недостатки такого пути:
− отсутствует единый комплекс САПР трубопроводов, способный решать весь комплекс задач в САПР;
− нет возможности решить ряд актуальных задач моделирования и проектирования трубопроводов, связанных с рядом специфических областей применения (подземные трубопроводы, трубопроводы в условиях дальнего севера и т.п.).

Т.о., актуальна задача построения нового САПР трубопроводов, лишенного перечисленных недостатков.

Требования к современному САПР трубопроводов

Будем рассматривать выделенные выше недостатки современных САПР трубопроводов как постановку задачи на разработку нового САПР трубопроводов, лишенного названных недостатков, но – сохраняющий все их преимущества.

Перечислим требования к такому САПР трубопроводов:

1. Система должна обеспечить автоматизацию разработки трубопровода на всех уровнях проектирования.
2. Система должна сократить срок проектирования на предприятии.
3. Система должна позволить снизить требования к квалификации проектировщика.
4. Система должна повысить качество проектируемого продукта и конструкторской документации.
5. Система должна уметь проектировать разные типы трубопроводов.
6. Система должна высчитывать передачу тепла между материалами.
7. Система должна позволять создавать теплоизолирующее покрытие.

8. Система должна учитывать передачу тепла через изолированные трубы.
9. При перемещении некоторых материалов (например, сырой нефти) система должна спроектировать оборудование, подогревающее или охлаждающее трубу.
10. Система должна рассчитывать возможное давление грунта на трубу при проектировании подземного трубопровода.
11. Система должна уметь прокладывать трубу через водные пространства.
12. Система должна уметь рассчитывать прочность конструкций.
13. Система должна автоматически создавать опоры для труб.
14. Система должна позволять ставить на трубу предохранительные клапаны.
15. Система должна предупреждать пользователя при отсутствии предохранительного клапана.
16. Система должна иметь возможность развития: САПР представляется как развивающаяся открытая система, в которой предусмотрена возможность замены существующих компонентов и включения новых (расширение элементной базы), генерации на основе заложенной информации новых знаний.
17. Система должна содержать обширную элементную базу.
18. Система должна уметь использовать данные предыдущих проектов.
19. Структура трубы, ее форма и размеры должны описываться с помощью конечного числа параметров. Необходимо предусмотреть ввод пользователем следующих параметров:
    - материал;
    - диаметр трубы;
    - проводимый материал;
    - глубина погружения/высота поднятия;
    - общая длина.
20. Предусмотреть возможность редактирования параметров.
21. Система должна предоставлять возможность получения геометрической модели в трёх измерениях.
22. Необходимо высокое качество графической модели.
23. Система должна по команде показывать трубу в разрезе.
24. Необходимо реализовать возможность поворота, приближения/отдаления камеры, установки её положения при работе с полученными трёхмерными моделями.
25. Система должна реализовывать ландшафт и постройки.
26. Система должна высчитывать температуру окружающей среды.
27. Для заданных параметров трубы должна производиться симуляция проведения материала.
28. Симуляция должна наглядно показывать преимущества и недостатки конструкции.
29. Для отображения результатов симуляции на экран должен выводиться отчет о возможных происшествиях или успешных результатах.
30. В системе должен быть реализован список предупреждений о возможных рисках для конструкции.
31. При изменении любого параметра симуляция должна быть пересчитана.
32. Свойства трубы должны описываться с помощью характеристик.
33. Необходимо предусмотреть возможность создать критическую ситуацию для расчета надежности конструкции.
34. Система должна предоставлять пользователю возможность просмотра справочной информации.
35. Система должна позволять сохранение результатов проектирования (графиков, чертежей, документации) в формате, предназначенном для печати.
36. Система должна обеспечивать автоматическое сохранение с частотой времени, определённой пользователем, а также возможность восстановления несохраненных документов.
37. Система должна иметь удобный, интуитивно понятный пользовательский интерфейс.
38. Для эффективного применения в конструкторско-технологических разработках необходима "подстройка" системы автоматизированного проектирования трубопроводов под конкретные нужды того или иного пользователя. Для такой настройки необходим открытый API-интерфейс. Использование подобных возможностей позволяет повысить эффективность труда на рутинных операциях.
39. При создании трубопровода по частям система должна собирать их в единую систему по команде пользователя.
40. Системные требования для использования системы автоматизированного проектирования трубопроводов:
    - процессор Intel Pentium 3.0 ГГц;
    - Microsoft Windows XP/Vista/7;
    - оперативная память 2 ГБ;
    - свободное место на диске 5 ГБ;
    - видеопамять 128 Мб.

Заключение

В работе выполнен анализ функциональных возможностей современных САПР трубопроводов, на основе выявления их достоинств и недостатков определены перспективные направления развития САПР трубопроводов.

Как главную проблему современных САПР трубопроводов можно определить отсутствие средств полноценного функционального моделирования трубопроводов и – отсутствие средств выполнения прочностных расчетов, адаптированных на различные типы трубопроводов – и различные условия их применения.

Сформирован набор комплекс требований на новый САПР трубопроводов, лишенный недостатков, но – сохраняющий достоинства современных САПР трубопроводов.

Как перспективную работу следует определить:

- развитие данных эскизных требований до уровня полноценного технического задания, построенного исходя из требований IEEE [4];
- построение на основе данного технического задания нового программного комплекса САПР трубопроводов, отвечающий современным требованиям.

Литература

1. Глоссарий-ASCO [Electronic resourse] / Интернет-ресурс. – Режим доступа: http://www.asco.su/Library/Glossary/term.asp?ID=72

2. Журнал «САПР и графика». Раздел «Решения компании AVEVA для проектирования промышленных предприятий» [Electronic resourse] / Интернет-ресурс. – Режим доступа: https://sapr.ru/article/15803

3. Журнал «САПР и графика». Раздел «Немного о PLANT-4D» [Electronic resourse] / Интернет-ресурс. – Режим доступа: https://sapr.ru/article/20013

4. Кобёрн А. Современные методы описания функциональных требований к системам. — М.: Лори, 2002.

Чернышов Д.Н., Григорьев А.В. Анализ функций и перспективы развития САПР. *Рассмотрены понятие и классификация трубопроводов, а также лучшие практики. Выделены несколько требований совладельцев к САПР.*

Ключевые слова: *САПР, трубопровод, классификация, Aveva PDMS, AutoCAD PLANT-4d, требования.*

Chernishov D.N., Grigoriev A.V. Consideration of functional capabilities and development prospects of CAD. *The meaning and the classification of pipelines as well as the best CAD programs are considered. Some co-owner requirements are described.*

Key words: *CAD, pipeline, classification, Aveva PDMS, AutoCAD PLANT-4d, requirements.*

СЕКЦИЯ «ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В МЕДИАИНДУСТРИИ И ДИЗАЙНЕ»

УДК 004.588

Дополненная реальность как концепция создания обучающего приложения

Бобелюк М.Б., Губенко Н.Е.
Донецкий национальный технический университет
bobeluyk.marina@mail.ru

М.Б. Бобелюк, Н.Е. Губенко. Дополненная реальность как концепция создания обучающего приложения. *В докладе рассматриваются актуальные вопросы использования дополненной реальности в образовании. Представлена структура Mobile Augmented Reality Education. Приводится структура и диаграмма состояний разрабатываемого обучающего приложения.*

Ключевые слова: *дополненная реальность, AR, образовательная система, MARE, Mobile Augmented Reality Education, Unity, Vuforia, Android SDK.*

Постановка проблемы

В последние годы перед учениками школ возникла проблема избытка информации, перенасыщение новыми технологиями, а в следствии – дефицит внимания. Школьные учебники и материал, изложенный в них, не вызывают того интереса, как новая мобильная игра или обновление приложения.

Образовательной системе необходимо приспосабливаться к усложняющимся процессам и росту технологий современного мира, где ученики должны пользоваться большим количеством информации и новыми способами ее представления в учебных целях. Такая технология, как «Дополненная реальность» может помочь решить возникающие проблемы.

Цель статьи

Определение «Дополненной реальности», анализ концепции и разработка диаграммы состояний обучающего приложения позволят сделать вывод об эффективности применения изучаемой технологии в образовательных целях.

Определение и характеристика технологии

Дополненная реальность (AR - Augmented Reality) – это новая концепция использования человеко-машинного интерфейса и компьютеров для создания трехмерных виртуальных объектов в реальном мире, с возможностью взаимодействия с ними. Потенциал AR-технологии велик: она способна сделать наш мир более понятным, удобным и интерактивным.

В повседневной жизни человека дополненная реальность преимущественно используется в маркетинговых целях, как информационно-развлекательное средство. Поэтому представляет большой и логичный интерес применения технологии в образовательных целях. Традиционные методы обучения предполагают прослушивание лекций, работу с учебными пособиями в качестве основных способов получения знаний. AR обладает характеристиками, отличающими ее от других средств обучения, недоступными, а иногда даже не возможными в реальном мире.

В большинстве научных дисциплин понимание абстрактной информации обуславливает успешное обучение. Работа с абстрактными моделями предполагает построение мысленных моделей. При этом ученики зачастую испытывают недостаток в прямых аналогиях, помогающих построить такие модели [1]. Дополненная реальность решает эти проблемы.

Кроме этого технологии дополненной реальности позволяют в полной мере использовать то, что человек получает 80% информации из окружающего мира с помощью зрения, при этом люди запоминают 20 % того, что они видят, 40 % того, что они видят и слышат, и 70 % того, что они видят, слышат и делают [2].

Структура MARE

Основные элементы в образовании с дополненной реальностью отображены в структуре методологии MARE (Mobile Augmented Reality Education) (рис. 1). Она была разработана для AR-приложений сферы здравоохранения, но применима и для других областей знаний [3].

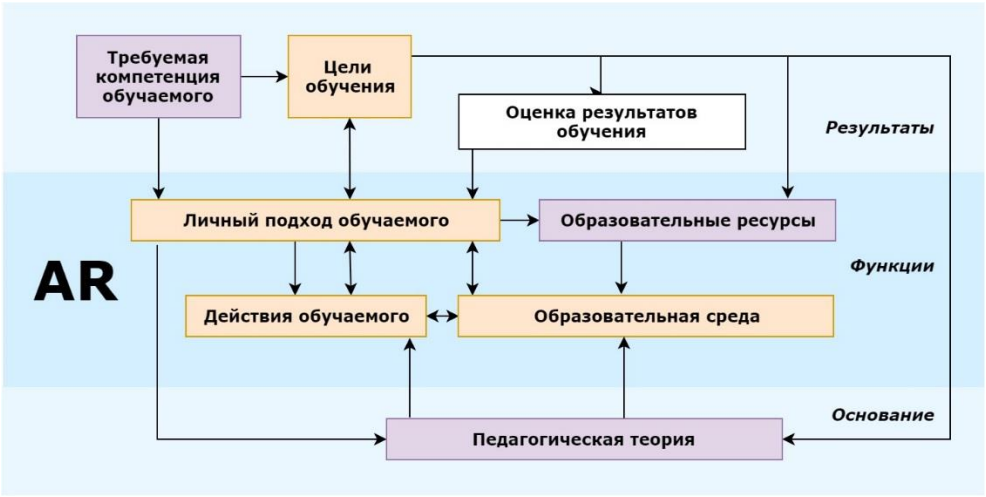


Рисунок 1 – Структура Mobile Augmented Reality Education

В представленной схеме показано, что к сфере AR относятся прежде всего образовательные ресурсы, совокупность которых создает образовательную среду.

В нашем случае – это школьные учебники. Совмещая с AR-технологиями – это интерактивные учебники с визуализацией информации и возможностью внедрения звуковых эффектов.

Преимущества AR-технологий

Анализ текущей ситуации в сфере образования позволяет сделать вывод, что AR-технологии обладают существенными преимуществами, среди которых можно выделить следующие пункты:

1) Простота внедрения.

Не требуется кардинальное изменение методики преподавания. Это не ликвидация бумажных учебников, к которым все привыкли, а расширение их возможностей.

2) Устранение возрастного разрыва между поколениями и технологиями.

AR не вызывает существенных сложностей в восприятии у людей старшего поколения – учителям не нужно осваивать новые технологии.

3) Отсутствие необходимости дополнительного финансирования: учащиеся могут пользоваться своими собственными устройствами.

Технологии VR вызывают опасения, и прежде всего у учителей и родителей — не уйдут ли дети в виртуальный мир с головой. AR по своей сути требует контакта с реальностью, не заменяя, а дополняя ее. Это устраняет возможные психологические опасности применения данной технологии в образовании, начиная с самого раннего возраста [4].

Мировые компании-лидеры активно поддерживают образовательные учреждения, разрабатывая новые образовательные программы. Например, компания Google бесплатно продвигает в школах свой проект Cardboard, к началу 2016 года было готово более 100 учебных программ. Кроме школ, проектами виртуальной и дополненной реальности интересуются многие медицинские образовательные учреждения [5].

Примеры AR-книг

1) Incredebooks – воплощая сказки в жизнь

В то время как электронные книги и мобильные игры быстро вытесняют книги в твердом переплете, Incredebooks сохраняют интерес к чтению книг среди детей нынешнего поколения. Incredebooks объединили обычную книгу с последними мобильными технологиями*:* дети и взрослые могут не только прочитать рассказ, но и визуализировать его с помощью приложений дополненной реальности [6].

В сравнении с другими AR-книгами Incredebooks обладают следующими отличительными особенностями: сказки сопровождаются музыкой и звуковыми эффектами, есть элементы геймификации.

2) Газета «Tokyo Shimbun»

Совместно с рекламной фирмой издательство японской газеты «Tokyo Shimbun» разработало программное обеспечение, которое переводит взрослые газеты на «детский язык». Приложение дополненной реальности AR News показывает упрощенную версию текста, понятную детям [7]. Если навести гаджет на заголовок статьи, на экране появляются мультимедийные персонажи-помощники, активируется диалоговый

режим.

Разработка приложения

На основе проведенного анализа AR-технологии была разработана структура (рис. 2) и диаграмма состояния (рис. 3) универсального обучающего приложения с дополненной реальностью. Разрабатываемое приложение состоит из 3-х модулей:

1) модуль сканирования,
2) модуль просмотра 3D-объекта,
3) информационный модуль.

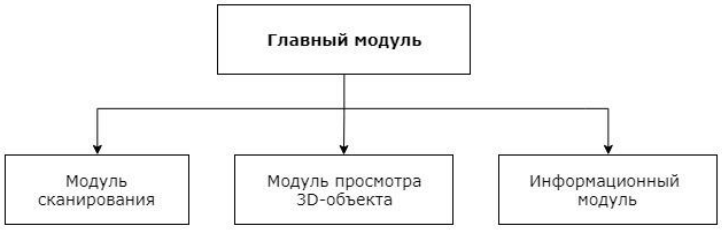


Рисунок 2 – Структура приложения

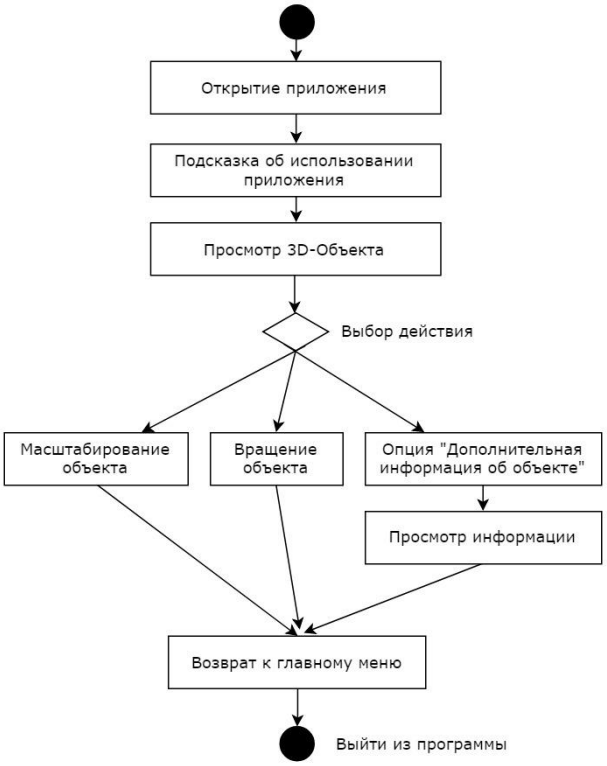


Рисунок 3 – Диаграмма состояний

Для реализации обучающего приложения будут использованы:

- межплатформенная среда разработки компьютерных игр Unity,
- фреймворк Vuforia,
- среда разработки приложений Android SDK,
- пакет для создания трёхмерной графики и анимации Cinema 4D.

Выводы

Для достижения поставленной цели была проанализирована технология дополненной реальности, рассмотрена методология MARE, разработаны структура и диаграммы состояний приложения, приведены необходимые программные средства для его реализации.

Образование с использованием дополненной реальности, позволяет наглядно вести лекции и семинары, проводить тренинги, показывать обучающимся все аспекты реального объекта или процесса, что в целом дает колоссальный эффект, улучшает качество и скорость образовательных процессов.

Литература

1. Марчев Д.В. Использование технологий иммерсивной виртуальной реальности в обучении / Марчев Д.В., Пылькин А.Н., Бакина Ю.О // Всероссийская научно-методическая конференция: виртуальная и дополненная реальность-2016: состояние и перспективы. – 2016. – С. 254-256.

2. Решение для тренировочных и обучающих программ в виртуальной реальности [Электронный ресурс]. – Режим доступа: https://proektoria.online/projects/reshenie-dlya-trenirovochnyx-i-obuchayushhix-programm-v-virtualnoj-realnosti/.

3. Egui Zhu1. Design of Mobile Augmented Reality in Health Care Education: A Theory-Driven Framework / Egui Zhu1, Anneliese Lilienthal1, Lauren Aquino Shluzas, Italo Masiello, Nabil Zary// JMIR Medical Education. – 2018. – Vol. 1, No. 2.

4. Книги с дополненной реальностью как эффективный образовательный инструмент [Электронный ресурс]. – Режим доступа: http://q-ar.ru/knigi-s-dopolnennoj-realnostyu-kak-effektivnyj-obrazovatelnyj-instrument/.

5. 9 сфер применения виртуальной реальности: размеры рынка и перспективы [Электронный ресурс]. – Режим доступа: https://vc.ru/flood/13837-vr-use.

6. INCREDEBOOKS – воплощая сказки в жизнь [Электронный ресурс]. – Режим доступа: http://augmentedreality.by/news/inncredebooks-reality-books/.

7. Приложение AR News переводит взрослые статьи на детский язык [Электронный ресурс]. – Режим доступа: https://hi-news.ru/software/prilozhenie-ar-news-perevodit-vzroslye-stati-na-detskij-yazyk.html.

М.Б. Бобелюк, Н.Е. Губенко. Дополненная реальность как концепция создания обучающего приложения. *В докладе рассматриваются актуальные вопросы использования дополненной реальности в образовании. Представлена структура Mobile Augmented Reality Education. Приводится структура и диаграмма состояний разрабатываемого обучающего приложения.*

Ключевые слова: *дополненная реальность, AR, образовательная система, MARE, Mobile Augmented Reality Education, Unity, Vuforia, Android SDK.*

Marina Bobelyuk, Natalia Gubenko. Augmented reality as an effective tool in solving the problems of the education system. *Complemented reality as conception of creation of teaching application. In a lecture the pressing questions of the use of the complemented reality are examined in education. The structure of Mobile Augmented Reality Education is presented. The structure and state diagram of the developed training application is given.*

Key words: *augmented reality, AR, educational system, MARE, Mobile Augmented Reality Education, Unity, Vuforia, Android SDK.*

УДК 004.588

«Обучающая система с адаптацией по форме изложения материала на основе ментальных карт»

Давыденко Д. П., Губенко Н. Е.
Донецкий национальный технический университет
кафедра компьютерного моделирования и дизайна.
darinochka.davidenko@yandex.ru

Давыденко Д. П., Губенко Н. Е. Обучающая система с адаптацией по форме изложения материала на основе ментальных карт. *В статье рассматриваются адаптивность изложения учебного материала в компьютерных обучающих системах для вовлечения обучаемых в познавательную деятельность.*

Ключевые слова: *адаптивность, адаптивные обучающие системы, адаптация, компьютерные обучающие системы, ментальные карты.*

Постановка проблемы

В условиях роста количества информации, появления новых технологий перед системой образования встает ряд серьезных задач, связанных с организацией познавательной деятельности. Тенденция информатизации становится все более очевидной: создаются электронные учебники, развивается дистанционное обучение и разрабатываются компьютерные обучающие системы. Основным преимуществом применения информационных технологий в учебном процессе является индивидуализация обучения и эффективная адаптивность представления учебного материала.

Цель работы: разработка компьютерной обучающей системы, способной адаптировать уровень представления учебного материала к индивидуальным способностям и возможностям обучаемого.

Компьютерные обучающие системы предоставляют возможность обучающимся самостоятельно управлять ходом учебного процесса и оказывают положительное влияние на мотивацию, что способствует лучшему усвоению учебного материала. Образование при использовании информационных технологий приобретает персонифицированный, ориентированный характер. Обучающийся становится основным, если не единственным субъектом образовательного процесса, а его главным элементом – не только знание, но и информация [1].

С точки зрения управления учебным процессом все обучающие системы можно разделить на два класса [2]:

– обучающие системы, в которых управление процессом обучения возложено на пользователя. Учебные материалы содержатся на машинном носителе в текстовом и графическом форматах: электронный учебник или методическое пособие, электронная библиотека.

– обучающие системы, самостоятельно управляющие учебным процессом. Содержит изложение учебной дисциплины или ее раздела в соответствии с ее логикой на машинном носителе в текстовом, графическом, аудио, видео форматах. В конце каждой порции изложения учебной дисциплины в данных системах обучаемому предоставляются проверочные задания.

В отличие от систем первого класса, в данных системах ответы и действия обучаемого влияют на дальнейший ход процесса обучения. Степень управления учебным процессом напрямую зависит от степени адаптации системы под конкретного обучаемого, поэтому обучающие системы данного класса разделяются на подклассы по степени их адаптивности и способами реализации адаптации:

1) Компьютерная обучающая система с линейной моделью обучения – это система, в которой структура представления учебного материала является последовательной. В зависимости от результатов проверки обучаемому предоставляется очередная (следующая) порция информации, либо он возвращается к дополнительному изучению предшествующей порции знаний.

2) Компьютерная обучающая система с разветвленной моделью обучения – это система, в которой задано несколько вариантов изложения материала, различающихся по степени подробности, глубине изложения, а также несколько вариантов предлагаемых в конце каждой порции проверочных заданий с различными уровнями сложности. Данная система адаптируется по глубине, степени подробности изложения изучаемого материала и сложности проверочных заданий, что позволяет ей формировать индивидуальную траекторию обучения.

3) Компьютерная обучающая система с адаптацией по форме изложения – это система, в которой обучаемый имеет возможность выбирать форму изложения учебного материала: текстовая, или графическая,

или аудио, или видео форма. Система может обладать всеми или несколькими свойствами компьютерной обучающей системы с разветвленной моделью обучения.

4) Мультиагенная компьютерная обучающая система с адаптацией по объекту и целям обучения – это система, в которой управление учебным процессом осуществляется по целенаправленной установке обучаемого, каждая из которых в отдельности обладает всеми свойствами обучающих систем предыдущих подклассов. Целевые установки составляется каждый раз под конкретного обучаемого, и под его цели обучения.

Адаптивной называется «образовательная система, способная каждому ученику помочь достичь оптимального уровня интеллектуального развития в соответствии с его природными задатками и способностями. Обладая такими свойствами, как гибкость, полиструктурность, открытость, адаптивная образовательная система выводит обучаемого на более высокий потенциально возможный уровень развития, адаптируя его к своим требованиям»[3].

Компьютерные адаптивные обучающие системы (АОС) обладают рядом характерных черт:

– индивидуальность

– гибкость (возможность заниматься в удобное для себя время, в удобном месте и темпе)

– модульность (возможность формировать учебный план, отвечающий индивидуальным потребностям)

– параллельность (параллельное с профессиональной деятельностью обучение)

– охват (одновременное обращение ко многим источникам учебной информации)

– экономичность (эффективное использование учебных площадей)

– социальное равноправие (равные возможности получения образования)

Компьютерные АОС положительно влияют на учащегося, повышают его творческий и интеллектуальный потенциала за счет самоорганизации, стремления к знаниям, умения взаимодействовать с компьютерной техникой и самостоятельно принимать ответственные решения. Обучение в компьютерных обучающих системах за частую прибегают к использованию асинхронного подхода. При асинхронном подходе обучающийся сам определяет темп обучения.

Преимущества АОС перед традиционными:

– обучение идет быстрее и усваивается глубже, если обучающийся проявляет активный интерес к изучаемому предмету.

– обучение является более эффективным, если формы приобретения знаний и навыков таковы, что без труда могут быть перенесены в условия "реальной жизни", для чего они и предназначены.

– обучение идет быстрее, если обучающийся узнаёт результат каждого своего ответа.

– обучение идет качественнее, если программа по предмету построена по принципу последовательного изложения материала.

– знание результатов своей работы стимулирует выполнение очередного задания.

– процесс обучения следует организовать так, чтобы каждый учащийся мог усваивать программу.

Адаптацию учебной деятельности можно определить, как процесс изменения параметров и структуры модели обучаемого и выработке соответствующих обучающих воздействий на основе осведомляющей информации с целью достижения заданного состояния обучаемого при его начальной неопределенности и изменяющейся педагогической среды [4].

Предлагается подход к организации адаптивной системы обучения основан на следующих принципах:

– построение модели обучаемого на основе анализа результатов тестирования первичных знаний обучаемого

– создание модели процесса обучения в соответствии с возрастом обучаемого;

– динамическое формирование структуры и содержания учебного материала средствами адаптивной обучающей системы.

Главное требование к адаптивным обучающим системам состоит в обеспечении максимальной степени индивидуализации процесса обучения, т. е. адаптация к каждому конкретному обучаемому. В настоящее время при обучении наиболее актуальной является задача создания и широкого внедрения в учебный процесс адаптивных компьютерных систем обучения.

Виды адаптации обучающих систем

Механизмы адаптации являются важной темой исследований последнего времени в области электронного обучения. Анализ существующих решений позволяет выделить восемь видов адаптации, реализуемых в обучающих системах [6]:

Адаптация, ориентированная на интерфейс (адаптивная навигация), которая приводит к изменению свойств элементов и вариантов компоновки экранного интерфейса.

Адаптация, ориентированная на обучение, в которой процесс обучения динамически адаптируется к последовательности содержания курса. Путь обучения является динамичным и персонализированным как для

каждого обучаемого, так и для каждого курса в целом, так что обучаемый может идти по различным вариантам обучения в зависимости от курса, успеваемости и т.п.

Адаптация, ориентированная на содержание, которая приводит к динамическому изменению содержания. Например, информация курса подразделяется на три уровня детализации, и соответствующий уровень отображается в зависимости от значений определенных факторов.

Интерактивная поддержка решения задач, которая помогает обучаемому на следующем шаге получить правильное решение задачи. Руководство может осуществляться как в режиме реального времени, так и в автономном режиме, или по заранее установленным правилам.

Гибкая фильтрация информации, которая обеспечивает получение только той информации, которую должен видеть обучаемый.

Гибкое группирование, которое позволяет создавать группы обучаемых и организовывать поддержку взаимодействия по выполнению конкретных задач.

Гибкое оценивание, в котором модели оценки, реального содержания и тестирования могут изменяться в зависимости от успехов обучаемого и рекомендаций преподавателя.

Изменения на лету – возможность изменять/адаптировать курс преподавателем или автором курса в режиме реального времени.

Как показывает анализ, большинством этих свойств будет обладать разрабатываемая компьютерная адаптивная обучающая система “ZOOМИР” для обучаемых дошкольного и школьного возраста. Она предназначена для формирования первичных и базовых знаний о систематике животного мира, имеет развернутый сценарий, возможность просмотра видео о животных и снабжено увлекательными викторинами, загадками и кроссвордами для дошкольников и тестированием для школьников.

Адаптивная обучающая система должна обеспечивать максимальную продуктивность изложения материала с учетом индивидуальных возможностей учащихся. АОС можно представить в виде следующей иерархической структуры (см. рис.1).

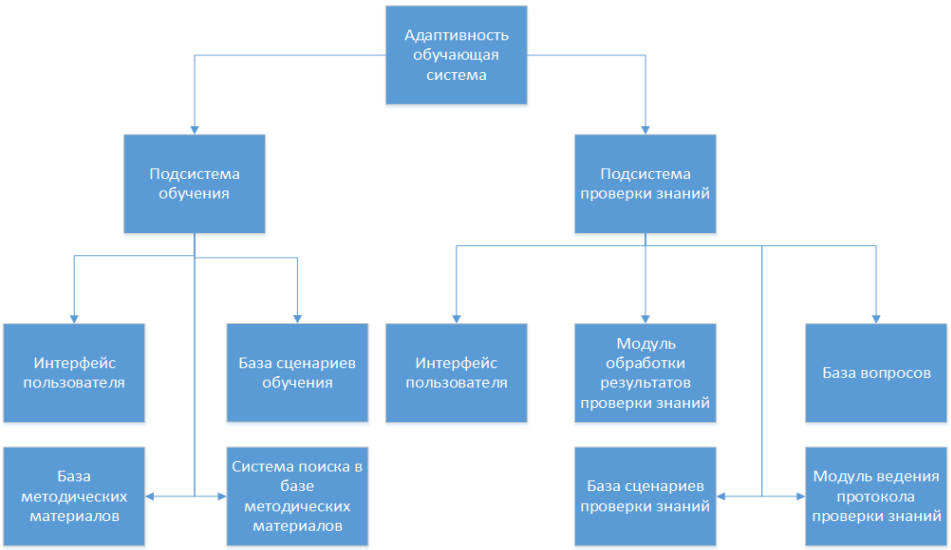


Рисунок 1 - Структурная схема АОС

Адаптивность представления учебного материала с применением ментальных карт

Для усвоения увеличивающегося объема учебного материала за ограниченное время необходима интенсификация образовательного процесса за счет систематизации, структурирования информации. Выбор новых методов и активных средств также позволяет решить указанную проблему. Одной из современных методик структурирования информации является использование ментальных карт. Автором методики «карты ума», способствующей творческому мышлению, запоминанию и организации умственной деятельности является психолог Тони Бьюзен.

Ментальные карты (Mind Mapping) — это способ систематизации знаний с помощью схем; это технология изображения информации в особом графическом виде [7].

Ментальные карты помогают эффективно структурировать и обрабатывать информацию, а также использовать творческий и интеллектуальный потенциал как разработчика обучающей системы, так и обучаемого.

Явные преимущества ментальных карт:

– наглядность

– привлекательность и эстетичность

– запоминаемость

– творчество

– возможность пересмотра.

Ментальные карты визуализируют информацию с помощью различных приемов (изображения, значки, цвета, контуры), что очень помогает запоминанию.

Обучающий материал в разрабатываемой АОС представлен в виде вложенных ментальных карт, что позволяет обучающемуся легко ориентироваться в содержании материала и легко запоминать информацию (см. рис.2).

Рисунок 2 – Пример ментальной карты в АОС “ZOOМИР”

Выводы

Таким образом, нами планируется создание адаптивной компьютерной обучающей системы с использованием современных технологий, что позволит реализовать оперативный контроль качества обучения. Использование адаптивной компьютерной обучающей системы и структурирование обучающего материала с помощью ментальных карт повысит эффективность обучения.

Литература

1. Ильин Г. От педагогической парадигмы к образовательной // Высшее образование в России.– 2000. – № 1. – С. 64 – 69.
2. Ананьева Т.Н., Черткова Е.А. Методология разработки компьютерных обучающих систем для сферы образовательных услуг // Теоретические и прикладные проблемы сервиса. 2007. № 2. С. 48-51.
3. Капустин Н.П. Педагогические технологии адаптивной школы. – М.: Академия, 1999. – 216 с.
4. Ловцов Д.А., Богорев В.В. Адаптивная система индивидуализации обучения // Педагогика, 2001, №6. – С. 24-28
5. 3. Blanchard E., Razaki R., Frasson C. Cross-Cultural Adaptation of e-Learning Contents: a Methodology // Proceedings of the International Conference on E-learnung: ELEARN2005. − 2005. – P. 1895
6. Burgos D., Tattersall C., Koper R. How to represent adaptation in eLearning with IMS Learning . – The Open University of the Netherlands, 2006.
7. Что такое Ментальные карты и для чего они? //Статья. URL: http:// mind-manager.ru/article/what_ is_mindmaps.

Давыденко Д. П., Губенко Н. Е. Обучающая система с адаптацией по форме изложения материала на основе ментальных карт. *В статье рассматриваются адаптивность изложения учебного материала в компьютерных обучающих системах для вовлечения обучаемых в познавательную деятельность.*

Ключевые слова: адаптивность, адаптивные обучающие системы, адаптация, компьютерные обучающие системы, ментальные карты.

Davydenko D.P., Gubenko N. Ye. Training system with the adaptation of the presentation of the material based on mental maps. In the article adaptive materials on educational materials in computer learning systems are considered to involve trainees in cognitive activity.

Keywords: *adaptability, adaptive learning systems, adaptation, computer learning systems, mental maps.*

УДК 004

Анализ корпоративного портала с целью улучшения продуктивности компании

Дементьева Е.Е., студент гр. ПИм-171, II курс
Научный руководитель: Рейзенбук К.Э., ст. преподаватель
Кузбасский государственный технический университет
имени Т.Ф. Горбачева, филиал в г. Кемерово
г. Кемерово

Множество компаний за последнее время, чтобы повысить эффективность, начали развивать порталы внутри компании. Корпоративный портал – это веб-интерфейс, который дает сотрудникам доступ к информации и сервисам компании [1]. Портал – это своего рода внутренний офис, где сотрудники обмениваются данными [2].

Внутренний портал нужен для повышения рабочей дисциплины, хранения большого количества информации, для ускорения взаимодействия сотрудников между собой, так же он позволит облегчить работу, повысить продуктивность, сэкономит рабочее время, снизит расходы.

Внутренний портал снабжают множеством функций, которые решают потребности компании. С помощью этих функций мы получаем систему хранения значимой информации. Если понадобится, какой-либо документ, то не нужно будет обращаться к бумажным архивам, и тратить на поиски большое количество времени, всю информацию можно найти в портале, даже если эта информация пятилетней давности.

Так же с помощью корпоративного портала, сотрудники с легкостью будут общаться между собой, узнавать любую информацию о друг друге (номер рабочего телефона, номер кабинета, ФИО).

Так же одно из значимых – это система управление задачами и проектами [3]. Есть часто повторяющиеся действие, например, такие как последовательность подтверждения нормативных документов или согласование служебных записок с руководителями отделов, подписание на замещение сотрудников во время отпуска.

При наличии удобного и понятного интерфейса для контроля всех задач, сотрудники быстро будут справляться со всеми обязанностями, не придется печатать документ и идти по всем руководителям и согласовывать с каждым что-либо.

Например, в компании «Илва» есть свой внутренний портал, первоначально он предназначался для публикаций новостей, и, каких-либо материалов, имелся форум для внутреннего общения, создавались базы документов. Сейчас же чтобы внутрикорпоративные коммуникации совершенствовались, было разработана системы управления задачами, календари, хранилища документов, возможность создать профиль с предоставляющими правами на доступ к документам, уведомления и многое другое.

Внедрение портала компании – это только начало большой работы. Только отслеживая реакции пользователей, можно понять, насколько продуктивно он работает.

Было решено внести еще больше изменений, чтобы помогать сотрудникам оптимизировать совместную работу. Был внедрен в раздел «Маршруты» подраздел «Управление маршрутами», где можно отследить, кто во время отсутствия сотрудника его замещает, так же когда ставится замещение, то автоматически тот сотрудник, который замещает отсутствующего, имеет права к его приоритетам в портале. То есть, может иметь доступ к определенным разделам, или согласовывать служебные записки, которые сразу переходят по маршруту к замещающему лицу (см. рис.1).

Так же была добавлена возможность увидеть действующие документы или те, которые находятся в архиве. На случай, если сотруднику понадобился, документ, который еще действует для него, он может просто перейти и посмотреть все, что ему нужно. Или же кто-то из сотрудников давно уволился, но есть необходимость в каком-то документе, который он согласовывал, можно зная или номер этого договора или фамилию, или даже дату согласования найти этот договор.

В данном корпоративном портале имелся раздел служебных записок, но он был недостаточно расширен и функционален, поэтому, было решено добавить множество усовершенствований данной вкладки. Такие как доступ в здание в нерабочее время, или на просьбу отпустить с рабочего места, и, чтобы повысить контроль и аналитику компании, ведь с расширением компании, становится сложнее контролировать сотрудников, была введен тип записки: «отсутствовал на рабочем месте». Когда сотрудник опоздал, или по какой-то причине не появился на работе, он пишет записку с какого по какое время он отсутствовал, по какой причине, и отправляет на согласования, руководителям, это очень помогает следить за рабочей дисциплиной (см. рис.2).

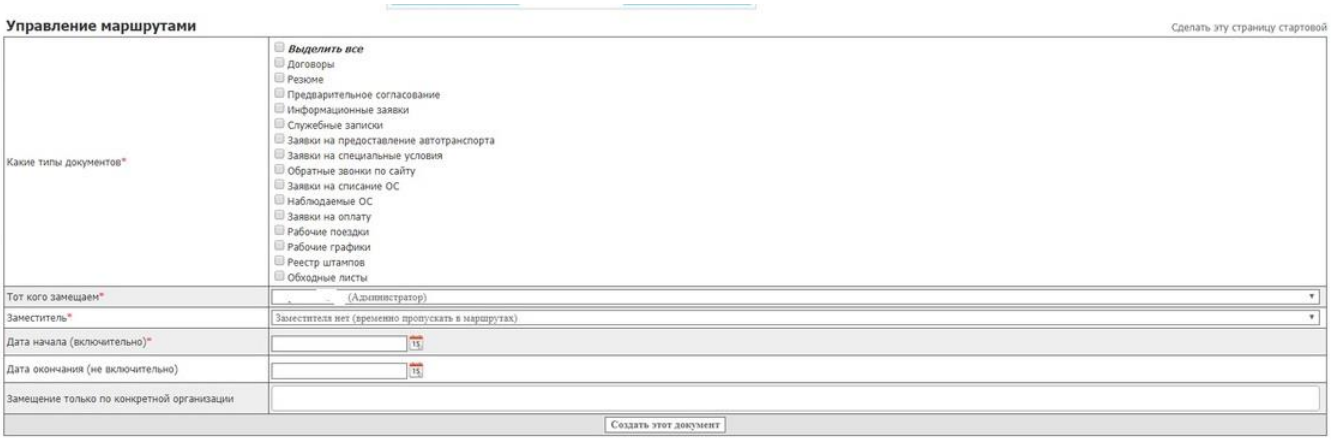


Рисунок 1 – Форма управление маршрутами

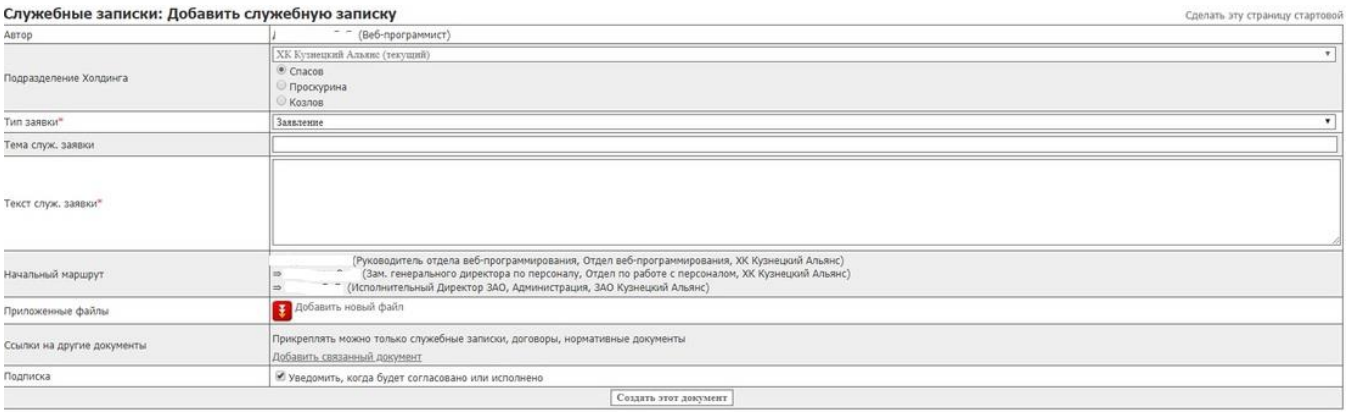


Рисунок 2 – Форма добавления служебной записки

Так как иногда пользователи могут, что-то перепутать, согласовать или отменить документ, или возникла ошибка, и им нужно срочно исправить ее, они звонят программисту для решения, и, чтобы он смог быстро зайти к ним на страницу, была создана вкладка: «Войти как» (см. рис.3).

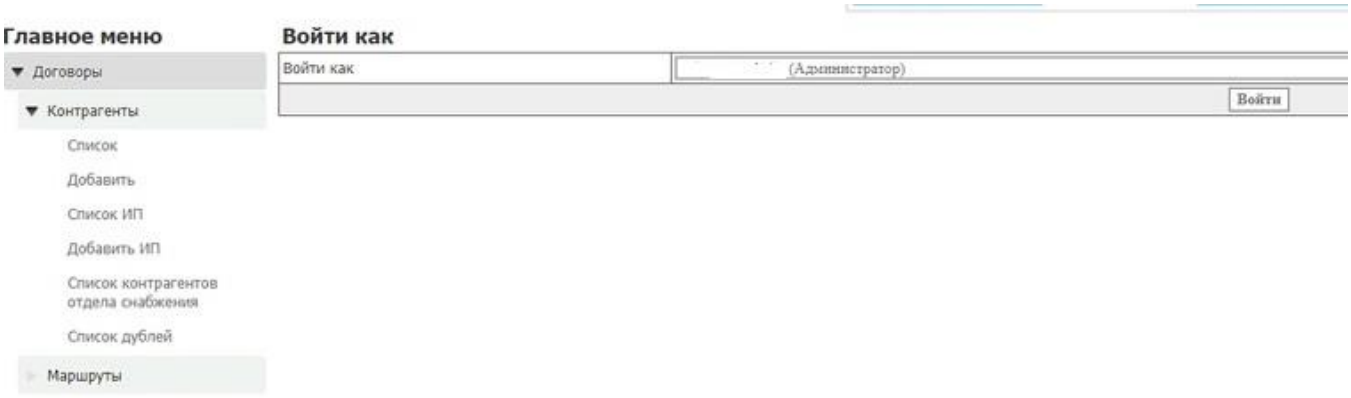


Рисунок 3 – Форма «Войти как»

В данной форме программист может зайти под фамилией сотрудника, понять, в чем проблема, и быстро ее решить.

Еще много внедрений находятся в разработке, благодаря всем этим изменениям компания будет становиться намного престижней и восходить на новый уровень.

Главный плюс от запуска всех новшеств – организационный. Люди будут работать гораздо продуктивнее, снизится количество бумажной волокиты.

Если в крупных компаниях, где работает большое количество людей, будет такая экономия времени, то это значительно снизит расходы. Поэтому чтобы был такой эффект от портала, нужно правильно и грамотно им пользоваться: внедрять, дорабатывать, уметь настраивать и конечно же отслеживать эффективность.

Список литературы:

1. Корпоративные порталы: функции, задачи и метрики эффективности. Режим доступа: https://www.uplab.ru/blog/corporate-portals/

2. Корпоративный портал. Режим доступа: http://www.aif.ru/boostbook/korporativnyi-portal.html

3. Корпоративный портал. Что это? Зачем? Кому он нужен? Режим доступа: https://habr.com/post/326724/

УДК 004.05

Анализ моделей исследования удовлетворенности потребителей мультимедийной продукции

Кандаурова А.О., Губенко Н.Е.
Донецкий национальный технический университет
rafalelka97@gmail.com

Кандаурова А.О., Губенко Н.Е. Анализ моделей исследования удовлетворенности и потребителей мультимедийной продукции. *В данной статье рассматриваются понятия удовлетворенности и лояльности потребителей. Определяется их роль в успешности разработчиков мультимедийной продукции. Проводится анализ существующих современных методик и моделей исследования лояльности потребителей.*

Ключевые слова: *удовлетворенность потребителей, лояльность, мультимедийная продукция, системы управления отношениями с клиентами.*

Постановка проблемы

В современном мире на практике не принято прогнозирование уровня удовлетворенности потребителей создаваемого продукта, что впоследствии может стать причиной «провала» его на рынке. В организациях, занимающихся разработкой мультимедийной продукции, процесс общения с клиентом зачастую налажен не так хорошо. Это связано со спецификой работы, потому что разработка продукта – во многом творческий процесс, который сложно документировать, привести к единым требованиям. Это приводит к тому, что часто возникают неточности в организации работы, неясности и грубые ошибки со стороны, как разработчика, так и клиента. Весомая причина этому – недопонимание между клиентом и разработчиком. В свою очередь, оно вызвано недостатком информации о проекте, искаженной ее подачей и так далее. В связи с этим многие организации прибегают к стратегии, направленной на повышение удовлетворенности и лояльности клиентов. В рамках осуществления данной стратегии проводится мониторинг удовлетворенности и лояльности потребителей, внедряются системы управления отношениями с клиентами (CRM), проводятся анкетирования потребителей. Обеспечение удовлетворенности потребителей является одним из ключевых инструментов успеха создаваемого продукта. Удержать имеющихся, пополнять ряды постоянных клиентов, создать более эффективные продукты – задачи общие для всех разработчиков, и их эффективное решение невозможно без внедрения в практику управления непрерывного мониторинга удовлетворенности потребителей.

Этим обусловлена актуальность выбранной темы для организаций, заинтересованных в создании системы исследования удовлетворенности и лояльности потребителей и получении научно обоснованных данных от потребителей для принятия управленческих решений как стратегического, так и тактического плана.

Роль удовлетворенности потребителей в успешности разработчика

Удовлетворенность определяется как психологическое состояние человека в результате достижения желаемой цели. Она характеризует степень соответствия ожиданий от приобретаемого продукта его фактическому состоянию. Удовлетворенность потребителей определяется как «чувство, испытываемое потребителем после приобретения или использования продукта» [1]. В известных источниках отмечается, что удовлетворенность формируется в результате сравнения ожидаемых и фактических свойств продукта и всего того, что с ним связано.

Цель изучения удовлетворенности потребителей – добиться того, чтобы у максимального числа покупателей ожидания сбылись, еще лучше «перекрывались» характеристиками продукта, вызывая у них восторг. Для эффективного управления уровнем удовлетворенности необходимо выяснить механизм ее формирования и влияющие факторы. Измерение удовлетворенности потребителей является обязательным атрибутом систем управления взаимоотношениями с потребителями (CRM).

Перед началом разработки продукта необходимо найти решение, которое могло бы сократить время работы и минимизировать ошибки на стадии создания. В ходе решения этих проблем стоит уделить особое внимание удовлетворенности потребителей, поскольку именно она лежит в основе формирования их

лояльности к разработчику, что впоследствии становится источником его финансового благополучия, тем самым играет важную роль в его успешности на рынке.

На сегодняшний день определение уровня удовлетворенности потребителя осуществляется со стандартами управления качеством ISO 9000:2000, ISO 9001:2000, ISO 9004:2000 [2].

{Качество} = {Удовлетворенность потребителя} = {Ценность} / {Стоимость}

Стандарты охватывают следующие области: ответственность руководства, управление ресурсами, реализацию продукции, измерение, анализ и улучшение.

Оценка и управление уровнем удовлетворенности потребителей позволяет:

– делать лучше то, что значимо для потребителей;

– определить запросы потребителей и их относительную важность;

– понять, как потребители воспринимают создаваемый продукт;

– определить неудовлетворенные запросы и области, которые наиболее значимы потребителям;

– выявить неадекватно установленные разработчиком приоритеты потребителя;

– выявить лишние функции продукта;

– поставить цели улучшения продукта и контролировать процесс их достижения;

– повысить результативность путем увеличения лояльности потребителей.

Анализ удовлетворенности потребителей помогает гораздо проще составить общую картину создаваемого мультимедийного продукта, выделить основные требования к нему, а также гораздо легче оценить работу разработчика на предмет ошибок.

Связь, которая устанавливается удовлетворенностью между разработчиком и процессом потребления мультимедийной продукции, выражается через отношение потребителя к товару и зависит от определенного момента времени, конкретного потребительского сегмента, товарного рынка и его структуры.

Удовлетворенность потребителей формируется под влиянием некоторых основополагающих факторов:

– сложившейся репутации организации-разработчика;

– воспринимаемого качества товара;

– ожиданий потребителей относительно товара;

– воспринимаемой ценности, полученной в процессе потребления.

Стоить заметить, что удовлетворенность потребителей прямым образом формирует лояльность потребителей. Лояльность потребителей обуславливается положительным отношением потребителей в отношении всей деятельности организации-разработчика, ее продукции и услуг, к персоналу компании, ее имиджу, торговой марке, логотипу. Удовлетворенность и лояльность тесно связаны между собой, следовательно, только в высшей степени удовлетворенные потребители остаются лояльными по отношению к товарам и услугам организаций. На сегодняшний день в связи с этим разработано несколько универсальных моделей исследования лояльности потребителей, которые работают с разной эффективностью на различных группах товаров. Поэтому целесообразно провести их анализ на предмет применения к оценке лояльности потребителей мультимедийной продукции.

Современные модели и методики исследования лояльности потребителей

Методики изучения лояльности, как правило, подразделяются, на две группы – это эмпирические методики и математические методики. Однако сами по себе они встречаются довольно редко. Обычно их применяют комплексно, так как они позволяют определять разные характеристики исследуемого процесса. Эмпирические методики направлены на выявление наличия лояльности и определение её уровня, а математические – на построение кривой лояльности, выявление удовлетворённости, расчёт индекса поддержки и влияния факторов, посредством которых формируется лояльность.

Выделяют шесть основных методик исследования лояльности клиентов, среди которых присутствуют как эмпирические, так и математические.

Метод разделения потребностей возник в 50-х годах XX столетия на основе опыта многих компаний, занимающихся торговлей и предоставлением услуг. Смысл его заключается в определении степени лояльности клиента в численном эквиваленте. К примеру, если человек приобрел нужный ему товар у условной компании 7 раз из 10, это говорит, что она удовлетворяет 70% его потребностей. И уровень лояльности здесь определяется тем, сколько раз клиент обращается к одной и той же компании, игнорируя аналогичную продукцию других компаний. Но у метода разделения потребностей есть один существенный недостаток – клиенты могут покупать товар той или иной компании не только потому, что являются действительно лояльными. И определить, какие именно покупки были совершены по причине истинной лояльности, очень проблематично. К тому же, доля повторных покупок в процентном соотношении является очень субъективной величиной, чтобы определять лояльных и нелояльных клиентов.

Конверсионная модель была предложена двумя исследователями Яном Хофмейром и Бутчем Райсом. Она позволяет определять степень приверженности клиентов. Основными показателями здесь являются: удовлетворенность брендом, важность выбора бренда, альтернативы и колебания. Удовлетворенность указывает на то, что чем выше ее показатель, тем более велика вероятность возникновения приверженности.

Но удовлетворенность нельзя соотнести с поведением, из-за чего раскрыть причины поступков клиентов в полной мере не представляется возможным. Если бренд не играет роли для клиента, то и приверженности достичь будет трудно, а значит, бренд должен быть интересен человеку. И если бренд имеет значение для него, то он обязательно уделит время на то чтобы определиться, какой именно ему подходит более всего. Еще одна причина, по которой клиент сохраняет лояльность компании – это понимание того, что альтернативы хуже выбранного бренда. Однако если клиент сможет найти более интересную для него альтернативу, он сможет изменить свои предпочтения.

Методика ANA была разработана специалистами авиакомпании «ANA». Исследуя вопросы лояльности клиентов, они выделили четыре основных клиентских сегмента:

– клиенты, которые не имеют возможности выбора и вынуждены приобретать продукт одной компании;

– неудовлетворенные и отрицательно настроенные клиенты;

– неопределившиеся клиенты;

– постоянные клиенты.

Большое значение в модели ANA отводится привлечению постоянных потребителей из сегмента неопределившихся клиентов. Но компания должна не просто удовлетворить потребности клиента, но и «сработать» сверх его ожиданий, т.к. клиент, который удовлетворён частично, может в дальнейшем выбрать другую компанию, при этом предполагая, что новый выбор будет лучше. Таким образом, для сохранения клиента компания должна как привлекать новых потребителей, так и удерживать старых, мотивируя их на новые приобретения.

Методика, предложенная специалистом по лояльности клиентов Питером Уилтоном, интересна тем, что в ней имеется несколько уровней лояльности. В частности, автор делит лояльных клиентов на «адвокатов» и «союзников» компании. Адвокатами являются люди, которые относятся к бренду положительно, верны ему и дают рекомендации другим людям. Но конечной целью компании должно быть создание «союзников» – людей, не только приверженных компании и довольных ей, но и участвующих в бизнес-процессах.

Автором еще одной методики является известный американский специалист по вопросам маркетинга Дэвид Ааакер. Он предлагает несколько вариантов измерения лояльности, основанных на исследованиях:

– покупательского поведения;

– учета затрат на переключение;

– удовлетворенности;

– отношения к бренду;

– приверженности.

Самым простым способом измерения степени лояльности является наблюдение за покупательским поведением. Ключевыми параметрами здесь являются:

– показатели повторных покупок;

– процентный показатель покупок (сколько покупок из пяти приходится на каждый бренд);

– количество купленных товаров разных брендов (сколько покупателей приобрело товар одного бренда, сколько – двух и так далее).

Затраты на переключение подразумевают изучение предпочтений покупателей в приобретении товаров одного бренда и их нежелания менять эти предпочтения. Степень удовлетворённости является основополагающим фактором, а отношение к бренду может проявляться в уважении, доверии, готовности заплатить цену, превышающую цены конкурентов.

На данный момент времени методика Аакера нашла широкое применение. В целях её реализации используются различные виды опросов и анкетирования, направленные на определение каждого конкретного показателя лояльности, например, отношения или уровня удовлетворённости. Затем происходит построение индексов лояльности, для чего находятся арифметические значения средних баллов по всем аспектам. Таким образом, появляется возможность сделать оценку абсолютно всех факторов, формирующих степень лояльности.

Создателем еще одной модели является Жан-Жак Ламбен – один самых известных представителей европейской маркетинговой школы. Его оценка лояльности осуществляется по трём критериям:

– качество ключевых выгод продукта;

– процесс оказания услуг;

– ценность услуг, воспринимаемая клиентом.

Управляя лояльностью, компания должна оценивать изменения приверженности клиентов, сопоставляя ее с периодами наибольших скачков, когда наблюдалось увеличение прибыли, связанное с притоком новых клиентов и повторными покупками, а также решать новые задачи и находить новые направления для развития.

Анализируя, представленные методики исследования лояльности, можно сделать вывод, что при их комплексном применении можно определить все необходимые показатели: наличие лояльности и ее уровень и тенденции развития, а также проанализировать удовлетворённость продуктом и сервисом компании.

Вся эта информация помогает узнать, каким образом необходимо выстраивать дальнейший процесс по управлению лояльностью, понять, какие изменения происходят в лояльности во времени и найти способы улучшения качества сервиса, в которых будут учтены интересы потребителей и их особенности. Для решения этого вопроса разрабатывается система, диаграмма прецедентов которой приведена на рисунке 1.

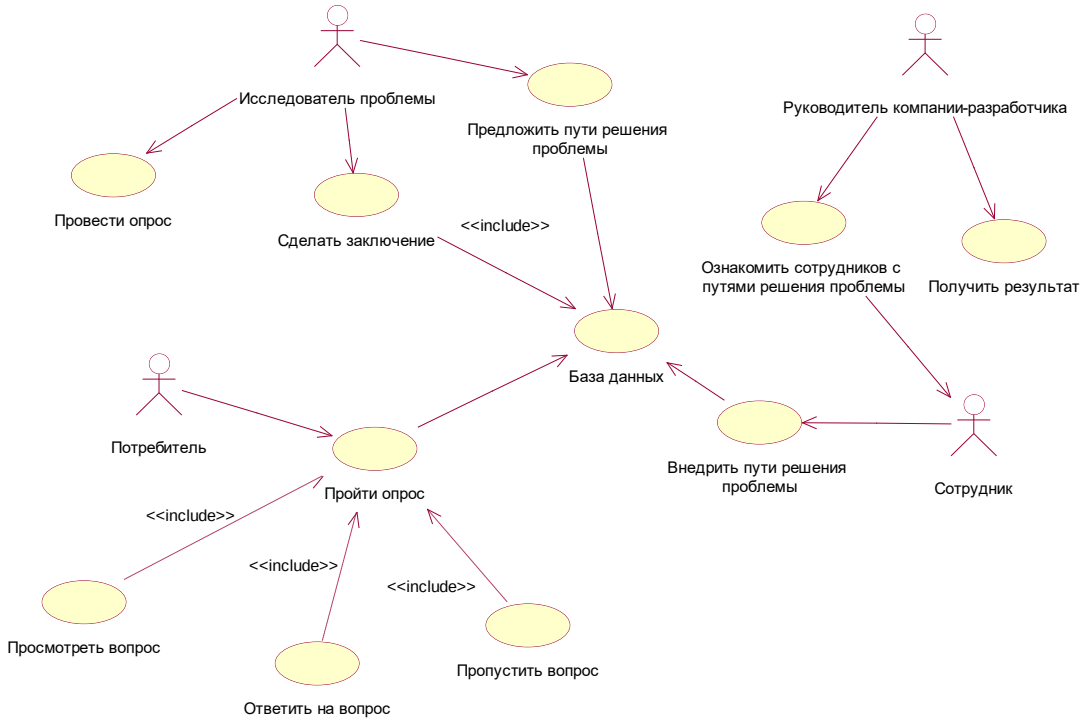


Рисунок 1 – Диаграмма прецедентов разрабатываемой системы

Выводы

Таким образом, исследование удовлетворенности и лояльности потребителей мультимедийной продукции является важным элементом в деятельности разработчиков. Поскольку, опираясь на желания потребителей, организация будет способна выполнять их требования в полной мере и предвосхищать их ожидания, точнее, тем самым повышать качество создаваемых продуктов, что позволит достичь не только финансового успеха, но и потребительской приверженности. Следовательно, возникает целесообразность создания системы исследования удовлетворенности и лояльности потребителей для реализации стратегии компании-разработчика мультимедийной продукции. Проведено исследование методик и моделей исследования лояльности потребителей. В ходе анализа было выявлено, что при их комплексном применении можно определить все необходимые показатели для исследования удовлетворенности потребителей продуктом и сервисом компании. Приведена диаграмма прецедентов разрабатываемой системы анализа уровня удовлетворенности потребителей.

Литература

1.Салимова Т.А. Управление качеством. – 2-е издание, М.: Омега -Л, 2008. – 416с.

2. Международные стандарты ИСО серии 9000 [Электронный ресурс] // Студми. Учебные материалы для студентов, 2018: [сайт] – Режим доступа: https://studme.org/53662/menedzhment/mezhdunarodnye_standarty_iso_serii_9000 – Загл. с экрана.

3. Исследование лояльности клиентов [Электронный ресурс] // 4Brain, 2018: [сайт] – Режим доступа: https://4brain.ru/blog/ – Загл. с экрана.

4. Полынская, Г. А. Сравнение методов оценки удовлетворенности потребителей при использовании разных способов сбора данных / Г. А. Полынская, // Управление экономическими системами. – 2014.

5. Турко С.В. Автореферат диссертации на соискание ученой степени кандидата экономических наук "Мониторинг удовлетворенности и лояльности потребителей" // Кафедра стратегического управления и развития человеческих ресурсов Московского государственного университета экономики, статистики и информатики – Москва, 2006.

Кандаурова А.О., Губенко Н.Е. Анализ моделей исследования удовлетворенности и потребителей мультимедийной продукции. *В данной статье рассматриваются понятия удовлетворенности и лояльности потребителей. Определяется их роль в успешности*

разработчиков мультимедийной продукции. Проводится анализ существующих современных методов и моделей исследования лояльности потребителей.

Ключевые слова: *удовлетворенность потребителей, лояльность, мультимедийная продукция, системы управления отношениями с клиентами.*

Kandaurova A.O., Gubenko N.E. Analisis of customer satisfaction survey models for multimedia products. *This article discusses the concepts of customer satisfaction and loyalty. Their role in the success of multimedia product developers is determined. The analysis of existing modern methods and models of consumer loyalty research is conducted.*

Key words: *customer satisfaction, loyalty, multimedia products, customer relationship management systems.*

Планирование собственного игрового проекта на основе анализа существующих проектов подобного жанра и стиля

Кондратова Е.И., Киселёва О.В.
Донецкий национальный технический университет
linwindy.lekond@gmail.com, olgakiseleva_donntu@mail.ru

Кондратова Е.И., Киселёва О.В. Планирование собственного игрового проекта на основе анализа существующих проектов подобного жанра и стиля. *В статье представлено тезисное описание проблем, с которыми может столкнуться разработчик при планировании игрового проекта, сделан краткий обзор игр в установленных жанровых и стилистических рамках, сделаны выводы об общих и отличительных чертах уже существующих проектов, подходящих под установленные рамки. На основе проведённого анализа был выведен план для создания собственного игрового проекта.*

Ключевые слова: *статья, видеоигры, геймдев, планирование, проект, стилистика.*

Введение

В настоящее время игровая индустрия, наравне с кино и книгами, стала неотъемлемой частью современной культуры. Даже если вы не приверженец видеоигр, они так или иначе влияют на развитие технологий и тенденций развлекательной индустрии.

Для начинающих разработчиков игр сейчас создано немало инструментов, позволяющих быстро освоится в индустрии и начать программировать свои игры. Например, движок Unity Engine предоставляет возможность не только быстро освоить нужное ПО, но и создать собственную игру без необходимости программировать её механику – все необходимые скрипты уже включены в опциально загружаемые ассеты.

Однако всем разработчикам так или иначе приходится сталкиваться с проблемой выбора дизайна и истории для будущего игрового продукта. Критически необходимо установить рамки проекта в самом начале его разработки, чтобы получить готовый и успешный проект в сжатые сроки. На данный момент не существует универсального способа, к которому можно прибегнуть в придумывании своего дизайна, однако через анализ существующих продуктов на рынке игровой индустрии можно выделить несколько главных критериев, по которым игре дают оценку.

Анализ существующих игр

Каждый разработчик игр так или иначе знаком с определённым количеством уже существующих игр и имеет свои вкусы и предпочтения, на которых в дальнейшем основывает свой будущий проект. Однако прежде чем приступать к непосредственно созданию проекта, необходимо установить рамки, в которых этот проект будет разрабатываться.

В начале стоит трезво оценивать свои возможности – финансы, время и уровень профессионализма во всех областях, требующихся для создания игрового продукта. Если разработчик не является хотя бы начинающим художником или дизайнером, то не стоит ставит во главу проекта сложную в исполнении графику, а если первый раз имеет дело с созданием игр – не стоит пытаться сделать очевидно сложную MMO RPG.

Важной частью подготовки к созданию своего проекта является выбор направления и анализ существующих игр в этом направлении: что на рынке имеет популярность, какие требования у современной аудитории, какая составляющая игры важна для привлечения игроков.

В данной статье будут преимущественно рассмотрены инди-игры в жанре квест с элементами приключения в 2D стиле, направленные на прохождение сюжета, т.е. имеющие конечную цель.

Для осуществления анализа были выбраны следующие игры:

Oxenfree (Night School Studio, 2016) [1];

Night in the Woods (Infinite Fall, 2017) [2];

Life is Strange: Before the Storm (Deck Nine Games, 2017) [3];

Все перечисленные игры всё ещё пользуются спросом на рынке и в своё время привлекли немалое внимание своей аудитории.

Oxenfree

Инди-игра, выполнена в жанре приключенческий квест.

Механика: Walkie-Talkie игра, что означает, что взаимодействие с окружающим миром и разговор с неигровыми персонажами происходит одновременно, в отличие от большинства игр такого жанра, где диалоги и исследование мира строго разделены и выполняются игроком как отдельные действия. Интерфейс диалоговой системы приведён на рисунке 1.

Особенности:

выборы в диалогах влияют на сюжет и концовку;

все реплики персонажей озвучены, и каждая реплика главного героя (или её отсутствие) является результатом выбора игрока;

важным элементов взаимодействия является карманное радио, которое можно достать в любой момент игры и прокручивать почти независимо от других действий;

игра не просто даёт возможность перепрохождения – она сюжетно подталкивает игроков проходить игру заново, чтобы найти выход из сложившейся ситуации;

существует режим NewGame+, учитывающий результат последнего прохождения и открывающий новые концовки.

Интерфейс:

объекты, доступные для взаимодействия, отмечены кружочком, когда персонаж находится в определённом радиусе от них (см. рис.1);

с помощью кнопки Tab можно вызвать карту местности, на которой отмечено текущее задание и местонахождение персонажей игры см. рис.2);

радио показано как циферблат радиочастот над головой главного персонажа (см. рис.3).

Рисунок 1 – Дизайн диалоговой системы в игре Oxenfree

Рисунок 2 – Карта местности в игре Oxenfree

Рисунок 3 – Дизайн интерфейса карманного радио в игре Oxenfree

Не главной, но важной частью игры являются также коллекционные предметы – аномалии и письма

Адлер, которые игроку предоставляется найти самостоятельно. Аномалии не связаны с сюжетом, но дают насколько интересных кусочков информации, в то время как письма, если найдены, дают игроку дополнительный диалоговый выбор в финале игры.

Night in the Woods

Также инди-игра, выполнена в жанре приключенческий квест. Все персонажи игры – животные, игра выполнена в уникальном векторном стиле.

Механика: игроку позволяется прыгать по горизонтальным поверхностям, и каждый уровень открывается новая территория, доступная для исследования. Разговоры также являются важной частью механики, однако в этой игре они отделены от исследования и происходят как отдельные действия игрока.

Особенности:

- выборы в диалогах влияют только на то, какую часть истории игрок увидит в своём прохождении, а какую – пропустит. Концовка не зависит от выбора игрока;
- реплики персонажей написаны в специальных диалоговых облаках (рис.4), озвучка отсутствует;
- не все реплики главного персонажа опциальны, некоторые прописаны заранее;
- некоторые игровые локации становятся доступны только на определённой сюжетной развилке, что позволяет игроку переигрывать по крайней мере дважды, прежде чем он посетит все локации.
- Интерфейс:
- выбор реплики происходит в пределах одного диалогового облака;
- интерактивные объекты обозначаются специальным кругом с символом глаза или диалогового облака, в зависимости от того, какое действие можно с объектом совершить;
- через кнопку Tab доступен блокнот, в котором через карикатурные рисунки отмечается прогресс и опыт персонажа, приобретённый в процессе прохождения (см. рис.5).
-

Рисунок 4 – Диалоговое окно в игре Night in the Woods

Рисунок 5 – Блокнот в игре Night in the Woods

В игре не присутствуют коллекционные предметы как таковые, однако есть некоторые собирательные элементы, непосредственно связанные с сюжетом.

Life is Strange: Before the Storm

Данная игра не принадлежит к жанру инди, однако имеет особенности, схожие с описанными выше играми, и может дать дополнительные интересные решения для создаваемого проекта. Классифицируется как приключенческое интерактивное кино.

Механика: игрок может исследовать мир, взаимодействовать с его элементами и общаться с неигровыми персонажами. В каждом эпизоде существует возможность кастомизации внешнего вида игрока, а именно – элементов одежды.

Особенности:

- выборы в диалогах влияют на сюжет и на концовку;
- реплики персонажей полностью озвучены и сопровождаются анимацией, как и мысли главного персонажа относительно некоторых элементов окружающего мира;
- большинство реплик главного персонажа не являются опциальными и прописаны заранее, часто игрок может выбирать только ключевые моменты;
- присутствуют некоторые сюжетные сцены, на наличие или отсутствие которых влияют диалоговые выборы игрока;
- в игре реализована уникальная механика диалогов «backtalk», в котором каждая реплика персонажа имеет вес, и только совокупность всех реплик может принести желаемый результат.
- Интерфейс:
- на главном экране игрок может видеть собранные им предметы, если таковые есть (их применение заложено в сюжете и не контролируется игроком непосредственно) (рис.6);
- интерактивные предметы подписаны и подсвечены, если выделены (рис.6);
- текущая миссия записана на ладони управляемого персонажа и доступна по зажиму ПКМ;
- по нажатию Tab доступно контекстное меню с блокнотом, телефоном и другой сюжетно важной информацией.

Рисунок 6 – Интерфейс главного экрана игры Life is Strange: Before the Storm

Коллекционным элементом здесь являются графити, места для которых игроку нужно найти самому, однако в одной из вкладок меню Tab находятся подсказки к расположению этих мест.

Проектирование собственной игры

Итак, после анализа представленных игр можно выделить особенности и игровые механики, которые войдут в разрабатываемую игру. В данном случае, выделим общие элементы:

- выборы в диалогах должны влиять на развитие сюжета, давая игроку возможность почувствовать уникальный опыт во время прохождения;
- реиграбельность игры – важный критерий, определяющий, как долго игры останутся актуальны;
- меню паузы и дополнительное контекстное меню всегда различаются и имеют различный метод доступа;
- интерактивные объекты должны быть явно выделены и очевидны для игрока.

Для установки чётко определённых рамок разработки можно воспользоваться специально разработанным для этих целей шаблоном [4]. Документ под названием «One Page Game Design Document Template» был составлен общими усилиями двух разработчиков на форуме Game Dev Underground [5] и содержит в себе несколько главных пунктов, которые нужно определить разработчику прежде, чем приступать непосредственно к созданию игры.

Также, обращаясь к видео основателя Game Dev Underground о том, что делает игру интересной [6], стоит отдельны выделить три особенности хорошей игры:

1. Игра ставит чёткую проблему и предоставляет игроку найти способ её решить.

2. Игра отмечает прогресс игрока и даёт ему почувствовать, что его действия приносят результат.
3. Игра заинтересовывает игрока, а значит, должна чем-то отличатся от рутинных заданий.

Учитывая эти пункты, можно привести пример: при разработке игры о студенте, которому нужно срочно закрыть сессию, разработчику необходимо продумать концепцию так, чтобы она отличалась от реальности. Иначе игра потеряет большую аудиторию в лицах тех студентов, для которых этот опыт слишком свеж в реальности – они просто не захотят сталкиваться с проблемами сессии ещё и в виртуальном пространстве.

С данной ситуацией отлично справилась Life is Strange: Before the Storm, что и является причиной, по которой она была рассмотрена в данной статье. Игра позволяет игроку прочувствовать опыт серьезных проблем, с которыми подавляющее большинство студентов никогда не столкнутся, но при этом оставляет атмосферу студенчества на таком уровне, что каждый может проассоциировать себя с главным персонажем и представить себя на его месте.

Выводы

Таким образом, после проведения анализа уже существующих проектов, были сделаны выводы относительно важных элементов, которыми должна обладать инди-игра с конечной сюжетной целью, а также выделены конкретные пункты, которые необходимо продумать прежде, чем приступать к непосредственной разработке игрового продукта.

Литература

1. Night School Studio [Электронный ресурс] // Oxenfree – Режим доступа: http://nightschoolstudio.com/oxenfree/
2. Steam [Электронный ресурс] // Night in the Woods – Режим доступа: https://store.steampowered.com/app/481510/Night_in_the_Woods/
3. Square Enix [Электронный ресурс] // Life is Strange: Before the Storm – Режим доступа: https://lifeisstrange.square-enix-games.com/en-us/games/before-the-storm
4. Google Docs [Электронный ресурс] / Tim Ruswick, josehzz // One Page Game Design Document Template – Режим доступа: https://docs.google.com/document/d/1npEvqcMZSp0IX2hWw6Qq0WqJVfmVqS_YOGFWnnwfh-A/edit
5. Game Dev Underground [Электронный ресурс] / Tim Ruswick // Game Design Documents – P. 1. – Режим доступа: https://gdu.io/topic/game-design-documents
6. YouTube [Электронный ресурс] / Game Dev Underground // What Makes A Game Fun - 3 Tips To Make A Fun Game – Режим доступа: https://www.youtube.com/watch?v=z-wJNIOZcnU&t=1s

Кондратова Е.И., Киселёва О.В. Планирование собственного игрового проекта на основе анализа существующих проектов подобного жанра и стиля. *В статье представлено тезисное описание проблем, с которыми может столкнуться разработчик при планировании игрового проекта, сделан краткий обзор игр в установленных жанровых и стилистических рамках, сделаны выводы об общих и отличительных чертах уже существующих проектов, подходящих под установленные рамки. На основе проведённого анализа был выведен план для создания собственного игрового проекта.*

Ключевые слова: *статья, видеоигры, геймдев, планирование, проект, стилистика.*

Kondratova Elena, Kiselyova Olga. Planning your own game project based on the analysis of existing projects with similar genre and style. *The article provides a thesis description of problems that the developer may encounter while planning a game project, a brief review of games within the established genre and style framework and conclusions about these existing projects' similar and distinctive features. Based on the performed analysis, a plan for creating one's own game project was made.*

Key words: *article, videogames, gamedev, planning, project, style.*

Обзор онлайн систем для развития логического мышления

Лащенова В.Э., Киселева О.В.
Донецкий национальный технический университет
lerchik71@gmail.com olgakiseleva_donntu@mail.ru

Лащенова В.Э., Киселева О.В. Обзор обучающих логических систем и разработка собственной обучающей логическому мышлению системы. *В статье представлены обзоры сайтов, которые являются самыми популярными при запросе пользователя о развитии своего логического мышления. В соответствии с этим сделаны выводы о достоинствах и недостатках таких систем и предложена идея о создании своей обучающей системы в глобальной сети Интернет.*

Ключевые слова: *разработка, пользователь, логическое мышление, сайт, интерфейс*

Введение

Несмотря на то, что логика появилась задолго до нашей эры, однако она и сейчас является не только частью такой дисциплины как математика, но большинство ученых выделяют логику, в одну большую дисциплину, со своими законами и формулировками. Конечно, нельзя утверждать, что в наше время логика является все той, же что и на этапе своего появления. Данная дисциплина претерпевала, на всем этапе своего развития, различные модификации.

В настоящее время, можно не только изучать логику со стороны ее зачастую сложных для всеобщего понимания законов, но и начинать ее изучать уже с двухлетнего возраста. Это обосновывается большим количеством настольных и интернет-игр, которые захватывают не только детей, но и взрослых.

Стоит отметить, что с развитием Интернет-технологий, на данный момент, конечно все большую популярность в современной жизни занимают Интернет-игры, которые не просто позволяют разнообразить досуг, но и проводить время с пользой за изучением или развитием чего-либо.

Из вышеуказанного можно сделать вывод о том, что развитие логического мышления посредством игры, а если точнее, то Интернет-игр является популярным и требует дальнейшего развития.

Анализ существующих обучающих систем

Каждый разработчик либо программист перед тем как создать свой проект, обязан провести анализ уже существующих систем для того, чтобы выделить основные достоинства и недостатки. Делается это для того, чтобы в свою систему «перенять» все самое лучшее и сделать оптимизацию по максимуму. Этот подход уже на ранних этапах развития проекта позволит избежать дальнейших ошибок в будущем, что существенно сократит финансовые и временные расходы. Также стоит отметить, что должен быть совершен опрос заинтересованных пользователей и проведен ряд тестирований для выявления того, как пользователь представляет себе свое обучение. Делается это для того, чтобы выяснить как хочет обучаться пользователь: посредством игр, тестов, картинок, задач и т.д. Также следует учитывать и возрастные категории, т.е. для каждого возраста – свой интерес в данном вопросе.

Для проведения анализа и обзора были выбраны следующие сайты:

- Викиум;
- 4BRAIN;
- Cognifit.

Викиум

Данный сайт позволяет развивать мышление, логику и внимательность. Интерфейс:

- практически все элементы сайта являются динамическими, т.е. появляются по мере пролистывания сайта вниз;
- все функциональное меню расположено сверху и представляет собой кнопки без ниспадающего меню, что наглядно не нагромождает страницу(рис.1);
- одним из главных достоинств данного сайта является его красочность; это означает, что пользователю долгое время не надоедает смотреть в монитор, но и при этом это положительно сказывается на его психике.

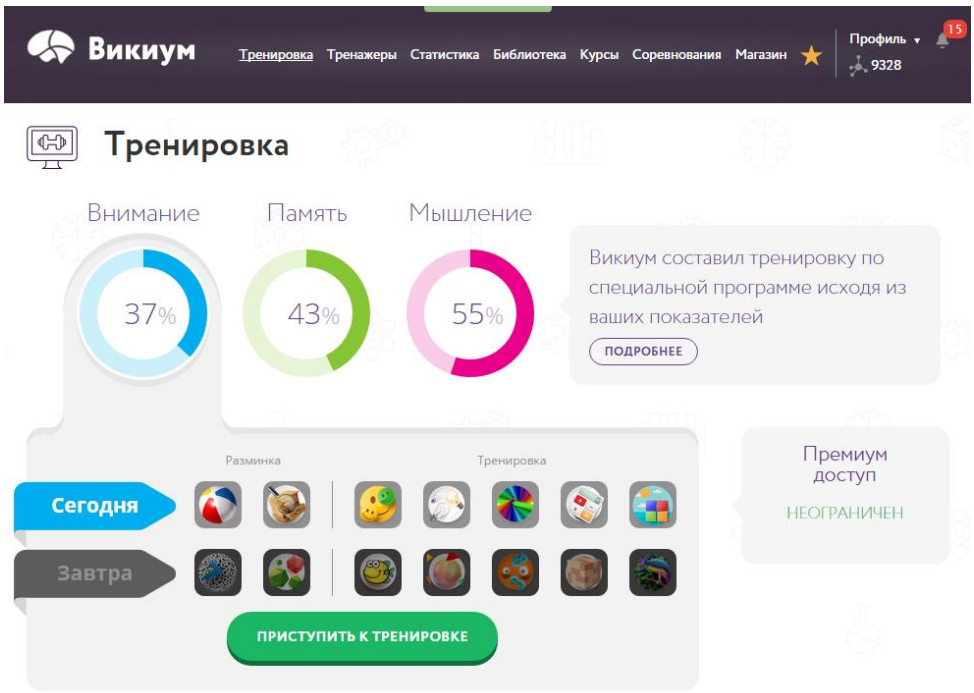


Рисунок 1 – Дизайн обучающего сайта Викиум

Из рисунка 1 можно выделить и главные особенности данного сайта. В первую очередь, это конечно авторизация, которая необходима для того, чтобы у пользователя была возможность просматривать свою статистику обучения, в данном случае, в процентах. При этом учитываются только основные показатели такие как, внимание, память и мышление. Также по мере сложности пройденных тестов формируется рейтинг игроков и уровни с поощрением, что также привлекает пользователей и делает обучение более интересным(рис.2). Процесс обучения осуществляется посредством только лишь игр, без учебной литературы, тестов либо задач, что является недостатком данной обучающей системы, так как такой метод обучения подходит не всем пользователям.

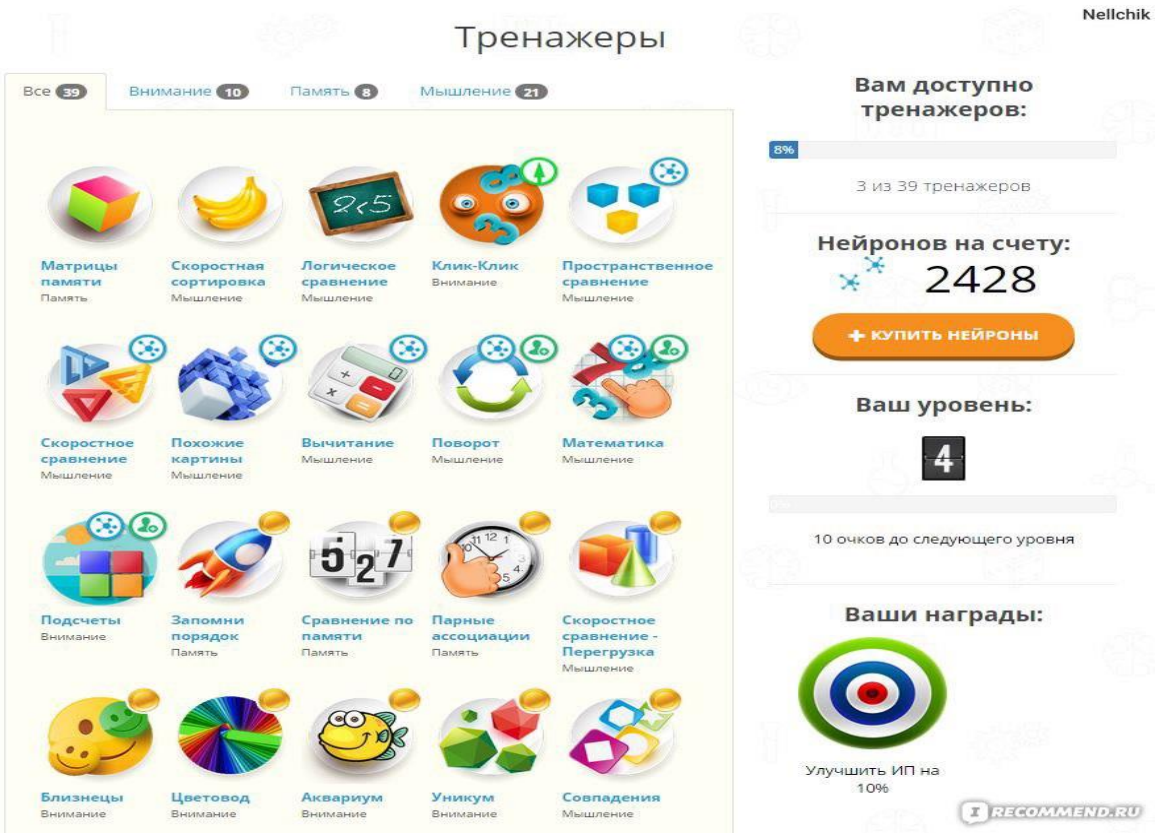


Рисунок 2 – Тренажеры сайта Викиум

4BRAIN

Данный сайт также ориентирован на развитие логического мышления (см. рис.3).

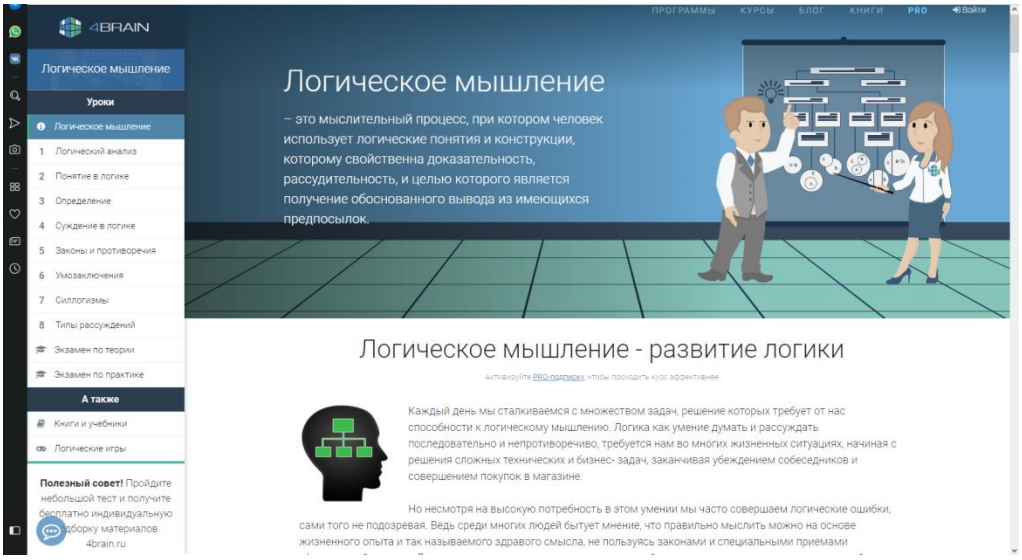


Рисунок 3 – Интерфейс сайта 4BRAIN

Что касается интерфейса, главная страница сайта представляет собой огромное количество информации с разной тематикой, которая просто разбита на блоки. Логические уроки, которые пользователь хочет пройти находятся в конце страницы, однако не все пользователи, увидев такой объем ненужной информации элементарно пролистают сайт до уроков.

слева расположено меню с кнопками без ниспадающих списков, что смотрится довольно компактно и удобно.

Сами уроки представляют собой обычный текст, разделенный на блоки. Это является недостатком, так как глаза пользователя быстро устают от просмотра одной и той же картинки на экране. После того, как пользователь усвоил урок, он может пройти тест по данному уроку, если он авторизирован.

Из вышеуказанного можно сделать вывод о том, что данному сайта не хватает красочности и соответствующего дизайна для каждой тематики для разнообразия. Также стоит отметить и сам процесс обучения, в котором отсутствуют игры либо тесты, что однозначно отвергает аудиторию в качестве детей и подростков.

Cognifit

Данная обучаемая система также, как и предыдущие ориентирована на развитие логического мышления пользователей. Стоит отметить, что система изначально ориентирована на подростков и взрослых людей.

Механика: игрок может исследовать мир, взаимодействовать с его элементами и общаться с неигровыми персонажами. В каждом эпизоде существует возможность кастомизации внешнего вида игрока, а именно – элементов одежды.

Интерфейс:

− наличие комбинированного интерфейса, т.е. присутствуют и функциональные кнопки и ниспадающее меню, что на данном сайте очень гармонирует со всем остальным дизайном (см. рис.4);

− интерфейс имеет единую дизайн-концепцию, сохраняющуюся на всей иерархии сайта.

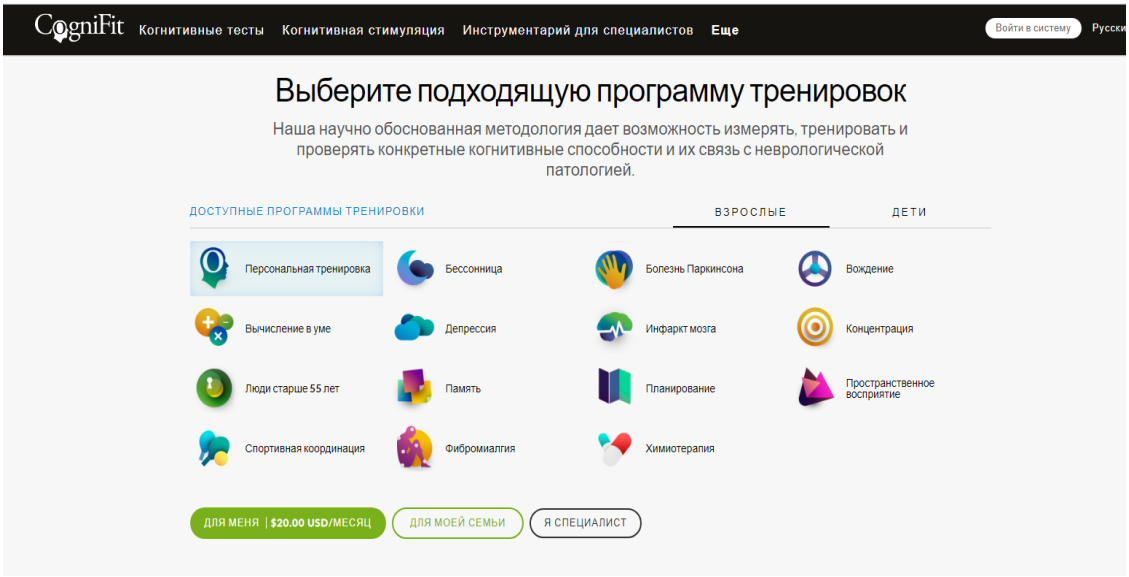


Рисунок 4 – Интерфейс сайта Cognifit

Из достоинств данного сайта следует отметить разнообразные категории тренировок. Это могут быть игры, тренажеры либо же просто статьи (см. рис.5).

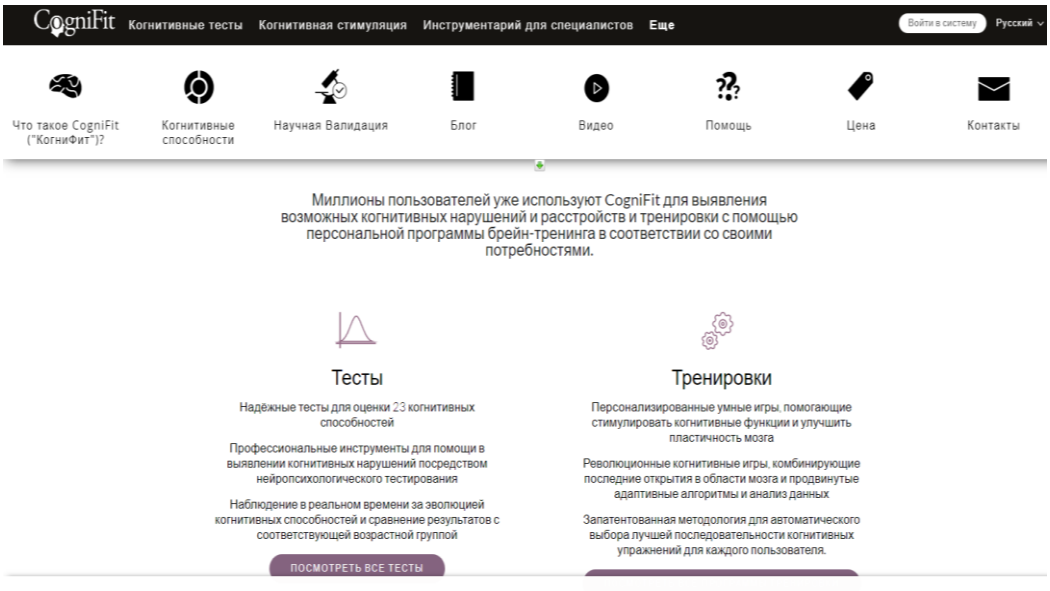


Рисунок 5 – Методы обучения сайта Cognifit

Из вышеуказанного можно сделать вывод о том, что данная система содержит основные виды тренировок, что позволяет ей быть довольно гибкой системой.

Проектирование обучающей логическому мышлению системы

Таким образом, при проектировании собственной обучающей системы можно выделить следующие основные критерии:

- разработка разнообразных методов обучения для разных возрастных категорий;
- наличие статистики, позволяющей анализировать свои результаты в обучении;
- наличие красочного и простого дизайн-интерфейса.

Разработка методов будет заключаться в разном подходе к обучению, в зависимости от того, какой возраст у пользователя. То есть это могут быть игры, тесты или логические задачи.

Наличие статистики необходимо для того, чтобы пользователь видел свои результаты и продвижения в учебе. Например, один и тот же тест он может пройти в начале и в конце месяца с разным результатом.

Красочный и просто дизайн необходим для поддержания эмоционального состояния пользователя, так как его ничего не должно отвлекать либо раздражать. Процесс обучения должен быть приятным и запоминающимся.

Литература

1. Wikium [Электронный ресурс]– Режим доступа: https://wikium.ru
2. 4brain [Электронный ресурс] // lnd – Режим доступа: https://4brain.ru/lnd/index.php?cb=smartread
3. Cognifit INC [Электронный ресурс] – Режим доступа https://www.cognifit.com/ru

Лащенова В.Э., Киселева О.В. Обзор обучающих логических систем и разработка собственной обучающей логическому мышлению системы. *В статье представлены обзоры сайтов, которые являются самыми популярными при запросе пользователя о развитии своего логического мышления. В соответствии с этим сделаны выводы о достоинствах и недостатках таких систем и предложена идея о создании своей обучающей системы в глобальной сети Интернет.*

Ключевые слова: *разработка, пользователь, логическое мышление, сайт, интерфейс*

Lashchenova Valeriia, Kiseleva Olga. Overview of learning logical systems and the development of its own logical learning system. *The article presents reviews of sites that are most popular when a user requests to develop their logical thinking. In accordance with this, conclusions were made about the advantages and disadvantages of such systems and the idea of creating their own training system in the global Internet was proposed.*

Keywords*: development, user, logical thinking, site, interface*

УДК 004

Анализ проблем архитектуры компьютерных игр

Тилинина Н.Ю., Губенко Н.Е.
Донецкий национальный технический университет
n.aprimavista@yandex.ua

Тилинина Н.Ю., Губенко Н.Е. Анализ проблем архитектуры компьютерных игр. *В представленной статье рассматриваются основные проблемы существующих современных методов разработки архитектур компьютерных игр, проводится анализ в результате которого представлены условия создания «хорошей архитектуры».*

Ключевые слова: *архитектура, графика, ПК, разработчик, компьютерная игра, компоненты системы*

Постановка проблемы

Игры представляют собой огромную индустрию разработки программ, как правило, состоящих из миллионов строк кода, однако сам процесс разработки не значительно изменился со времен создания первых программ. Отсутствие промежуточных этапов реализации игры от ее идеи до уровня кода или, иначе говоря, непосредственное кодирование - обычная ситуация в настоящее время, но при таком подходе к задаче успех или неудача проекта полностью зависят от мастерства и опытности разработчиков. Создание архитектуры, которая бы унифицировала взаимодействие между главными подсистемами игры и оставляла при этом возможность масштабирования и гибкость, является необходимым условием для развития индустрии компьютерных развлечений.

«Архитектура — это базовая организация системы, воплощенная в ее компонентах, их отношениях между собой и с окружением, а также принципы, определяющие проектирование и развитие системы» (Стандарт IEEE 1471). Абсолютно любая игра имеет архитектуру вне зависимости от того, проектировалась она или нет. Архитектура может быть восстановлена уже по существующей игре. При создании игры проработка архитектуры может существенно уменьшить объем работ при её реализации, так как исправление архитектуры в общем случае проще, чем исправления кода реализации [1]. Описание архитектуры позволяет скрыть сложность игры при помощи абстракции и разделения ответственности между подсистемами. Наличие описания архитектуры, как правило, упрощает обсуждение и принятие решений, касающихся разрабатываемой игры.

Современные методы и их недостатки

В настоящее время архитектура проектируется и разрабатывается под конкретную игру. Разумеется, разработчики игр могут перенести часть архитектуры или какие-то проектные решения, созданные и спроектированные для одной игры, в другую, однако это возможно не всегда, хотя и является по факту устоявшимся методом проектирования, результаты применения которого зависят от индивидуального опыта разработчика.[2] Таким образом, до сих пор лишь опытным разработчикам удается достигать желаемых результатов, но при этом проект редко сдается в срок и еще реже удается спланировать сам процесс разработки.

Одной из проблем подобного метода создания архитектуры игрового приложения является то, что такие качественные критерии как гибкость и расширяемость редко учитываются при планировании и проектировании системы. Так, например, компании id software при создании новой игры приходилось переписывать практически весь код предыдущей. Что наглядно видно на примере использования архитектуры игры Quake 3 для создания Doom 3. Несмотря на то, что обе игры принадлежат к одному жанру FPS и во многом похожи. Фактически, их единственное отличие заключается в улучшенной графике. С тех пор как развитие игр фактически свелось к улучшению графики, основной причиной переделывания структуры одной игры при создании новой является то, что существующие модели и архитектурные шаблоны не поддаются расширению. Пример проблем, с которыми столкнулась id, безусловно не единичен. Большинство (если не все) компании тратят время на переписывание звуковой системы игры, системы интерфейса пользователя и т.д., просто потому что существующий код невозможно встроить в новую игру.

Методы Кевина Хавкина и Дэвида Астла, которые описаны в книге под названием OpenGL Game Programming - хороший пример типичной «методички» о разработке игр. Книга раскрывает многие тонкостей графики в играх и рассказывает о том, как их можно реализовать с помощью OpenGL API. Здесь освещены аналитическая геометрия, принципы освещения, текстурирования, трансформаций и различные другие темы, посвященные программированию трехмерной графики. После прочтения этой книги читатель будет иметь

основательные познания в области графики и OpenGL API, но использование этих знаний в контексте сложной системы, такой как, например, игра, останется для него загадкой [3].

Получается, что несмотря на то, что охвачены многие технические детали формирования пикселей с помощью OpenGL, практически не содержится информации об условиях создания хорошей и легко модернизируемой архитектуры игры. Методы, используемые в книге, имеют чрезвычайно упрощенную архитектуру. Один единственный игровой объект вмещает в себя все - графику, ИИ, физику и т.д. (см. рис.1)

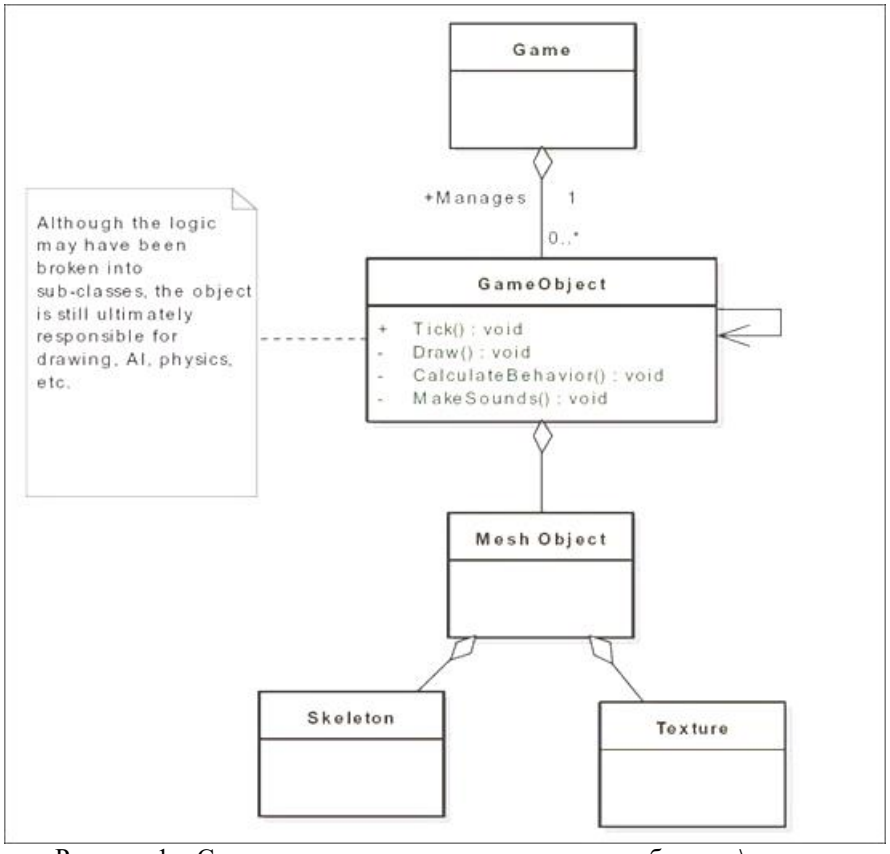


Рисунок 1 – Структурная декомпозиция на уровне объектов\классов

Такой подход к описанию игры возможен в случае рассмотрения какой-либо детали, специфичной части игры, но он является неправильным в случае реальных игр. Простейшего логического деления на уровне классов абсолютно недостаточно для проекта, объем которого стремиться к миллионам строк кода.

Для того чтобы увидеть проблемы, связанные с подобным микроскопическим подходом к архитектуре, рассмотрим некоторые вопросы, с которыми постоянно сталкиваются разработчики компьютерных игр. Во-первых, подобная схема не затрагивает вопросов совместимости, крайне важных при коммерческом подходе к игре: игра должна работать как на многочисленных игровых консолях, так и на ПК. Во-вторых, подобный код лишен возможности повторного использования из-за того, что объект намертво привязан к своему поведению. Таким образом, данная часть архитектуры не является ни гибкой, ни масштабируемой из-за того, так как она представляет собой систему со слишком тесной связью и, изменяя ее, разработчик рискует повлиять на всю систему в целом.

Руди Ракер в своей книге Software Engineering and Computer Games представляет метод с повторно используемой архитектурой. Автор рассматривает создание каркаса игры по принципу "Документ\Вид" приложения. Упор делается на то, что, используя описанный каркас, любой разработчик может создать игру, просто расширяя этот каркас, предлагаемый автором в качестве средства решения всех проблем [4].

Метод создания архитектуры "документ\вид" используется для того, чтобы отделить данные от кода, ответственного за их визуализацию, и вследствие этого появляется возможность изменения данных без переписывания кода. Эта идея предоставляет большую гибкость в вопросах, связанных с игровыми объектами, в тоже время она достаточно ограничена. Искусственный интеллект и физика все еще остаются в ведении объекта, что делает их изменение чрезвычайно затруднительным. Таким образом, в авторской архитектуре система визуализации представляет собой нечто отделенное от объекта, однако при этом продолжается использование механизма прямого, непосредственного взаимодействия между графикой и данными, что делает оба компонента взаимозависимыми и что противоречит выбранному шаблону проектирования "документ/вид".

Роллингс и Моррис, авторы Game Architecture and Design, дают обзор уже существующих архитектурных моделей игровых приложений и пытаются выделить основные элементы игровой архитектуры (см. рис 2).

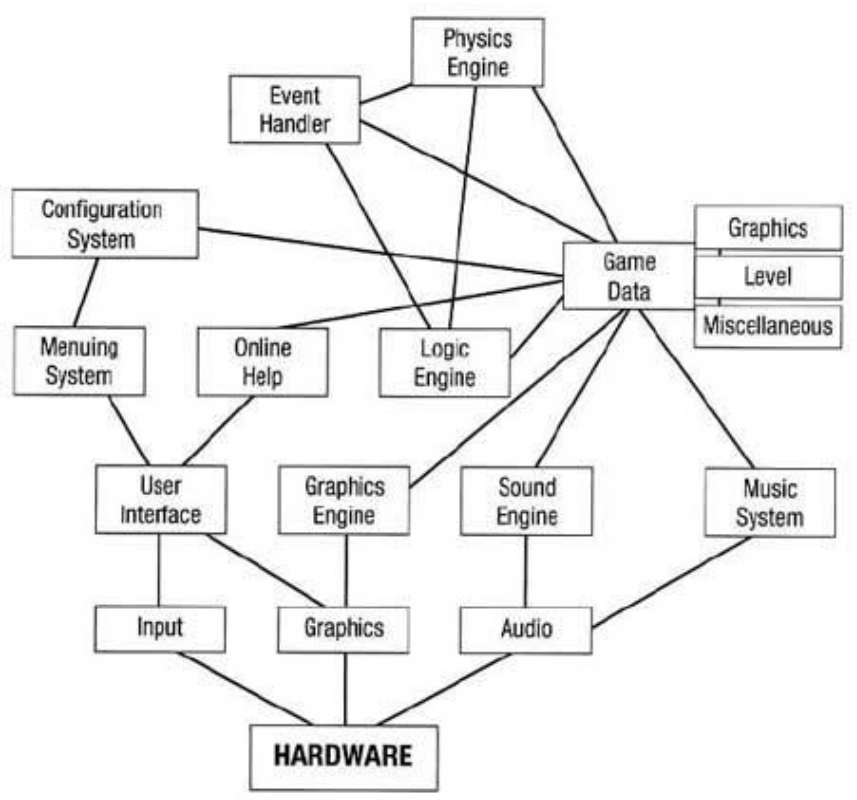


Рисунок 2 – Архитектура Роллингса и Морриса

Игра, построенная по схеме компонентов, представленная на рисунке 2, будет работать, но подобная сеть взаимоотношений и многочисленные связи между компонентами, сильно ограничивают расширяемость системы и возможность ее повторного использования в других проектах. Удобная архитектура должна не только логически разделять систему на подсистемы, она также должна предусматривать легкую замену или модификацию любой подсистемы без перестраивания системы целиком [5].

Отчасти проблемы заключаются в том, что в основе практически любой модели игры лежит принцип объектно-ориентированности, а, следовательно, в каждой игре присутствует чрезмерное количество взаимосвязей между компонентами системы. Игры практически всегда имеют дело с объектами, живущими в виртуальном мире. Игровые объекты обладают собственным поведением, сами отрисовываются на экране, а иногда даже сами говорят. Этот метод кажется логичным и его распространение, по всей видимости, связано со всеобщим признанием объектно-ориентированной парадигмы. Однако ограниченность данного подхода проявляется, как только сложность и запутанность таких функций как отрисовка (rendering) и принятие решений (ИИ) начинает возрастать. В свою очередь такое усложнение может привести к громоздкости игровых объектов в частности и архитектуры вообще.

Переход к сторонним компонентам

Компьютерные игры как разновидность программного обеспечения переживают сегодня революционный этап. Качество игр стремительно приближается к качеству современных фильмов, но играм недостает модульности и возможности привлечения сторонних фирм для создания отдельных компонентов игры, то есть того, что давно используют в индустрии кино. Фильмы создаются несколькими компаниями, каждая из которых специализируется на конкретной сфере: звуке, спецэффектах и т.п. Такой уровень разделения труда позволяет достигать выдающихся результатов. Можно сказать, что практически каждый кадр картины запланирован. В тоже время компьютерные игры только недавно начали движение от полной разработки игры компанией производителем к технологии использования сторонних компонентов, называемой COTS.

Переход к системам, построенным на COTS, является первым шагом на пути масштабного изменения игр, принципов их разработки и планирования. Однако несмотря на то, что метод использования готовых компонентов способен улучшить качество игры и сократить время ее разработки, использование объектно-ориентированного подхода приведет к тому, что компоненты едва ли будут более полезны, чем обычные библиотеки функций, помогающие оперировать объектами. В таком случае игровые объекты будут все еще ответственны за обработку своих же данных, а именно, за рендеринг, за искусственный интеллект, звук, и т.д.

Другими словами, при создании архитектуры, разработчики пытаются использовать преимущества COTS-компонентов, но вместе с этим оперируют слишком большими и сложными игровыми объектами. Что, в свою очередь, приводит к тому, что при разработке объектов создатели игры жестко привязаны к используемым COTS-компонентам (см. рис. 3).

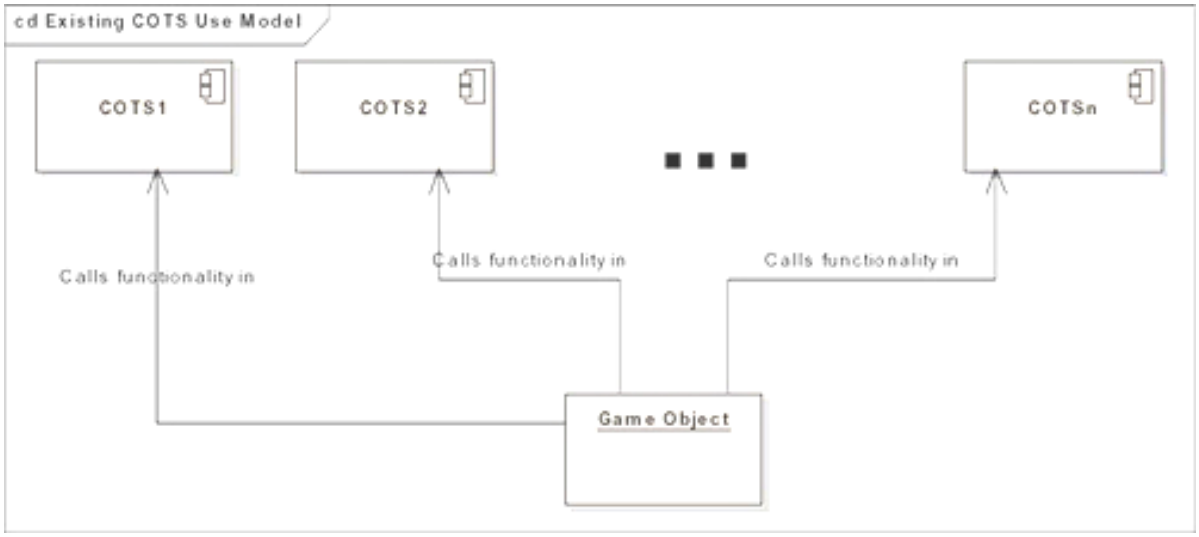


Рисунок 3 – Сочетание метода COTS-разработки и объектно-центрированной модели

К тому же, объектно-ориентированный подход ограничивает возможность повторного использования, даже при применении технологии COTS-компонентов. Код, описывающий объект, всегда остается наихудшим в смысле повторного использования, но при совмещении объектно-ориентированных и COTS-технологий этот код содержит в себе еще и вызовы функций различных специализированных компонентов. Таким образом, при переходе на новый проект разработчики часто вынуждены переписывать код вследствие подобного взаимодействия объекта и компонентов. Даже хорошо спланированная архитектура с разработанной иерархией классов не способна уменьшить риск переписывания кода из-за того, что объекты остаются жестко связанными с используемыми COTS-компонентами.

Применение готовых игровых «движков»

К настоящему моменту в развитии игровой индустрии можно выделить общую тенденцию совмещения всех COTS-компонентов в игровом ядре, также известном как игровой «движок». Компании, которые разрабатывают игры покупают очень мощные и эффектные игровые «движки», что позволяет им создавать игру, используя уже проверенный и надежный каркас. Этот прием является выдающимся примером повторного использования кода, вместе с тем, возможными последствиями таких покупок могут быть ограничения возможностей разработчиков потому, что сторонний движок может не учитывать особенностей игры. Использование игровых «движков» создает ограничения через архитектуру «движка». Они обычно создаются под конкретную игру, и это сказывается на их архитектуре. Например, попытка использовать Unreal Engine, который подходит игре от первого лица, для создания футбольного симулятора закончится неудачей.

Решением данных проблем является проектирование игр на более высоком уровне абстракции, нежели на уровне игрового «движка». Из этого вовсе не следует, что коммерческий движок с возможностью повторного использования не может быть создан с помощью предлагаемой архитектуры. Тем не менее нужно понимать разницу между архитектурой и полностью законченной системой.

Условия хорошей архитектуры

Таким образом, с учетом представленных выше проблем, можно сформулировать список вполне разумных и универсальных критериев, которые помогут в создании хорошей архитектуры компьютерной игры:

1. Эффективность системы. В первую очередь программа, конечно же, должна решать поставленные задачи и хорошо выполнять свои функции, причем в различных условиях.
2. Гибкость системы**.** Любое приложение приходится менять со временем — изменяются требования, добавляются новые. Чем быстрее и удобнее можно внести изменения в существующий функционал, чем меньше проблем и ошибок это вызовет — тем более гибкая и конкурентоспособная система.
3. Расширяемость системы. Возможность добавлять в систему новые сущности и функции, не нарушая ее основной структуры. На начальном этапе в систему имеет смысл закладывать лишь основной и самый необходимый функционал. Но при этом архитектура должна позволять легко наращивать дополнительный

функционал по мере необходимости. Причем так, чтобы внесение наиболее вероятных изменений требовало наименьших усилий.

4. Возможность повторного использования. Систему желательно проектировать так, чтобы ее фрагменты можно было повторно использовать в других системах.

Приложение следует проектировать так, чтобы изменение его поведения и добавление новой функциональности достигалось бы за счет написания нового кода (расширения), и при этом не приходилось бы менять уже существующий код. В таком случае появление новых требований не повлечет за собой модификацию существующей логики, а сможет быть реализовано прежде всего за счет ее расширения. Именно этот принцип является основой «плагинной архитектуры» (Plugin Architecture).

Выводы

Ознакомившись со статьей можно сделать вывод, что разработка архитектуры является важным этапом в создании любой компьютерной игры. Наличие описания архитектуры, как правило, упрощает обсуждение и принятие решений, касающихся разрабатываемой программы. Следовательно, целесообразно использовать различные, уже готовые, «шаблоны» по ее разработки. Проведено исследование методов и моделей создания архитектуры компьютерной игры, в ходе которого были выявлены основные недостатки современных методик и представлены предложения условий для разработки «хорошей архитектуры».

Литература

1. Создание архитектуры программы или как проектировать табуретку [Электронный ресурс] // Stimul, 2018: [сайт] – Режим доступа: http://www.stimul.biz/ru/ru/lib/articles/mind-maps– Загл. с экрана.

2. Архитектура игры [Электронный ресурс] // gamedev.ru, 2018: [сайт] – Режим доступа: https://gamedev.ru/code/terms/Architecture – Загл. с экрана.

3. Astle D., Beginning OpenGL game programming / Astle D., Hawkins K. - 1-е изд. – Course Technology PTR, 2004. – 337 с.

4. Rucker R., Software Engineering and Computer Games/ Rucker R. - 1-е изд. – Addison-Wesley Professional, 2002. – 642 с.

5. Rollings A., Game Architecture and Design/ Rollings A., Morris D.- 1-е изд. – New Riders, 2003. – 960 с.

Тилинина Н.Ю., Губенко Н.Е. Анализ проблем архитектуры компьютерных игр. *В представленной статье рассматриваются основные проблемы существующих современных методов разработки архитектур компьютерных игр, проводится анализ в результате которого представлены условия создания «хорошей архитектуры».*

Ключевые слова: *архитектура, графика, ПК, разработчик, компьютерная игра, компоненты системы.*

Tilinina N.Yu., Gubenko N.E. Analysis of the problems of computer games architecture. *The present article discusses the main problems of the existing modern methods of developing computer games architectures, analyzes the result of which presents the conditions for creating a "good architecture".*

Key words: *architecture, graphics, PC, developer, computer game, system components.*

УДК 004.89

Отдельные аспекты безопасности информационных систем

Юрченко А.С., Яшаров Д.А., Ефименко К.Н.
Донецкий национальный технический университет,
студент гр. ПМК-16[1], доцент кафедры прикладной математики[2]
Coolmadykaqq@mail.ru, KN_Efimenko@mail.ru

Юрченко А.С., Яшаров Д.А., Ефименко К.Н. Отдельные аспекты безопасности информационных систем. *В работе рассматриваются средства и основные механизмы несанкционированного доступа к информационным системам с помощью SQL-инъекций и способы защиты от SQLi-атак.*

Ключевые слова: *информационная система, защита информации, SQL-инъекция, база данных.*

Введение

Стремительное развитие интернет-технологий оказывает всестороннее влияние на современное общество, в котором важнейшую роль играет информация. Что в свою очередь определяет актуальность вопросов безопасности информации в информационных системах. При рассмотрении безопасности информационных систем обычно выделяют две группы проблем: безопасность компьютера и сетевая безопасность. К безопасности компьютера относят проблемы защиты данных, хранящихся и обрабатывающихся компьютером, который рассматривается как автономная система. Под сетевой безопасностью понимают вопросы, связанные, прежде всего с защитой данных в момент их передачи по линиям связи и защитой от несанкционированного удаленного доступа в сеть. Поэтому обеспечение безопасности в сети является задачей значительно более сложной. Уязвимость информационно-справочных систем обусловлена рядом факторов: большие объемы; большое количество пользователей; анонимность доступа; передача информации. При этом безопасной информационной системой считается система, которая защищает данные от несанкционированного доступа, всегда готова предоставить их своим пользователям, надежно хранит информацию и гарантирует неизменность данных. Таким образом, безопасная система по определению обладает свойствами конфиденциальности, доступности и целостности.

Цель работы – анализ отдельных аспектов проблемы обеспечения информационной безопасности в электронных информационных системах (защита серверов баз данных от SQL-инъекций).

SQL-инъекция

В настоящее время широкое распространение получил такой способ несанкционированного доступа к базам данных информационных систем, как внедрение SQL-кода или SQL-инъекция (от англ. SQL injection). Он представляет собой способ взлома сайта или программы, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода. Внедрение SQL, в зависимости от типа используемой СУБД и условий внедрения, может дать возможность атакующему выполнить произвольный запрос к базе данных (например, прочитать содержимое любых таблиц, удалить, изменить или добавить данные), получить возможность чтения и/или записи локальных файлов и выполнения произвольных команд на атакуемом сервере. Атака типа внедрения SQL (SQLi-атака) может быть возможна из-за некорректной обработки входных данных, используемых в SQL-запросах [1-5].

При SQLi-атаках нарушители обычно используют уязвимости web приложений для получения несанкционированного доступа к закрытым данным организации, путем ввода кода в поля формы web-сайта. Таким образом, SQL-инъекция – это использование отсутствия проверки корректности ввода при передаче SQL команд через web приложение на выполнение базе данных. Нарушители пользуются тем, что программисты часто используют простую конкатенацию SQL команд и пользовательских параметров, для чего вместо этих параметров они пишут свои команды. В результате нарушитель может выполнить произвольную SQL команду (или команды) в базе данных через web приложение.

SQL-инъекции обычно начинаются с поиска уязвимостей приложения, в котором пользовательские параметры используются в SQL запросах к базе данных. Атаки, преследующие четко определенную цель, используют найденные уязвимости до тех пор, пока не будет выяснена структура используемой базы данных. Их цель заключается в получении закрытой информации, хранящейся в базе данных. На практике этот процесс, как правило, автоматизируется с помощью широко известных программ, позволяющих нарушителю быстро и без особого труда определить и использовать уязвимости приложения.

В итоге злоумышленники могут получить информацию об интеллектуальной собственности организации, о счетах пользователей или другую конфиденциальную информацию. Также успешная SQL-

инъекция может дать нарушителю возможность украсть пароль администратора, что позволит получить полный контроль над приложением. В других случаях взломанный сайт может содержать добавленный нарушителем код, из-за действия которого посетители будут скачивать вредоносное ПО. SQLi-атаки также позволяют проводить манипуляции с данными, тем самым, предоставляя возможность, например, проводить дефейс web-сайта.

Средства SQLi-атак

Несмотря на то, что методы SQLi-атак постоянно совершенствуются, проведение самих атак не требует никаких базовых знаний. При этом высокая частота атак является признаком использования средств автоматизации. Основными утилитами для проведения SQL-инъекции являются Sqlmap и Havij [3-5].

Sqlmap – свободно распространяемое программное обеспечение. Данная программа автоматизирует процесс поиска и использования уязвимостей для SQL-инъекции, а также позволяет получить контроль над сервером баз данных. Sqlmap позволяет определять используемую БД, получать информацию из базы данных, получать доступ к файловой системе, а также выполнять команды в операционной системе через внеполосное подключение. Эта программа поддерживает пять методов SQL-инъекции: логический вслепую, по времени вслепую, на основе ошибок, с помощью оператора UNION и на основе вложенных запросов.

Havij – простая программа для автоматизации SQLi-атак с оконным интерфейсом (рис. 1). Ее возможности совпадают с возможностями предыдущей программы, единственное отличие – более дружественный интерфейс. Havij распространяется itsecteam, Иранской компанией, занимающейся безопасностью. Havij внедряет выражение «UNION ALL SELECT», и далее продолжает добавлять поля в union-запрос, чтобы выяснить число используемых столбцов. Каждое выражение выбирает статическую «случайную» шестнадцатеричную строку, содержащую различимую последовательность символов. Это позволяет легко определять строки при отображении web страницы.

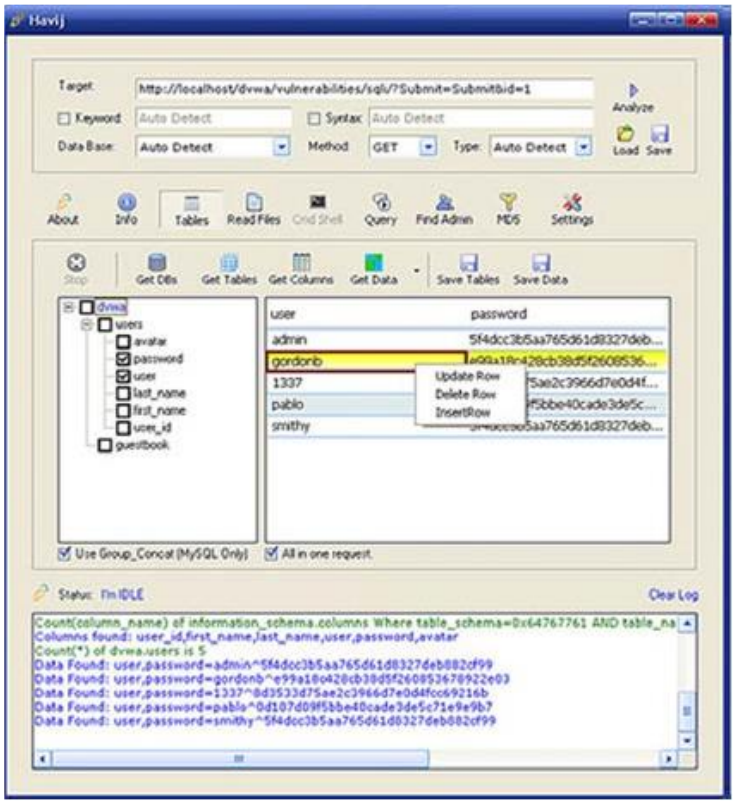


Рисунок 1 – Рабочее окно программы Havij

Основные механизмы SQLi-атак

Методика атак типа внедрение SQL-кода заключается в поиске скриптов, уязвимых для атаки. Злоумышленник изучает поведение скриптов сервера при манипуляции входными параметрами с целью обнаружения их аномального поведения. Манипуляция происходит всеми возможными параметрами [3-5]:

- данными, передаваемыми через методы POST и GET;
- значениями [HTTP-Cookie];
- HTTP_REFERER (для скриптов);
- AUTH_USER и AUTH_PASSWORD (при использовании аутентификации).

Как правило, манипуляция сводится к подстановке в параметры символа одинарной (реже двойной или

обратной) кавычки.

Аномальным поведением считается любое поведение, при котором страницы, получаемые до и после подстановки кавычек, различаются (и при этом не выведена страница о неверном формате параметров).

Принцип атаки внедрения SQL заключается в следующем. Допустим, серверное ПО, получив входной параметр id, использует его для создания SQL-запроса. Рассмотрим следующий PHP-скрипт:

```
$id = $_REQUEST['id'];
$res = mysqli_query("SELECT * FROM news WHERE id_news = " . $id);
```

Если на сервер передан параметр id, равный 5 (например, так: http://example.org/script.php?id=5 (недоступная ссылка)), то выполнится следующий SQL-запрос:

```
SELECT * FROM news WHERE id_news = 5
```

Но если злоумышленник передаст в качестве параметра id строку –1 OR 1=1 (например, так: http://example.org/script.php?id=-1+OR+1=1 (недоступная ссылка)), то выполнится запрос:

```
SELECT * FROM news WHERE id_news = -1 OR 1=1
```

Таким образом, изменение входных параметров путём добавления в них конструкций языка SQL вызывает изменение в логике выполнения SQL-запроса (в данном примере вместо новости с заданным идентификатором будут выбраны все имеющиеся в базе новости, поскольку выражение 1=1 всегда истинно – вычисления происходят по кратчайшему контуру в схеме).

Другим способом внедрения SQL-инъекции является внедрение в строковые параметры. Предположим, серверное ПО, получив запрос на поиск данных в новостях параметром search_text, использует его в следующем SQL-запросе (параметры экранируются кавычками):

```
$search_text = $_REQUEST['search_text'];
$res = mysqli_query("SELECT id_news, news_date, news_caption, news_text, news_id_author
        FROM news WHERE news_caption LIKE('%$search_text%')");
```

Сделав запрос вида http://example.org/script.php?search_text=Test (недоступная ссылка) мы получим выполнение следующего SQL-запроса:

```
SELECT id_news, news_date, news_caption, news_text, news_id_author FROM news
  WHERE news_caption LIKE('%Test%')
```

Но, внедрив в параметр search_text символ кавычки (который используется в запросе), мы можем кардинально изменить поведение SQL-запроса. Например, передав в качестве параметра search_text значение '**)+and+(news_id_author='1**, мы вызовем к выполнению запрос:

```
SELECT id_news, news_date, news_caption, news_text, news_id_author FROM news
  WHERE news_caption LIKE('%') and (news_id_author='1%')
```

При выполнении SQLi-атак обычно используются следующие механизмы.

1. Непосредственная манипуляция запросом. Простой способ для нарушителя изменить поведение приложения – передать любой фильтр на результат SQL-запроса, созданного с помощью пользовательских параметров. Нарушитель может заменить фильтр, сгенерированный приложением, на фильтр, всегда возвращающий true (или false, в зависимости от смысла запроса). Часто это можно сделать, добавив в запрос логическое выражение с известным значением параметра, например, “OR 1=1” чтобы получить true или 1 или “/**/and/**/=3”, чтобы получить false. Данный тип векторов, как правило, используется, чтобы получить информацию об уязвимостях приложения, которые могут быть использованы SQL-инъекцией.

2. Определение структуры базы данных. Более сложные атаки на базу данных, используемую приложением, основываются на знании структуры таблиц. Предположим, что приложение выводит информацию об ошибках, тогда одним из способов определить количество столбцов таблицы может быть использование оператора Order By. База данных вернет ошибку, когда параметр упорядочения – не будет являться существующим столбцом. Некоторые нарушители начинают подбор со значения 10000: “Order by 10000”. Так как 10000 – номер не существующего столбца, нарушитель надеется получить ошибку. После ошибки он уменьшает подставляемое значение (например, “Order by 50”) до тех пор, пока команда не выполнится без ошибки. Обычно для максимальной эффективности используется бинарный алгоритм поиска. В конце выполнения алгоритма нарушитель знает число столбцов в таблице, последний параметр запроса является им. Процесс продолжается запросами к метаданным базы данных до тех пор, пока не будут найдены имена таблиц и столбцов, которые могут содержать ценную информацию. Пример данной атаки содержит следующий пользовательский параметр:

“) and 1=convert(int,(select top 1 table_name from information_schema.tables))”

3. SQL-инъекция с помощью UNION SELECT. В данном типе атак используется оператор UNION SELECT. Этот оператор позволяет получать общие результаты двух различных SQL запросов, не имеющих ничего общего. При успешном исходе нарушитель может получить информацию из определенной таблицы, даже если приложение не было разработано для работы с этой таблицей. Некоторые из проведенных атак просто выполняли проверку на возможность выполнения UNION SELECT. Например, подставляли в значение параметра строку “UNION SELECT 12345”. Приведем более опасный по своим последствиям пример. Следующее значение было использовано на странице форума для получения персональной информации пользователя, идентификатор которого был известен нарушителю.

“XXXX%’) UNION SELECT MEMBER_ID, M_STATUS, M_NAME + ‘/’ + M_EMAIL + ‘/’,

M_LEVEL, M_EMAIL, M_COUNTRY, M_HOMEPAGE, M_ICQ, M_YAHOO, M_AIM. M_TITLE, M_POSTS, M_LASTPOSTDATE, M_LASTHEREDATE, M_DATE, M_STATE FROM FORUM_MEMBERS WHERE (M_NAME LIKE“

В другом варианте, нарушитель мог провести данную атаку, если бы он знал, что используемое приложение является достаточно популярным, и у него был бы опыт взлома подобных систем. В данном примере атака основывается на опубликованной структуре Snitz Forums 2000, открытой системе информирования.

4. Основанная на времени слепая SQL-инъекция позволяет преодолевать некоторые виды защиты, используемой web-администраторами. Иногда при выполнении SQLi-атаки в качестве ответа сервер передает ошибку базы данных, связанную с неверным синтаксисом запроса. Для предотвращения раскрытия информации о базе данных в приложении при возникновении ошибки отображается специальная страница. Это затрудняет проведение SQL-инъекции, но не делает его невозможным. Слепая SQL-инъекция – техника для раскрытия данных из БД с помощью серии SQL запросов, возвращающих true или false. Нарушитель может проверить что вернул SQL запрос несколькими способами. Одним из этих способов является использование трудоемких операций, например, “waitfor delay”, которая отложит ответ сервера, если выражение возвращает true. Стоит отметить, что инъекция “waitfor delay” может быть использована для увеличения потребления ресурсов приложением и вызвать отказ в обслуживании.

5. Обход простой проверки параметров. Перед созданием запроса к базе данных в приложениях осуществляется проверка пользовательских параметров, но если она носит формальный характер, нарушитель может легко обойти ее, например, добавлением незначащих комментариев (обозначается /*and*/) в параметр (1’/**/aND/**/’8’=’3), постоянным переключением между верхним и нижним регистрами (x’ wAiTfOr dELay ‘0:0:20’) или использованием функций вроде concat() и char(), которые позволяют выполнять внедряемую команду в обход сигнатурного анализатора. Еще одним часто используемым способом является кодирование внедряемой строки с помощью ASCII, например,

1 DeCLARe @x varchar(99) set @x=0c77616974666 0x77616974666f722064656c61792027303a303a323027 exec(@x)--.

Данная команда транслируется базой данных в 1 waitfor delay ‘0:0:20’--.

География источников SQLi-атак

Для проведения запланированных атак нарушители используют ботнеты или взломанные хосты, поэтому географическое местоположение атакующего хоста не обязательно совпадает с текущим местоположением нарушителя, контролирующего хост и управляющего атакой. Тем не менее, с точки зрения безопасности, информация о местоположении может показать высокую концентрацию взломанных хостов, принадлежащих ботнетам, в определенных странах (рис. 2).

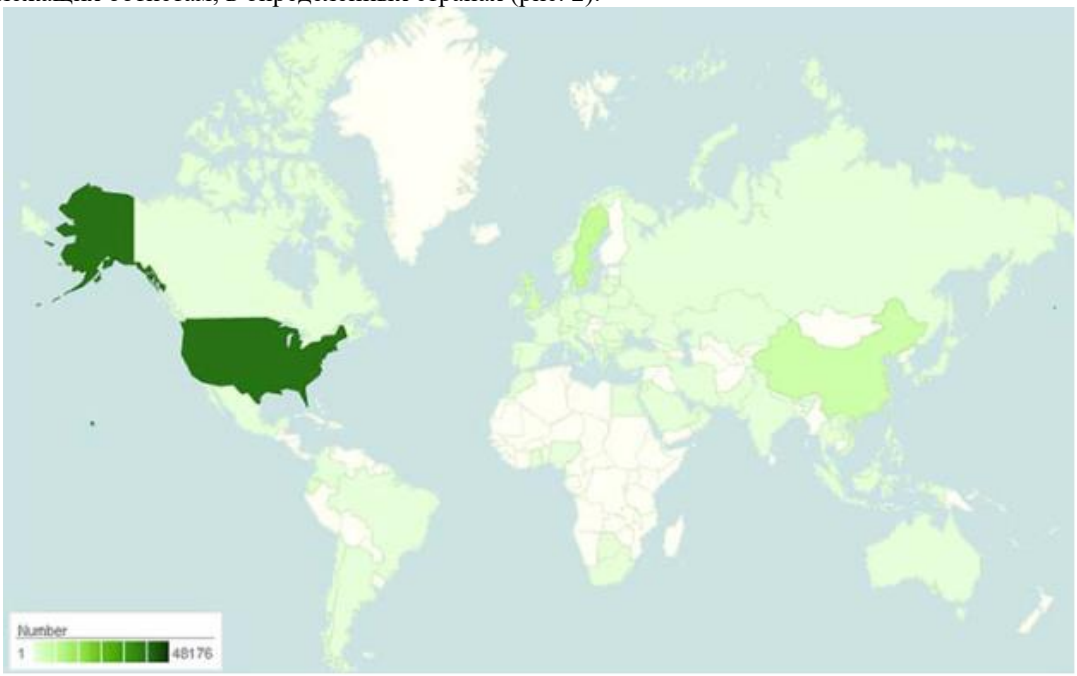


Рисунок 2 – География происхождения SQLi-атак

Выводы

В настоящее время SQL-инъекция стала одним из массовых способов атак на информационные системы. Благодаря общедоступным средствам ее автоматизации даже непрофессионал сможет провести

подобную атаку. Успешная SQL-инъекция позволяет злоумышленнику обойти защиту приложения и получить доступ к закрытой информации. Компании, понесшие большие убытки и испытавшие публичное унижение от SQLi-атак теперь встречаются довольно часто. Но они же являются отличным примером того, что случается при недостаточных мерах безопасности в совокупности с решимостью нарушителя.

Для эффективного решения проблемы защиты информационных баз данных от внедрения SQL-кода необходимо выполнять следующие рекомендации.

Разрабатывать программный код приложения с учетом возможности SQLi-атаки.

Определять SQL-инъекцию, используя знания об уровнях приложений (профили приложений) и баз данных, предварительно настроенных определенным образом. Определение SQL-инъекции должно осуществляться через проверку входных параметров.

Определять признаки, характерные для работы хакерских утилит. В реальной жизни SQLi-атаки в основном проводятся автоматически с помощью специальных программ. Существуют различные механизмы для определения использования данных программ. Например, политики с использованием метрик, или принудительная проверка всех ответов тонких клиентов.

Создать и регулярно пополнять черный список хостов, с которых проводились SQL-инъекции. Данный способ позволит быстрее определять и блокировать нарушителей. Благодаря данным исследований известно, что пиковый период активности хостов, начинающих SQLi-атаки, весьма мал, поэтому важно постоянно обновлять список из разных источников.

Литература

1. Свободная энциклопедия «Википедия» [Электронный ресурс] / Интернет-ресурс. – Режим доступа: ru.wikipedia.org. - Загл. с экрана.

2. Словари и энциклопедии на Академике [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://dic.academic.ru/dic.nsf/ruwiki/746725. - Загл. с экрана.

3. Форум «Хакер.Ru» [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://xakep.ru/2003/07/14/19146. - Загл. с экрана.

4. Проект «Find-XSS.net» [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://find-xss.net/news/article/what-is-sql-injection/. - Загл. с экрана.

5. Информационный портал по безопасности. [Электронный ресурс] / Интернет-ресурс. – Режим доступа: https://www.securitylab.ru/analytics/407944.php. - Загл. с экрана.

Юрченко А.С., Яшаров Д.А., Ефименко К.Н. Отдельные аспекты безопасности информационных систем. *В работе рассматриваются средства и основные механизмы несанкционированного доступа к информационным системам с помощью SQL-инъекций и способы защиты от SQLi-атак.*

Ключевые слова: *информационная система, защита информации, SQL-инъекция, база данных.*

Yurchenko A.S., Yasharov D.A., Efimenko K.N. Some aspects of information system security. *The paper discusses the means and basic mechanisms of unauthorized access to information systems using SQL-injections and ways to protect against SQLi-attacks.*

Key words: *information system, information security, SQL-injection, database.*

СЕКЦИЯ «СИСТЕМНЫЙ АНАЛИЗ И ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ»

УДК 004.056.5

Application of digital watermarks for development graphic password system

Shevlyakov A., Yefremchenko I., Gubenko N.
Donetsk National Technical University
name.ass@gmail.com, efr-inessa@yandex.ru

Shevlyakov A., Yefremchenko I., Gubenko N. Application of digital watermarks for development graphic password system. *The use of digital watermarks for the development of graphical password systems. This article analyzes the methods of steganography, which allow increasing the resistance of systems to methods of hacking systems with password authentication of users. The user authentication method based on digital watermarks is analyzed.*

Key words: *authentication methods, graphic password, digital watermark, authorization, information system, user.*

Formulation of the problem

With the advent of monitors and various devices with touch screens, graphic password systems are being developed that are created to rid the user of complex passwords and simplify authorization, but at the same time secure access to resources. As you know, the most vulnerable security link is the user himself, who does not always remember several complex passwords. Until now, in most cases, a well-known word or name is used, at best supplied with several additional digits. Vocabulary attacks, as well as the system of automatic selection of all possible combinations of characters, allow you to open such a password in a short period of time. Thus, the problem of security of protected resources when passing the identification / authentication of users by means of graphical password systems remains relevant. The solution to this problem can be achieved through the use of shorthand methods that will increase the resistance of systems to most known attacks and hacking methods on password systems.

The purpose of the article is to analyze the methods of using digital watermarks to improve graphic password systems.

Digital watermarks

Currently, computer steganography includes several areas of study:

1. Embedding information for the purpose of its hidden transfer;
2. Embedding digital watermarks, in particular to protect the copyright of electronic products, such as video, audio and image files in electronic form;
3. Embedding titles;
4. Embedding identification numbers.

Digital watermarks are of the greatest interest. They are used both for marking electronic files and for embedding and transmitting various information via communication channels [1].

Digital watermarks can also be of three types:

- fragile;
- semi-fragile;
- robust.

Fragile digital watermarks are destroyed with a slight modification of the filled container. They are used to authenticate signals. The difference from electronic digital signature means that fragile digital watermarks do allow some modification of the content. This is important for the protection of multimedia information, as a legitimate user may, for example, wish to compress an image. Another difference is that fragile digital watermarks should not only reflect the fact of the container modification, but also the type and location of this change.

Semi-fragile digital watermarks are stable with respect to one impact and unstable with respect to others. Semi-fragile digital watermarks are specifically designed to be unstable with respect to a certain kind of operation. For example, they can allow performing image compression, but prohibiting cutting out of it or inserting a fragment into it.

By robustness is meant the stability of digital watermarks to various kinds of impacts on the stegocontainer. Robust digital watermarks can be of 3 types. These are digital watermarks that can be detected by everyone, at least on one side, or it can be digital watermarks that are difficult to modify or extract the content (container) [2].

Thus, it is possible to single out the actual task of applying digital watermarks, which consists in authenticating the user, provided that it is resistant to the imposition of false messages by the intruder and the effects of random and deliberate errors in communication channels. In the search for a solution to this problem, it is necessary to use authentication methods based on digital watermarks and the use of both one and several types of digital watermarks.

The user authentication model based on the digital watermarks of the system is shown in Figure 1:

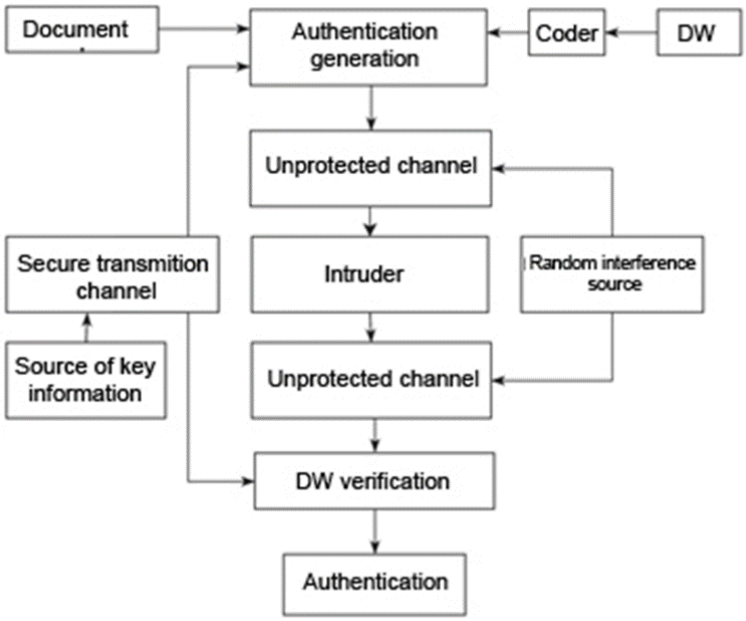


Figure 1 – User authentication model based on digital watermarks

User authentication algorithm based on digital watermarks:

1. The sender creates a document, imposes a digital watermark, individual for each sender, which is converted in the encoder to a convenient form for embedding in a certified message.

2. In the authenticated message builder, the watermark structure is embedded in the document using a confidential key.

3. In the communication channel the authenticated message is affected by the intruder, as well as random and intentional interference. As a result of this impact, a modified message is received at the reception to the watermark checker.

4. According to the digital watermark detection algorithm, its assessment is formed. The authenticity of the document is determined in accordance with this assessment.

In this case, the following solutions are possible:

- the authenticity of the message is confirmed;
- the authenticity of the message is not confirmed;
- the fragment is most likely authentic;
- fragment most likely imposed or distorted by transmission interference.

When forming the assessment of watermarks, errors may occur when the recipient of the message finds them.

In this model, under the digital watermarks is meant embedding certain information about the user and his password to a specific resource. Information is encoded and embedded in a sequence of graphic files. The user selects a specific sequence of files and confirms his choice. The system from this sequence extracts digital watermarks, which contain the user name and password, if the sequence is correct, the user is opened the resource and otherwise the access to the protected resource will be closed until the sequence is correct.

However, this model has several disadvantages:

- there are no restrictions on the quantitative choice of the sequence, that is, if, for example, the sequence was chosen incorrectly several times, then user access is automatically blocked;
- for the sustainability of hacking digital watermarks, the system must use several embedding methods.

Conclusion

Thanks to the use of digital watermarks in graphic password systems, attacks on these systems become more

difficult to implement and are significantly different from attacks on frequently used character systems.

This model is effective, however, improvements are needed:

– the model can be improved by introducing restrictions on the quantitative choice of the sequence;

– to implement the model, you can use different types (fragile, semi-fragile, robust) for each file from the sequence, and the methods for embedding can be chosen in an arbitrary sequence.

References

Ссылка на периодическое издание – сборник:

1. Introduction to digital steganography [Electronic resource] // Portal masters of DonNTU: [website]. [2006] – Access mode: http://masters.donntu.org/2006/fvti/khotov/library/stego.htm - Tit. from the screen.

2. Digital watermarks [Electronic resource] // Educational social network KazEdu.kz: [website]. [2009-2017] – Access mode: https://www.kazedu.kz/referat/133581/2 - Tit. from the screen.

3. Shokarev A. V. Graphical passwords using methods of steganography / Shokarev, A. V. // Innovative technologies and Economics in mechanical engineering: proceedings of the VII vseross. Scientific practice. Conf. With international. participation. - Tomsk: Publishing house of Tomsk Polytechnic University, 2009-P. 293-299.

Shevlyakov A., Yefremchenko I., Gubenko N. Application of digital watermarks for development graphic password system. *The use of digital watermarks for the development of graphical password systems. This article analyzes the methods of steganography, which allow increasing the resistance of systems to methods of hacking systems with password authentication of users. The user authentication method based on digital watermarks is analyzed.*

Key words: *authentication methods, graphic password, digital watermark, authorization, information system, user.*

УДК 004.934

Коррекция ошибок при распознавании речи в многопользовательских системах

Горбенко Г.К., Караулов А.С.
Инженерно-технологическая академия ЮФУ
tequilagg161@gmail.com, a.karaul@yandex.ru

Горбенко Г.К., Караулов А.С. Коррекция ошибок при распознавании речи в многопользовательских системах. *Статья посвящена вопросу использования современных технологий распознавания и обработки речи в многопользовательских системах. В работе также рассматриваются аспекты проблемы применения речевых технологий в многопользовательских системах и методы, позволяющие обнаружить ошибки в процессе обработки речи и способы их исправления.*

Ключевые слова: *многопользовательские системы, распознавание речи, обнаружение и коррекция ошибок при распознавании речи, автоматизированные многопользовательские системы с распознаванием речи, коррекция ошибок.*

Введение

Информационные и телекоммуникационные технологии развиваются достаточно быстро и проникают во многие сферы нашей жизни: в сферу услуг, производства, банковскую сферу, в образование и др. В связи с этим, актуальность проблемы доступа людей к информации не вызывает никаких сомнений.

Помощь в решении проблемы доступа к информации оказывают речевые технологии, которые могут использоваться в многопользовательских системах, например, в контактных и информационных центрах. Многопользовательские системы с распознаванием речи могут быть крупномасштабными, которые обрабатывают по несколько тысяч транзакций и распознают по несколько тысяч слов в сутки. Такие системы могут эффективно заменить труд нескольких тысяч операторов. В некоторых странах такие системы уже внедрены и на практике являются довольно успешными.

Создание естественных для человека средств общения с компьютером является в настоящее время важнейшей задачей современной науки, при этом речевой ввод информации осуществляется наиболее удобным для пользователя способом. Распознавание речи является задачей классификации образов акустических характеристик речевых сигналов. В работах по синтезу речи достигнуты достаточно хорошие практические результаты, для многих языков уже сейчас существуют так называемые «искусственные дикторы», которые качественно имитируют произвольную человеческую речь. Но именно в распознавании речи дела обстоят несколько иначе, за последние несколько лет цифровая обработка голоса сделала большой шаг вперед, но при этом трудностей не становится меньше.

Сценарии диалогов с различными процедурами обнаружения и коррекции ошибок

Эти сценарии возникают на этапе проектирования системы, когда принимаются решения относительно структуры и расположения элементов диалога и процедур подтверждения. На рис. 1 приведена классификационная схема речевого человеко-машинного взаимодействия, в основу которой положены следующие признаки: типы элементов диалога, местоположение процедур выявления и корректировки ошибок, способ реализации процедур выявления и корректировки ошибок. Произведенная классификация позволяет охватить самые разные варианты организации взаимодействия клиентов с системой и производить сравнение этих вариантов.

Методика выполнения сравнительного анализа вариантов приведена в [1-3] и состоит в использовании модифицированного варианта применительно к данной задаче принципа *квантификации*, когда варианты сравниваются на множестве одинаковых типовых количественных значений ряда параметров. Например, сравнительная оценка вероятности успешного завершения диалога осуществляется при одинаковых значениях вероятностей правильного распознавания речевых блоков и одинаковом допустимом числе переспросов для сравниваемых вариантов сценариев. Необходимость в таких допущениях диктуется большой размерностью моделей и широким диапазоном изменения параметров. Квантификация является достаточно популярным методом количественного анализа качества интерфейсов.

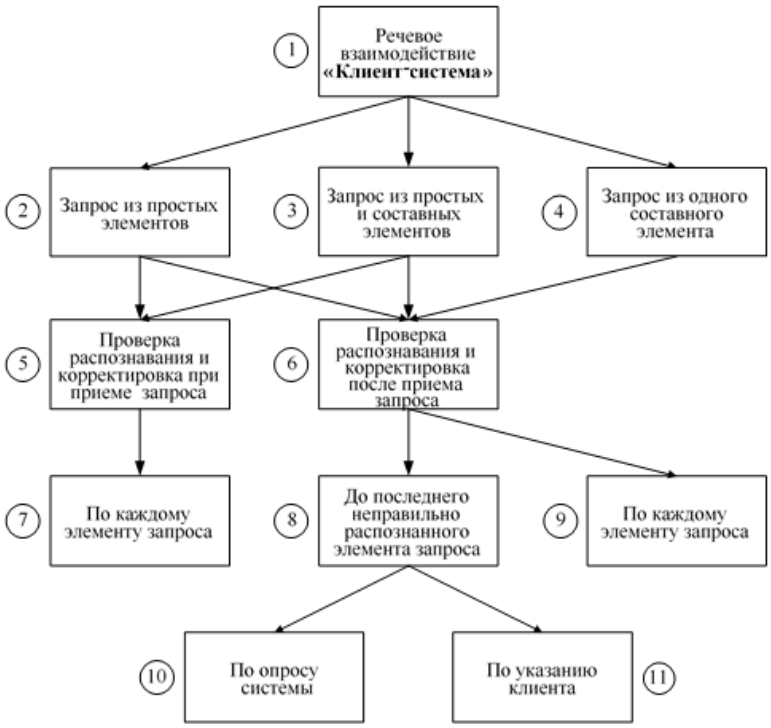


Рисунок 1 – Классификационная схема речевого человеко-машинного взаимодействия

В качестве характеристик для сравнения вариантов выбраны два главных количественных критерия оценки качества речевого диалогового человеко-машинного взаимодействия: вероятность успешного завершения диалога и продолжительность диалога. Поиск оптимального сценария осуществляется на основе минимизации оценок продолжительности диалога с обеспечением заданной вероятности его успешного завершения.

Предложенные в [1-3] модели анализа и формулы вычисления вероятностных и временных характеристик диалогов позволяют:

− определять необходимое число переспросов клиента для получения требуемой достоверности распознавания элементов диалога;

− определять среднее время, затрачиваемое на элемент диалога с учетом рассчитанного для заданной достоверности распознавания числа переспросов;

− вычислять вероятности успешного завершения диалогов различной структуры;

− анализировать и сравнивать между собой различные стратегии управления диалогом по предложенным критериям продолжительности диалога и при обеспечении требуемой вероятности успешного завершения диалогов;

− производить обоснованный выбор сценариев и алгоритмов управления диалогом для конкретных задач на основе разработанной методики.

Обнаружение и коррекция ошибок при распознавании речи

Стохастическая природа процессов, лежащих в основе работы распознавателей, обуславливает возможность появления на их выходе ошибочных результатов. Ошибки при распознавании снижают как полезность приложения, так и степень удовлетворенности пользователей. В связи с этим обнаружение и исправление ошибок распознавания является определяющим свойством речевых приложений. В [4-6] рассмотрены методы выявления ошибок на основе анализа показателя уверенности распознавателя в предлагаемой гипотезе и использования накопленных статистических данных о результатах распознавания.

Метод выявления ошибок на основе анализа показателя уверенности распознавателя в предлагаемой гипотезе состоит в сравнении величины этого показателя с пороговым значением, установленным для вызова процедуры подтверждения. Если показатель уверенности распознавателя в предлагаемой им гипотезе выше порогового значения, то считается, что вероятность ошибки невелика и гипотеза принимается, а диалог переходит на следующий этап. Если вероятность гипотезы ниже порога, то диалог развивается по пути активации процедуры подтверждения. Задача состоит в выборе оптимальной величины порога показателя уверенности в гипотезе.

Определение оптимальной величины порога показателя уверенности в гипотезе состоит в применении принципа минимума ожидаемых затрат, который применительно к данной задаче

формулируется следующим образом: порог уверенности должен быть таким, чтобы суммируемые по всем состояниям ожидаемые затраты для выбранного в соответствии с ним действия были минимальными.

Коррекция ошибок может быть значительно более эффективной, если использовать знания, накопленные в результате сбора и обработки статистических данных об ошибках. В базовых системах распознавания речи предусмотрены средства для сбора статистических данных. Процесс сбора статистики заключается в последовательном прослушивании диалогов, записанных в лог-файлах, и их описании. Описание диалогов состоит в оценке правильности распознавания и внесении дополнительных комментариев. Существующая процедура является трудоемкой и затратной по времени.

В [7] описан метод автоматизированного накопления статистики о правильности распознавания. Метод основан на анализе ответов на запросы подтверждения. Из ответа клиента на просьбу системы подтвердить гипотезу s_j, который может быть сформулирован по-разному («Да», «Да, правильно», «Правильно», «Да-да» и т.д.), извлекается смысл («да» или «нет») и фиксируется вместе со значением возвращаемой гипотезы s_j и показателем уверенности в ней. Ответ «да» соответствует успешному распознаванию данного слова или фразы. Ответ «нет» соответствует ошибке, которая относится к разновидности «замена». После ответа «нет» диалог переходит к фазе повторения ввода. Ответ «да» на запрос подтверждения повторного ввода (новой гипотезы s_i) позволяет зафиксировать не только факт имевшей место ошибки распознавания, но и само искажение: распознавание слова s_i, как s_j. Ответ «нет» на запрос подтверждения повторного ввода инициирует новую процедуру подтверждения, если это предусмотрено алгоритмом. Он также фиксируется и может свидетельствовать о том, что произносимое клиентом слово или фраза отсутствуют в грамматике или имеют ошибку в фонетической транскрипции, особенно если возвращаемая уверенность в гипотезе невелика.

Собранная таким образом статистика может быть использована для проверки фонетических транскрипций, для выявления часто встречающихся «лишних», т.е. отсутствующих в грамматике, слов и для составления таблиц искажений для коррекции ошибок, связанных с искажением слов.

Пусть система пытается распознать произнесенное слово из n слов $s_1, s_2, ..., s_n$, предлагая свой вариант s_0. Пусть также, l — математическое ожидание числа переспросов:

t_j — вероятность того, что система восприняла слово как s_j;

p_i — вероятность того, что было произнесено слово s_i;

q_{ij} — вероятность того, что слово s_i распознано как s_j;

r_{ij} — вероятность того, что произносилось слово s_i при условии, что система приняла его за слово s_j.

В частности, q_{ii} – вероятность того, что слово распознано правильно, а q_{i0} – вероятность того, что слово s_i не было распознано вообще. Вероятности p_i и q_{ij} определяются статистически. Здесь и далее i, j=1, 2, ..., n, $\sum_{i=1}^{n} p_i = 1, \sum_{j=0}^{n} q_{ij} = 1$ для всех i.

Нетрудно видеть, что $r_{ij} = \frac{p_i q_{ij}}{t_j}$, где вероятность $t_j = \sum_{k=1}^{n} p_k q_{kj}$.

Алгоритм, учитывающий статистику искажений, состоит в следующем [8]. Пусть система выдает гипотезу s_j и просит клиента сообщить, правильна ли она. В случае ответа «да» процедура заканчивается. В случае ответа «нет» система выбирает наибольшую вероятность r_{ij} из ряда $r_{1j}, r_{2j},...,r_{nj}$, в котором нет величины r_{jj} , и предлагает слово s_i в качестве новой гипотезы с просьбой подтвердить ее. Если и это не подтверждается, система ищет в ряду $r_{1j}, r_{2j}, ..., r_{nj}$ вторую по величине вероятность и предполагает, что было произнесено второе слово, и т.д.

Математическое ожидание числа переспросов l_j равно $l_j = r_{1j} + 2r_{2j} + 3r_{3j} \ldots + nr_{nj}$, где r_{ij} – вероятности слов, предлагаемых последовательно системой в соответствии с данным алгоритмом при условии, что на первом шаге система распознала слово как s_j. Среднее число переспросов l вычисляется по формуле:

$$l = \sum_{j=1}^{n} t_j l_{ij} = \sum_{j=1}^{n} \left(\sum_{i=1}^{n} p_i q_{ij} \sum_{i=1}^{n} i r_{m_{ij}} \right)$$

Сравнение числа переспросов для алгоритма коррекции ошибок с учетом статистических данных об искажениях слов с числом переспросов для алгоритма без учета вероятностей искажений произведено на примере распознавания произносимых пользователем цифр от 0 до 9. Вероятности q_{ij} были определены экспериментально. Вероятности p_i были приняты равными между собой, т.е. $p_0 = p_1 = ... = p_9 = 0{,}1$.

Среднее число переспросов по первому алгоритму составило 1,08, по второму – 1,18, что свидетельствует о преимуществах алгоритма с коррекцией ошибок с учетом статистики искажений. Таким образом, предложенная математическая модель позволяет оценить, насколько можно сократить среднее число переспросов, если использовать накапливаемую статистику о распознавании слов.

Предложенный метод обнаружения и исправления ошибок путем автоматизированного накопления статистических данных об ошибках и использовании знаний об искажениях слов является новым шагом в решении проблемы ошибок распознавания. Разработчики приложений, не имея возможности вмешаться во внутренние модели распознавателей и вынужденные рассматривать их как черный ящик, могут добиться

достаточно высоких показателей качества работы приложения путем применения предложенного метода по накоплению и использованию знаний о работе распознавателя.

Заключение

Использование речевых технологий даёт возможность увеличить количество пользователей за счет предоставления дополнительного, а в некоторых системах и единственного доступа к функциям этих систем. А это, в свою очередь, формирует новый уровень информационных услуг и повышает эффективность использования многопользовательских систем. Но практическое применение речевых технологий имеет еще ряд проблем, решение и исследование которых является актуальной задачей, потому что использование данных технологий имеет существенно важное теоретическое и практическое значение.

Литература

1. Кипяткова И.С., Ронжин А.Л., Карпов А.А., «Автоматическая обработка разговорной речи». – СПб.: ГУАП, 2013. – 314 с
2. Фланаган Дж. Анализ, синтез и восприятие речи. «Связь». Москва 1998.
3. Рабинер Л.Р., Шафер Р.В. Цифровая обработка речевых сигналов: пер. с анг. – М.: Радио и связь, 1991 г.
4. Насыпный В., Насыпная Г. Система распознавания, понимания смысла, анимационного моделирования и синтеза речи на основе стохастической информационной технологии / В. Насыпный, Г. Насыпная. – М.: Прометей, 2008. – 76 с.
5. Модели, методы, алгоритмы и архитектуры систем распознавания речи // Вычислительный Центр им. А.А. Дородницина РАН. М.: 2006.
6. Ле Н. В. Предварительная обработка речевых сигналов для системы распознавания речи / Н. В. Ле, Д. П. Панченко // Молодой ученый, 2011, №5, 75 с.
7. Компьютерное распознавание и порождение речи. [Электронный ресурс]. – Режим доступа: http://speech-text.narod.ru/chap3.html
8. Ронжин А.Л., Ли И.В. Автоматическое распознавание русской речи // Вестник Российской Академии Наук: научный и общественно-политический журнал, Том 77, Вып.2, 2007. С. 133-138.

Горбенко Г.К., Караулов А.С. Коррекция ошибок при распознавании речи в многопользовательских системах. *Статья посвящена вопросу использования современных технологий распознавания и обработки речи в многопользовательских системах. В работе также рассматриваются аспекты проблемы применения речевых технологий в многопользовательских системах и методы, позволяющие обнаружить ошибки в процессе обработки речи и способы их исправления.*

Ключевые слова: *многопользовательские системы, распознавание речи, обнаружение и коррекция ошибок при распознавании речи, автоматизированные многопользовательские системы с распознаванием речи, коррекция ошибок.*

Gorbenko G.K., Karaulov A.S. Error correction in speech recognition in multi-user systems. *The article is devoted to the use of modern speech recognition and processing technologies in multi-user systems. The paper also discusses aspects of the problem of using speech technologies in multi-user systems and methods that allow detecting errors in speech processing and ways to correct them.*

Key words: *multi-user systems, speech recognition, detection and correction of errors in speech recognition, automated multi-user systems with speech recognition, error correction.*

УДК 519.713

РЗАРАБОТКА И ИССЛЕДОВАНИЕ МЕТОДА ПОСТРОЕНИЯ ФУНКЦИИ ДЕГРАДАЦИИ ПОВЕДЕНИЯ КОНЕЧНОГО АВТОМАТА В РЕЗУЛЬТАТЕ ПЕРЕБРОСКИ ДУГ

Ковалев Д.В., Копытова О.М.
Донецкий национальный технический университет
danilkovalyov2805@gmail.com, omkop@list.ru

Ковалев Д.В., Копытова О.М. Разработка и исследование метода построения функции деградации поведения конечного автомата в результате переброски дуг. *Рассмотрены основные методы построения конечных автоматов, изучены различные формы представления характеристических функций конечного автомата. Исследованы основные виды конечных автоматов.*

Ключевые слова: *конечный автомат, детерминированный конечный автомат, машина, недетерминированный конечный автомат, граф.*

Общая постановка проблемы

Несмотря на стремительную поступь технологий, надежные методы прикладного программирования и искусство программирования развиваются крайне медленно. Путь, позволяющий преодолеть трудности, свойственные разработке программного обеспечения, пролегает через использование подходящих методологий. В эпоху пакетной обработки безраздельно царствовали алгоритмы. Этот период можно назвать эрой «блок-схем». При построении серверных приложений, отвечающих на запросы, большую роль играет «отсутствие состояния» – нет нужды сохранять состояния между двумя последовательными запросами. При построении удачного интерактивного приложения, управляемого событиями, многое зависит от того, продумана ли модель управления состояниями.

Конечный автомат – весьма удобная концепция, которую целесообразно использовать для структурирования приложений. Продуманное применение конечных автоматов облегчает организацию и сопровождение, как логики пользовательского интерфейса, так и логики приложения. Благодаря этому код будет более гибким и надежным, причем это относится к разработке не только мобильных, но и любых других приложений. Поскольку мобильные приложения должны использовать пространство экрана и системные ресурсы эффективно, конечные автоматы оказываются особенно полезными при разработке ПО для таких приложений.

Конечный автомат, как отмечалось выше, является средством формальной структуризации приложения. Вместо того чтобы использовать все переменные приложения в качестве расширенного определения его состояния, конечный автомат создает единственную переменную, в которой хранится информация о состоянии приложения. Такой переменной обычно является элемент перечисления некоторого множества действительных состояний, определяемого глобально или на уровне класса. Мощь подхода, использующего конечные автоматы, обусловлена тем, что он позволяет в явном виде определить действительные состояния для некоторого аспекта вашего приложения и задать соответствующие варианты поведения при переходах приложения из одного состояния в другое.

Конечные автоматы используются для формирования набора взаимосвязанных переменных или вариантов поведения и их логической организации, что облегчает обработку состояний.

Цель исследования

Целью данного исследования является изучение основных методов построения конечных автоматов, а также исследование основных форм представления характеристических функций конечного автомата.

Основная модель конечного автомата

Конечный автомат — это некоторая абстрактная модель, содержащая конечное число состояний чего-либо. Используется для представления и управления потоком выполнения каких-либо команд.

Конечный автомат идеально подходит для реализациииискусственного интеллекта в играх, получая аккуратное решение без написания громоздкого и сложного кода. В данной статье мы рассмотрим теорию, а также узнаем, как использовать простой и основанный на стеке конечный автомат.

Подобно другим теориям, развитие которых побуждается нуждами, науки и техники, теория конечных автоматов имеет дело с математическими моделями, предназначенными для приближенного отображения физических или абстрактных явлений. Значение этой теории состоит в том, что применение ее моделей не ограничивается какой-либо частной областью, а возможно непосредственно для решения проблем практически в любой области исследований от психологии до административного управления и от связи до лингвистики. Идеи и техника теории конечных автоматов используются для решения таких, казалось-бы не связанных, проблем, как исследование деятельности нервной системы человека, анализ английского синтаксиса и проектирование электронных вычислительных машин. В эпоху, когда темпы развития науки сильно зависят от межотраслевого кооперирования, унифицированный характер этой теории представляет несомненную ценность.

Большинство проблем, встречающихся в науке и технике, можно разбить на следующие две категории: задачи анализа, которые состоят в предсказании поведения определенной заданной системы, и задачи синтеза, состоящие в построении системы по заданному поведению. В этой книге предпочтение будет оказано скорее задачам анализа, чем синтеза. Как с точки зрения анализа, так и с точки зрения синтеза удобно переменные, которые характеризуют систему, различать следующим образом: входные переменные, которые представляют собой воздействия, генерируемые другой системой (не подлежащей исследованию), и которые влияют на поведение исследуемой системы, выходные переменные (реакции), представляющие собой те величины, характеризующие поведение данной системы, которые интересуют исследователя, промежуточные переменные – те величины, которые не являются ни входными, ни выходными переменными.

Таблицы, графы и матрицы переходов

После того как для выбранной системы установлены входной алфавит, выходной алфавит и множество состояний, словесное описание системы может быть формализовано при помощи таблицы, графа или матрицы. Таблицы, графы и матрицы – различные формы представления характеристических функций конечного автомата, который описывает данную систему (рис. 1) [1].

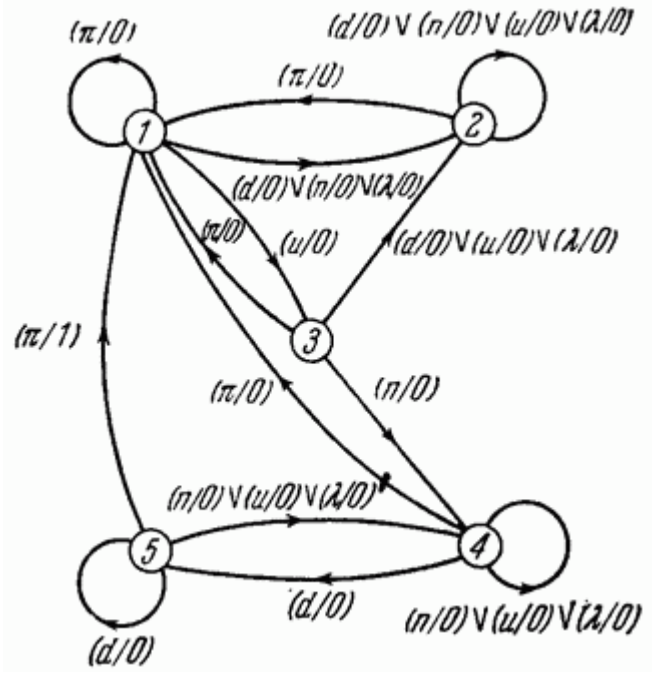


Рисунок 1 – Граф перехода

Такое представление совершенно необходимо для проведения любого точного анализа или синтеза конечного автомата, и мы будем им широко пользоваться в настоящей книге. Поскольку одной формой представления автомата выгодно пользоваться при одних обстоятельствах, другой – при других, полезно познакомиться со всеми формами представления.

Одним из важных применений таблицы переходов является использование ее для перечисления автоматов, принадлежащих тому или иному классу. Класс автоматов часто может быть определен при помощи ряда ограничений, накладываемых на распределение состояний и выходных символов в таблице переходов. Заданный класс автоматов может быть перечислен путем построения всех возможных таблиц

переходов, удовлетворяющих этим ограничениям. Часто мощность класса может быть сразу оценена путем подсчета, определяемого заданными ограничениями числа степеней свободы при построении таблиц переходов [4].

Детерминированный конечный автомат

Детерминированным конечным автоматом (ДКА) называется машина, распознающая цепочки символов, в которой для каждой последовательности входных символов существует лишь одно состояние, в которое автомат может перейти из текущего (рис. 1). Она имеет входную ленту, разбитую на клетки, головку на входной ленте (входную головку) и управляющее устройство с конечным числом состояний. Конечный автомат M можно представить в виде пятерки (S, L, a, s, F), где

1) S – конечное множество состояний устройства управления;
2) L – алфавит входных символов;
3) a – функция переходов, отображающая S x (I U {i}) в множество подмножеств множества S;
4) s – начальное состояние устройства управления;
5) F – множество заключительных (допускающих) состояний [2].

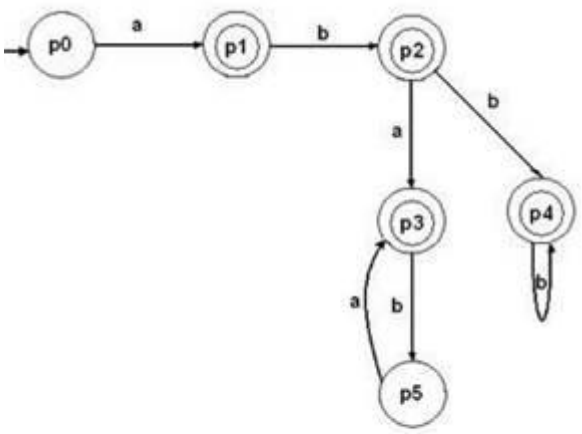


Рисунок 1 – Пример ДКА

ДКА выполняет шаги, определяемые текущим состоянием его блока управления и входным символом, обозреваемым входной головкой. Каждый шаг состоит из перехода в новое состояние и сдвига входной головки на одну клетку вправо. Что язык представим регулярным выражением тогда и только тогда, когда он допускается некоторым конечным автоматом.

Автомат заканчивает свою работу, если достигнуто одно из состояний множества S, или прочитан символ, не принадлежащий L, или входные данные исчерпаны.

Конечные автоматы широко используются на практике, например, в синтаксических, лексических анализаторах, и тестировании программного обеспечения на основе моделей.

Конечный автомат можно описать с помощью диаграмм состояний и таблиц переходов.

Диаграмма состояний (или иногда граф переходов) – графическое представление множества состояний и функции переходов. Представляет собой нагруженный однонаправленный граф, вершины которого – состояния КА, ребра – переходы из одного состояния в другое, а нагрузка – символы, при которых осуществляется данный переход. Если переход из состояния a1 в a2 может быть осуществлен при появлении одного из нескольких символов, то над дугой диаграммы (ветвью графа) должны быть надписаны все они.

Таблица переходов – табличное представление функции F. Обычно в такой таблице каждой строке соответствует одно состояние, а столбцу – один допустимый входной символ. В ячейке на пересечении строки и столбца записывается действие, которое должен выполнить автомат, если в ситуации, когда он находился в данном состоянии на входе он получил данный символ.

Недетерминированный конечный автомат

Недетерминированный конечный автомат – абстрактная машина, которая читает символы из входной цепочки и решает, допустить или отвергнуть эту цепочку. Он может изменить состояние, перейдя из одного состояния в другое. Внутреннюю структуру такого автомата можно представить графом переходов НКА тоже распознаёт цепочки символов, цепочка считается допустимой, если после её обработки множество состояний,

в котором оказался автомат, содержит хотя бы одно допускающее. Таким образом, НКА также задаёт некоторый язык.

Любой недетерминированный конечный автомат может быть преобразован в детерминированный так, чтобы их языки совпадали и такие автоматы называются эквивалентными. Однако, поскольку количество состояний в эквивалентном ДКА в худшем случае растёт экспоненциально с ростом количества состояний исходного НКА, на практике подобная детерминизация не всегда возможна. Кроме того, конечные автоматы с выходом в общем случае не поддаются детерминизации [2].

Для каждого состояния и каждого входного символа НКА имеет ноль или более вариантов выбора следующего шага. Он может также выбирать, сдвигать ему входную головку при изменении состояния или нет. Приведем определение недетерминированного конечного автомата.

С каждым НКА связан ориентированный граф, естественным образом представляющий функцию переходов этого автомата.

Приведем определение для графа (или диаграммы) переходов автомата M = (S, I, a, s, F). Графом переходов автомата M называют ориентированный граф G = (S, E) с помеченными ребрами (см.рис. 2).

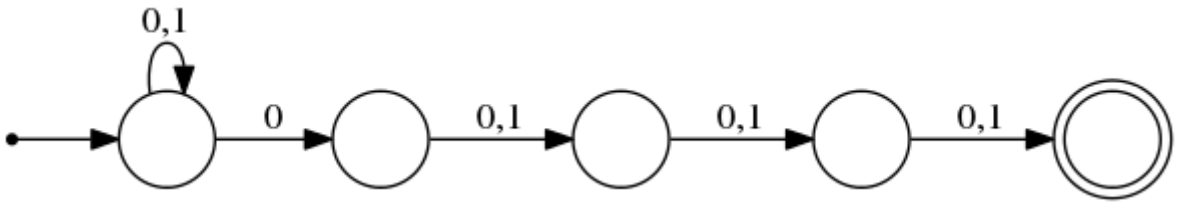


Рисунок 2 – Пример НКА

Существует дополнительный результат или возможность сопоставить какому-либо взятому НКА эквивалентную «детерминированную» машину. Однако детерминированный конечный автомат, эквивалентный данному НКА с n состояниями, может иметь вплоть до 2 в n степени состояний. Поэтому переход от НКА к детерминированному автомату не всегда дает эффективный способ моделирования недетерминированного конечного автомата.

Выводы

Идея применения конечных автоматов является чрезвычайно полезной концепцией, плодотворность которой прошла проверку временем. Использование конечных автоматов позволяет разработчикам создавать хорошо организованные приложения с гибкими возможностями. Их применение позволяет создавать ясный, понятный и надежно функционирующий код.

Были изучены основные методы построения конечных автоматов, исследованы основные формы представления характеристических функций конечного автомата.

Литература

1. Автоматное управление асинхронными процессами в ЭВМ и дискретных системах / Под. ред. В. И. Варшавского. – М. : Наука, 2006. – 400 с.
2. Крючкова, Е.Н. Теория формальных языков и автоматов / Е.Н. Крючкова – М. : Наука, 2008. – 225 с.
3. Донован, Дж. Системное программирование / Дж. Донован. – М. : Мир, 1975. – 540 с.
4. Горбунов, А.Н. Синтез конечного распознающего автомата / Методические указания к выполнению дипломной работы. / А.Н. Горбунов. – Брянск : Мир, 2004. – 20 с.
5. Поспелов, Д.А. Логические методы анализа и синтеза схем / Д.А. Поспелов. – СПб. : Энергия, 1974. – 368 с.
6. Гилл А. Введение в теорию конечных автоматов / А. Гилл – М. : Наука, 1966. – 272 с.

Ковалев Д.В., Копытова О.М. Разработка и исследование метода построения функции деградации поведения конечного автомата в результате переброски дуг. *Рассмотрены основные методы построения конечных автоматов, изучены различные формы представления характеристических функций конечного автомата. Исследованы основные виды конечных автоматов.*

Ключевые слова: *конечный автомат, детерминированный конечный автомат, машина, недетерминированный конечный автомат, граф.*

Kovalev Danil., Kopytova Olga Development and research of the method of constructing the function of degradation of the behavior of a finite automaton as a result of the transfer of arcs. *The basic methods for constructing finite automata are considered, various forms of representing the characteristic functions of a finite automaton are studied. The main types of finite automata are investigated.*

Key words: *finite-state machine, deterministic finite-state machine, machine, non-deterministic finite-state machine, graph.*

УДК 004.89

Исследование автоматизированных обучающих систем в сфере образования

Марченко В.В,. Ольшевский А.И.
Донецкий национальный технический университет
misterm16@yandex.ru, a_olshevskiy@mail.ru

Марченко В.В., Ольшевский А.И. Исследование автоматизированных обучающих систем в сфере образования. *Рассмотрены основные виды автоматизированных систем, а также автоматизированные системы обучения. Выявлены основные этапы модели процесса обучения. Была рассмотрена модель обучаемого, как один из элементов организации процесса обучения.*

Ключевые слова: *автоматизированные системы обучения, модель обучаемого.*

Общая постановка проблемы

В условиях современного информационного общества компьютер имеет широкое применение, в том числе в области автоматизации обучения. Различные средства мультимедии и виртуализации позволили создавать электронные учебники и методические материалы в более интересном формате.В настоящее время компьютер представляет собой информационную систему, а не вычислительную машину, что обусловлено доступом к большому количеству информации благодаря появлению всемирной сети Internet.

Многие проблемы образования стали решаемыми в свете новейших технологий и методик на основании использования информационных образовательных ресурсов вуза. В значительной мере это обусловлено возможностями включения в процесс обучения информационного обмена Интернет-ресурсов.

Однако проблема оценивания знаний обучающихся в полной мере не решена, что обусловлено примитивным подходом к использованию компьютеров – накоплению ограниченных тестовых вопросов и ответов. Что резко сужает деятельность персонального компьютера как обучающей системы. Качество полученных ответов и полноту знаний студента также необходимо учитывать при комплексной оценке запаса знаний студента. Кроме того, при контроле в автоматизированной системе необходимо особое внимание уделять защите информации и особенно оценке качества ответов и результатов проверки от несанкционированного доступа. Все перечисленное создает существенные трудности в использовании персонального компьютера для проверки знаний студента.

Цель исследования

Целью данного исследования является изучение основных видов автоматизированных систем, подробное рассмотрение автоматизированных систем обучения, а также модель обучаемого, как один из элементов организации процесса обучения.

Виды автоматизированных систем обучения

Автоматизированные системы представляют комплекс программных, технических, информационных, лингвистических, организационно-технологических средств и персонала, предназначенный для сбора, первичной обработки, хранения, поиска, вторичной обработки и выдачи данных в заданной форме (виде) для решения разнородных профессиональных задач пользователей системы.

Рассмотрим основные виды автоматизированных систем.

1. Автоматизированная система управления (АСУ).

Такие системы представляют собой комплекс технических и программных средств, который обеспечивает управление объектом в производственной или административной среде.

2. Автоматизированные системы научных исследований (АСНИ).

Таковыми являются программно-аппаратные комплексы, способные обрабатывать данные, поступающие от экспериментальных установок и измерительных приборов.

3. Системы автоматизированного проектирования (САПР).

Они реализуют принципы геометрического моделирования и компьютерной графики, служат для подготовки чертежей, более специализированные системы сосредоточены на технологии изготовления изделий конкретного назначения.

4. Геоинформационные системы (ГИС).

Это автоматизированные системы, имеющие большое количество графических и тематических баз данных, позволяющие преобразовать их в пространственную картографическую информацию. Основная задача этих систем – обеспечить наглядное представление различных «параметров» земной поверхности в форме структурированных карт, которые можно использовать и для научных исследований, и для оптимизации транспортных потоков, размещения сетей деловых объектов, даже оптимизации военных операций [1].

Автоматизированные системы обучения

Среди всех автоматизированных систем выделяют автоматизированную систему обучения, которая улучшает систему образования, позволяя освобождать время преподавателя от проверки заданий и оценке уровня знаний каждого студента. Такие системы включают в себя несколько составляющих:

– работу с лекционным и методическим материалом;

– работу с лабораторными работами;

– работу с научной литературой;

– лекционный курс в электронном варианте;

– систему оценивания знаний, включающую в себя тесты, контрольные и самостоятельные работы, рефераты и др.

Модель процесса обучения в автоматизированной системе состоит из 4 основных этапов. Рассмотрим их подробнее.

1. Этап обучение представляет собой передачу определенного объема обучаемого материала студентам, в виде лекций, методических материалов, а также контрольных и самостоятельных работ и др.

2. Этап контроля знаний подразумевает наличие контрольных вопросов в конце каждого задания и каждой лекции, с целью проверки степени освоенности материала у студента.

3. Этап когнитивного процесса обучения подразумевает способность преподавателя доложить лекционный материал, объяснить сложные моменты в доступной форме, заинтересовать, побудить студентов к положительному восприятию полученной информации, запоминанию и проведению самоконтроля полученных знаний, развитию познавательных способностей студентов.

4. Этап адаптации полученных результатов контроля знаний – это вынесение определенных выводов из полученных результатов контроля, переработка лекционного материала и внесение изменений процесс преподавания, приведение дидактического и лекционного материала к более высокому уровню, совершенному информационному виду и формам предъявлений учебного материала.

Особенностью общей стратегии построения модели обучения является взаимодействие между преподавателем и студентом, а также введение в процесс обучения некоторых программных продуктов в непрерывной форме информационных компьютерных технологий. Для обучения специалиста, имеющего высокий уровень знаний, в первую очередь необходимо определить, каким образом можно повлиять на восприятие студента с целью развития когнитивных способностей. Такое осуществимо только при индивидуальном подходе развития личностных способностей студента.

Метод преподавания также основан на способности преподавателя излагать материал в легкой форме, доступной для восприятия и понимания. В оценивание знаний обязательно включен промежуточный контроль, который представляет собой контрольные вопросы и задания к лабораторным работам, а также тестирование и реферативные работы на определенную тему – это совокупность требований, реализуемых процессом управления. Оценивание знаний представляет собой совокупность результатов объективного оценивания преподавателем уровня знаний, с оценкой знаний автоматизированной системы. Вышеописанная структура модели процесса обучения приведена рисунке 1.

Подсистемой воздействия, в данном случае, является личность конкретного студента, а принятие решений по итогам процесса обучения всего потока студентов – изменения, вносимые в учебный процесс и лекции. Эти изменения являются решающими для получения информации об общем состоянии уровня преподавания и его результатах. Изменения, вносимые в стратегию обучения – это разработка новых методов, методического материала и форм воздействия на объект – «студенты», а также введение индивидуального подхода к объекту воздействия.

Таким образом, при помощи механизма теории управления сложными системами можно формализовать управление не только состоянием уровня подготовки отдельного объекта – студента, но всего потока студентов данного курса. В этом процессе воздействия на уровень знаний студентов, можно выделить как элементы воздействия, определенные рычаги и каналы управления на объект управления. Эта схема воздействия в формализованном виде реализована в теории управления процессами и математическими моделями на основе автоматов. Сам процесс преподавания можно представить в виде схемы взаимодействия этих автоматов [2].

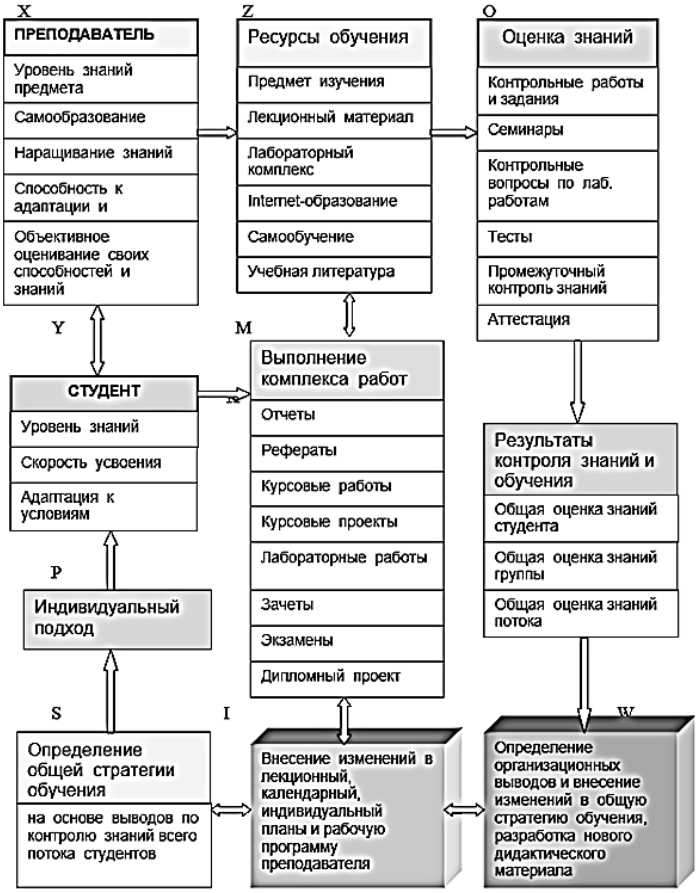


Рисунок 1 – Структурная схема концептуальной модели обучения

Модель обучаемого

Модель обучаемого (МО) – один из важнейших элементов организации процесса обучения. Носителем МО является внешняя по отношению к обучаемому среда. Главная функция МО представлять внешние знания о состоянии (свойствах) обучаемого и позволять моделировать изменения в результате выполнения определенной учебной деятельности. Оценивание учебной деятельности обучаемого как правило базируется на той или иной МО. Спецификация модели обучаемого показывает на основе каких данных, каким способом и в каких терминах формируется внешнее представление о результатах учебной деятельности. Данные представляемые моделью обучаемого используются в следующих основных направлениях:

– управление учебной деятельностью;

– принятия решений о достаточности подготовки учащегося для следующего этапа обучения;

– готовность к выполнению определенной деятельности [3].

В настоящее время существенное развитие преобладает новое направление в данном моделировании обучаемого, которое основывается на разработке имитационных моделей. В имитационных моделях знания обучаемого отображены в виде структур данных, а его умения – в виде процедур и механизма их интерпретации.

Выводы

Были изучены основные виды автоматизированных систем, подробно рассмотрены автоматизированные системы обучения и основные этапы модели процесса обучения, также была рассмотрена модель обучаемого, как один из элементов организации процесса обучения.

Литература

1. Грибова В.В. Обучающие виртуальные системы и средства их создания // Вестник компьютерных и информационных технологий. / В.В. Грибова, Л.А. Федорищев – 2012. – №3. – С. 48–51.

2. Джабраилова З.Г. Нечеткий логический подход к задаче оценки кадрового потенциала // Менеджмент в России и за рубежом. / З.Г. Джабраилова, М.Г. Мамедова – 2004. – №5. – 14 с.

3. Концепция разработки диагностических компьютерных тренажеров на основе знаний / В.В. Грибова [и др.] // AI&Human Resources. – 2009. – №12. – С. 27–33.

4. Слабодчикова А.А. Информационная подготовка в процессе профессионального становления студентов технических специальностей // Актуальные проблемы современной науки. – 2006. – №1. – С. 92–94.

5. Слабодчикова А.А. Формирование информационной готовности студентов как педагогическая проблема // Педагогические науки. – 2005. – №6. – 70 с.

Марченко В.В., Ольшевский А.И. Исследование автоматизированных обучающих систем в сфере образования. *Рассмотрены основные виды автоматизированных систем, а также автоматизированные системы обучения. Выявлены основные этапы модели процесса обучения. Была рассмотрена модель обучаемого, как один из элементов организации процесса обучения.*

Ключевые слова: *автоматизированные системы обучения, модель обучаемого.*

Marchenko V.V., Olshevsky A.I. The research of automated learning systems in education. *The main types of automated systems, as well as automated learning systems are considered. The main stages of the learning process model are revealed. The model of the student was considered as one of the elements of the organization of the learning process.*

Keywords: *automated learning systems, student model.*

УДК 004.9

Анализ проблемы тестирования инженерных кадров

Минлигареев М.А., Ткаченко П.В.
Кузбасский государственный технический университет имени Т.Ф. Горбачева
m-a_a130@mail.ru, lar.tkachenko@mail.ru

Минлигареев М.А., Ткаченко П.В. Анализ проблемы тестирования инженерных кадров. *В статье представлена формализованная проблема тестирования инженерных кадров и, опираясь на обзор способов тестирования решений олимпиадных задач и имеющихся на настоящее время систем тестирования, сформулированы требования к удовлетворяющей всем пользовательским потребностям системе тестирования, на основании которых предложено решение сложившейся проблемы.*

Ключевые слова: *система тестирования, системы, тестирование, олимпиада, задания, решение, компилирование, исходный код, программное обеспечение, программная инженерия.*

Введение

Студенты и школьники, увлекающиеся программированием – это потенциальные инженерные кадры, спрос на которые в настоящее время достаточно велик, но конкуренция, тем не менее, присутствует. Именно поэтому был поднят вопрос об их тестировании. В данном исследовании под тестированием понимается проведение олимпиад по информатике, чтобы выявить наиболее способных инженеров.

Уже сейчас уровень развития не только технического, но и программного обеспечения предоставляет большое количество возможностей для использования ЭВМ в сфере образовании. Достаточно важным фактором образования является проведение тестирования. Очевидно, что его автоматизация значительно облегчит работу всем задействованным в процессе лицам и, соответственно, рост потребности во внедрении автоматизированных информационных систем в процесс тестирования имеет место быть.

Формализация проблемы

На настоящий момент таких систем, которые были бы способны оказать содействие в процессе тестирования, огромное множество. Однако задачей тестирования инженерных кадров является не только тестирование в его типичном понимании, но также автоматическая проверка работоспособности программного кода при заданных ограничениях, а систем тестирования, способных оказать содействие в этом процессе столь низкое количество, что среди них нельзя выделить такую систему, которая бы удовлетворяла всем потребностям её конечных пользователей.

Тестирование программного обеспечения

Цель тестирования программного обеспечения – удостовериться в том, что программа работает именно так, как было задумано, и что она не делает того, что ей не следует делать [1].

В книге [1] отмечено, что тестирование занимает около 50% времени и более 50% стоимости разработки программного обеспечения. Качество тестирования сильно подвержено влиянию человеческого фактора. По этой причине многие исследователи и разработчики прикладывают значительные усилия для автоматизации тестирования.

Также в этой книге рассматривается множество широко применяемых видов тестирования. Они включают в себя такие виды, как:

- чтение кода;
- тестирование «черного ящика»;
- тестирование «белого ящика»;
- модульное тестирование;
- функциональное тестирование;
- отладка.

Обзор способов тестирования решений олимпиадных задач по программированию

На большинстве олимпиад по программированию предлагается решить одну или несколько задач. Формулировка задачи в большинстве случаев предполагает чтение входных данных, удовлетворяющих

условию задачи, получение требуемых результатов на основе этих данных и вывод результатов в формате, указанном в условии задачи [2].

Решением задачи является программа, написанная на одном из алгоритмических языков (например, C, C++, C#, Python и Java).

Запуск программы, ввод и вывод данных осуществляются различными способами, наиболее распространенные из которых приведены ниже:

решение представляет собой консольное приложение, где входные данные находятся в файле с заранее известным именем, а выходные данные также должны быть записаны в определенный файл;

вариация предыдущего подхода: один или оба файла заменяются на поток стандартного ввода (для входного файла) или вывода (для выходного файла). Этот и предыдущий подход характерен для соревнований формата International Collegiate Programming Contest [3], проводимую Association for Computing Machinery;

решение также представляет собой консольное приложение, однако для работы с проверяющей системой оно должно использовать функции некоторой библиотеки. Участникам олимпиады известно только описание этих функций, но не их реализация. С помощью этой библиотеки тестирующая система снабжает тестируемую программу входными данными, а также получает от нее результаты работы. Такой подход применяется для некоторых задач международной олимпиады школьников по информатике [4];

решение запускается с перенаправленными потоками стандартного ввода и вывода. К этим потокам подключена программа, написанная жюри, которая снабжает тестируемую программу данными и получает от нее результаты по определенному, как правило текстовому, протоколу. Задачи такого типа были предложены в качестве эксперимента на полуфинале International Collegiate Programming Contest [3], проводимую Association for Computing Machinery в Северо-Западном регионе в 2008 году;

решение представляет собой класс, реализованный на одном из объектно-ориентированных языков программирования. В нем должен быть определен метод с заданной сигнатурой, который вызывается проверяющей системой. Аргументы этого метода являются входными данными для такого решения, а возвращаемое значение считается результатом работы. Такой подход используется системой соревнований TopCoder [5].

Программа считается прошедшей определенный тест, если она при работе с ним не нарушила ограничений, завершилась корректно (без ошибок времени выполнения), и ее ответ признан правильным.

Обзор способов тестирования решений олимпиадных задач по программированию

На данный момент существует множество систем для тестирования результатов образовательной и олимпиадной деятельности.

Можно выделить следующие критерии, по которым стоит сравнивать такие информационные системы:

- сложность работы с системой для разработчика заданий;
- функциональные возможности для организации тестирования;
- функциональные возможности для обработки и представления результатов.

Из наиболее популярных систем тестирования по этим критериям были оценены следующие:

- «КТС Net»;
- «Indigotech» (см.рис.1);
- «Oprosnik»;
- «Let's test»;
- «OpenTest»;
- «TESTOR.RU»;
- «АСТ-Тест» (см.рис.2);
- «ПоЗнание»;
- «MiniTest-SL» (см.рис.3);
- «TestEdit»;
- образовательный портал «IT Test»;
- подсистема тестирования в системе дистанционного обучения «Moodle»;
- подсистема тестирования в системе дистанционного обучения «Прометей»;
- подсистема тестирования в системе дистанционного обучения «ДОЦЕНТ»;
- подсистема тестирования в системе дистанционного обучения «IBM Lotus LearningSpace»;
- подсистема тестирования в системе дистанционного обучения “Microsoft E-Learning».

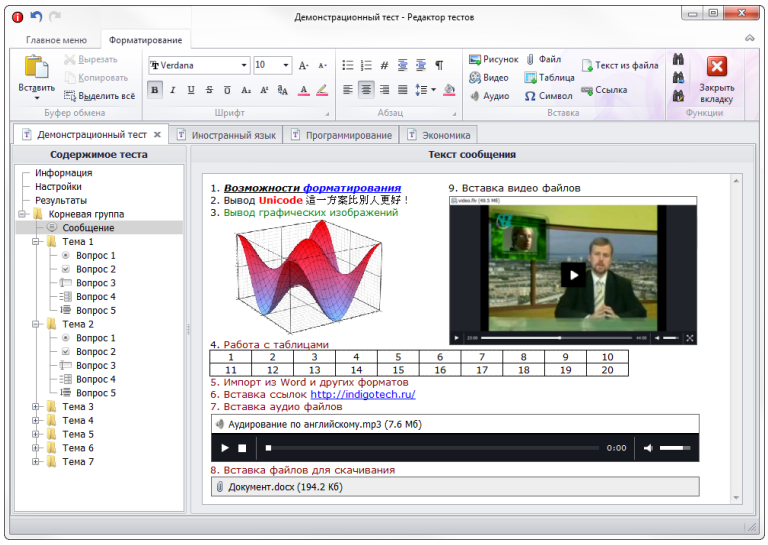


Рисунок 1 - Система тестирования «Indigotech»

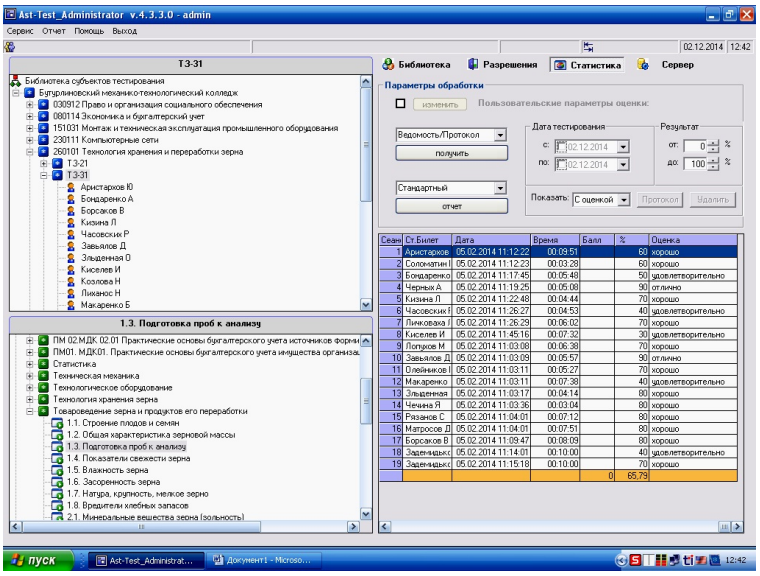


Рисунок 2 - Система тестирования «АСТ-Тест»

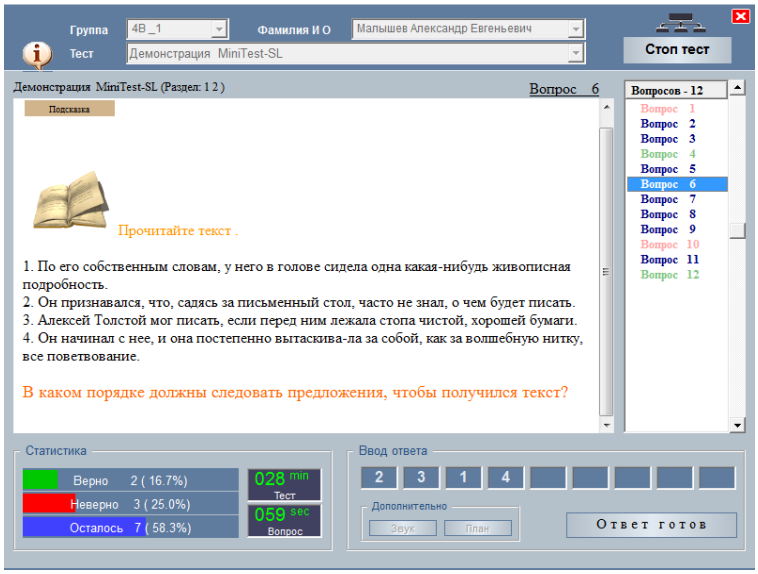


Рисунок 3 - Система тестирования «MiniTest-SL»

После тщательного анализа существующих в настоящее время систем электронного тестирования можно удостовериться в том, что эти системы не удовлетворяют основным требованиям проведения олимпиад по программированию для инженерных кадров.

Требования к системе тестирования

Опираясь на основные потребности пользователей системы тестирования инженерных кадров можно сформулировать следующие требования:

- возможность автоматической генерации логинов и паролей;
- возможность распределения ролей (прав);
- возможность выкладывания олимпиадных заданий с ограничениями на: максимальное время выполнения, максимальный объем используемой памяти, запрет на выполнение определенных операций (например, работа с сетью, графикой, оконной подсистемой);
- возможность выдачи заданий;
- возможность загрузки решенных заданий в систему в виде исходного кода, либо исполняемого файла;
- возможность компилирования загруженных заданий;
- возможность определения ошибок в загруженных заданиях;
- возможность выставления оценки на основании наложенных на задачу ограничений.

Заключение

Решение олимпиадных задач развивает навыки программирования, исследовательской работы, а на некоторых соревнованиях – навыки работы в команде.

Опираясь на вышеупомянутую информацию можно прийти к выводу, что решением проблемы тестирования инженерных кадров может явится разработка такой системы тестирования, которая будет удовлетворять всем изложенным выше требованиям.

Литература

1. Myers G. J. The Art of Software Testing, Second Edition. John Wiley & Sons, Inc., 2004.
2. Буздалов, М. В. Применение генетических алгоритмов для генерации тестов, выявляющих неэффективные решения олимпиадных задач по программированию, на примере задачи о рюкзаке / Буздалов, М. В., Шалыто, А. А. – Санкт-Петербург, 2009. – 65 с.
3. ACM International Collegiate Programming Contest [Электронный ресурс]. Режим доступа: http://en.wikipedia.org/wiki/ACM_ICPC.
4. International Olympiad in Informatics [Электронный ресурс]. Режим доступа: http://www.ioinformatics.org.
5. TopCoder [Электронный ресурс]. Режим доступа: http://www.topcoder.com/tc.

Минлигареев М.А., Ткаченко П.В. Анализ проблемы тестирования инженерных кадров. *В статье представлена формализованная проблема тестирования инженерных кадров и, опираясь на обзор способов тестирования решений олимпиадных задач и имеющихся на настоящее время систем тестирования, сформулированы требования к удовлетворяющей всем пользовательским потребностям системе тестирования, на основании которых предложено решение сложившейся проблемы.*

Ключевые слова: *система тестирования, системы, тестирование, олимпиада, задания, решение, компилирование, исходный код, программное обеспечение, программная инженерия.*

Minligareev Maxim, Tkachenko Pavel. Analysis of the problem of testing engineering personnel. *The article presents a formalized problem of testing engineering personnel and, based on a review of ways to test solutions to Olympiad problems and currently available testing systems, formulated requirements for a testing system that satisfies all user needs, based on which a solution to the current problem was proposed.*

Key words: *testing system, systems, testing, Olympiad, tasks, solution, compiling, source code, software, software engineering.*

УДК 004.424

Методика экспериментального определения показателей вычислительной сложности параллельных алгоритмов

Нескородев Р.Н., Белик Т.С., Галиахметова К.Р.
Донецкий национальный университет
nromn@i.ua, g2013.belik.t.13033@gmail.com, kamilla.galiahmetova@mail.ru

Нескородев Р.Н., Белик Т.С., Галиахметова К.Р. Методика экспериментального определения показателей вычислительной сложности параллельных алгоритмов. *В работе рассмотрены современные проблемы технологий программирования, связанные с методами разработки параллельных вычислений. На примере численного вычисления двойного интеграла при помощи технологии OpenMP проведена сравнительная оценка теоретических и экспериментальных показателей вычислительной сложности последовательного и параллельного алгоритмов.*

Ключевые слова: *технологии параллельных вычислений, последовательный и параллельный алгоритм, вычислительная сложность алгоритма, технология OpenMP.*

Введение

Почему сегодня важно параллельное программирование или параллельные вычисления? На сегодняшний день для решения довольно сложных задач требуются все более мощные вычислительные средства. В качестве примеров сложных задач - моделирование климата, определение аэродинамических характеристик летательного аппарата, космическая и военная отрасли, генная инженерия, сейсморазведка, нефте- и газодобывающая промышленность, автомобилестроение, проектировании электронных и электронно-вычислительных устройств, фармакология, синтез новых материалов, автоматизированные системы управления и др. Многие из этих задач требуют для своего анализа супер-ЭВМ.

В настоящее время в мире произведены, работают и продолжают выпускаться миллионы вычислительных машин, относящихся к различным поколениям, типам, классам, отличающихся своими областями применения, техническими характеристиками и вычислительными возможностями.

Первый двухъядерный процессор был анонсирован в 1999 году фирмой IBM (семейство Power4), продажа которых началась в 2001 г. .Современные персональные компьютеры и 64-битные операционные системы способны поддерживать до 64 ядер для рабочих станций и до 256 ядер для серверов.

Для использования всех ядер процессора необходимо параллельное программирование, которое не только обеспечит параллельное исполнение участков программы, но и решит проблемы синхронизации, взаимных исключений, равномерной загрузки отдельных ядер и др. Это ставит новые задачи перед разработчиками системного и прикладного программного обеспечения.

Цель данной работы – изучение принципов написания высокопроизводительных реализаций алгоритмов с использованием технологии OpenMP и проведение сравнительной оценки теоретических и экспериментальных показателей вычислительной сложности последовательного и параллельного алгоритмов. В качестве примера нами был выбран алгоритм численного вычисления двойного интеграла по методу параллелепипедов [1].

Показатели для оценки параллельных алгоритмов

Ускорение $S_p(n)$ для параллельного алгоритма размерностью n определяется отношением временной сложности последовательного алгоритма $T_1(n)$ и параллельного алгоритма для p процессоров $T_p(n)$ [2]

$$S_p(n) = T_1(n) / T_p(n). \quad (1)$$

Если достигнуто равномерное использование всех процессоров и нет накладных расходов, связанных с распараллеливанием, то $T_p(n) = T_1(n) / p$ и максимальное ускорение $S_p(n) = p$ **.**

Эффективность $E_p(n)$ для параллельного алгоритма размерностью n определяется ускорением этого

алгоритма по отношению к одному процессору, т.е.

$$E_p(n) = S_p(n) / p = T_1(n) / \left(p * T_p(n)\right).$$

(2)

Предельное значение эффективности равно 1 в случае достижения максимального ускорения. Показатели ускорения и эффективности взаимно противоположны. Для достижения максимального ускорения стремятся увеличить число процессоров. С другой стороны, увеличение числа процессоров уменьшает эффективность. В качестве комплексного показателя используется показатель стоимости.

Стоимость вычислений $C_p(n)$. Чем лучше алгоритм распараллелен, тем меньше его временная сложность $T_p(n)$. Чем больше процессоров p используется, тем дороже вычислительная система. Стоимость вычислений оценивается произведением этих показателей, т.е.

$$C_p(n) = p * T_p(n). \quad (3)$$

Для определения этих показателей можно использовать закон Амдала или закон Густафсона-Барсиса [3].

Закон Амдала определяет теоретическое значение ускорения без учета накладных расходов, связанных с параллельными вычислениями.

Обозначим время выполнения программы в последовательном режиме 1. Пусть β - часть программы, которая должна выполняться последовательно, тогда - $1-\beta$ - часть программы, которая может исполняться параллельно. Пусть количество процессоров равно p, все процессоры равномерно загружены при выполнении параллельной части программы, и накладными расходами можно пренебречь. Тогда ускорение определяется формулой

$$S_p = \frac{1}{\beta + \frac{1-\beta}{p}} = \frac{p}{\beta * p + 1 - \beta}.$$

(4)

Это есть первая формулировка закона Амдала.

Очевидно, что если число процессоров $p \to \infty$, то величина ускорения будет максимальной и равной:

$$\lim_{p \to \infty} S_p = \frac{1}{\beta}.$$

(5)

Формула (5) - вторая формулировка закона Амдала, согласно которому ускорение не может превосходить обратной величины доли последовательных вычислений.

Анализ результатов, полученных в соответствии с законом Амдала, показывает, что ускорение в пределе зависит только от части программы, которая должна выполняться последовательно, и практически не зависит от количества процессоров, что часто не подтверждается на практике.

Значительно ближе к практике закон Густафсона, согласно которому определяется объем работ (число команд), которые можно выполнить в случае последовательной и параллельной обработки. Значение отношения этих объемов работ определяет масштабируемое (с учетом числа процессоров) ускорение. Пусть по-прежнему, β - часть кода, которую нужно выполнить последовательно. Тогда параллельная часть составит $1-\beta$. Если в последовательном режиме выполняется V_s команд, то в параллельном режиме за это же время будет выполнено $V_p = \beta * V_s + (1-\beta) * V_s * p$. Ускорение в этом случае равно:

$$S_p = \frac{V_p}{V_s} = \beta + (1-\beta) * p = p - \beta * (p-1). \quad (6)$$

Формула (6) определяет оценку Густафсона-Барсиса.

Формулы (4) и (6) при одинаковых β в общем случае дают разные значения. Это связано с тем, что доля последовательно выполняемых операций и доля времени на их выполнение представляют совершенно разные понятия и просто так их нельзя полагать равными. Доля последовательных вычислений по определению не зависит от числа p используемых процессоров, а доля времени на их выполнение - зависит. Между β_A и β_Γ существует связь [3]

$$\beta_\Gamma = \frac{p\beta_A}{1+(p-1)\beta_A}.$$

Подставляя это выражение для β_Γ в (6), получаем, что

$$S_p^\Gamma = p-\beta_\Gamma(p-1) = p-\frac{p(p-1)\beta_A}{1+(p-1)\beta_A} = \frac{p}{p\beta_A+(1-\beta_A)} = S_p^A.$$

Таким образом, если учесть связь между β_A и β_Γ, то оценки ускорения по Амдалу и по Густафсону-Барсису полностью совпадают. Формулу Амдала следует применять для прогноза возможного ускорения. В данном случае величину β_A можно подсчитать, не пропуская программу на параллельной вычислительной системе. Формулу Густафсона-Барсиса можно применять для оценивания достигнутого ускорения, не пропуская программу на однопроцессорной машине. Величина β_Γ при этом измеряется в процессе решения задачи.

Экспериментальные методы определения вычислительной сложности алгоритмов

При экспериментальном определении реализуется программа и измеряется время выполнения участка программы, который надо проанализировать, или целиком программы. Для этого в код вставляются функции, которые возвращают текущее время, в начало и конец заданного участка программы, а затем вычисляется разность времен. Для увеличения точности экспериментального определения оно повторяется многократно. Для минимизации влияния других программ все программы, от которых не зависит анализируемая программа, предварительно завершаются.

Для экспериментального определения можно использовать:

- стандартные функции языка *C++*;
- функции *WinApi*;
- использование счетчика тактов центрального процессора - *Time-Stamp Counter (TSC)*;
- если измеряется время выполнения функции и среда разработки имеет средства профилирования, то используются эти средства для определения числа вызовов функции и времени ее выполнения.

Для анализа производительности программы будем использовать встроенные средства анализа производительности программной среды Visual Studio в следующей тестовой инфраструктуре:

Таблица 1 – Тестовая инфраструктура

Процессор	Intel Core i3-3120M CPU 2,5 GHz – 2 ядра Intel Core i3-3120M CPU 2,5 GHz – 2 ядра
Память	6 Гб
Операционная система	Microsoft Windows 8.1
Среда разработки	Visual Studio 2013

Пример разработки программы

Пусть необходимо вычислить значение двумерного интеграла $I = \iint\limits_D f(x,y)\,dxdy$, где D - прямоугольник $D = [a_1,b_1]\times[a_2,b_2]$. Приближенное значение интеграла I может быть вычислено по формуле параллелепипедов (аналог формулы прямоугольников для одномерного случая)

$$\begin{cases} I = \int\limits_{a_1}^{b_1}\int\limits_{a_2}^{b_2} f(x,y)\,dxdy \approx h_1h_2\sum\limits_{i=0}^{N-1}\sum\limits_{j=0}^{M-1} f(x_i,y_j) \\ x_i = a_1 + ih_1 + \frac{h_1}{2},\ y_j = a_2 + jh_2 + \frac{h_2}{2}, \end{cases}$$

где N – количество участков интегрирования по оси x, M – количество участков интегрирования по оси y, $h_1 = (b_1-a_1)/N$ и $h_2 = (b_2-a_2)/M$. В качестве тестового примера возьмем функцию

$$f(x,y) = \frac{\exp(\cos(\pi x)\sin(\pi y))+1}{(b_2-a_2)(b_1-a_1)}, \text{ где } x\in D \text{ и } a_1 = a_2 = 0,\ b_1 = b_2 = 16.$$

В последовательном режиме программа имеет вид

```
n1=(int)((b1-a1)/h);
n2=(int)((b2-a2)/h);
sum=0.0;
for(i=0;i<n1;i++)
{x=a1+i*h+h/2;
  for(j=0;j<n2;j++)
   {y=a2+j*h+h/2;
     sum+=f(x,y)*h*h;
   }
}
```

Определим ожидаемые показатели времени выполнения программы с учетом параллельного выполнения. Определим максимальное ускорение.

Предположим, что время выполнения одной итерации равно t. В этом случае в последовательном режиме потребуется $T_1(n) = t*n*n$. При параллельном выполнении максимальный размер порции для одного потока равен $n/p+1$. Тогда $T_p(n) = t*(n*n/p+1)$. В этом случае показатели:

$$S_p(n) = \frac{T_1(n)}{T_p(n)} = \frac{t*n*n}{t*(n*n/p+1)} = \frac{n*n*p}{n*n+p}, \qquad \lim_{n\to\infty} S_p(n) = p;$$

$$E_p(n) = \frac{S_p(n)}{p} = \frac{n*n}{n*n+p}, \qquad \lim_{n\to\infty} E_p(n) = 1;$$

$$C_p(n) = T_p(n)*p = t*p*n*(n*n/p+1) = t*n*n+t*p.$$

Предельное значение стоимости совпадает со значением времени в последовательном режиме. Ожидаемые параметры близки к идеальным.

Для реализации в параллельном режиме используем технологию разработки OpenMP, которая является одним из наиболее популярных средств программирования для компьютеров с общей памятью, базирующихся на традиционных языках программирования. За основу берётся последовательная программа, а для создания её параллельной версии пользователю предоставляется набор директив, функций и переменных окружения. Предполагается, что создаваемая параллельная программа будет переносимой между различными компьютерами с разделяемой памятью, поддерживающими OpenMP API.

Следующим шагом является распараллеливание базовой реализации алгоритма. Один из наиболее простых подходов – геометрическая декомпозиция данных. Данный подход предполагает разделение данных на части и применение к ним одного и того же алгоритма. В численном интегрировании имеем дело с прямоугольной сеткой, в каждом узле которой вычисляется функция и умножается на квадрат шага. Вычисление функции в одном узле сетки не зависит от соседних узлов, таким образом, поделив сетку между потоками, можно получить параллельную версию, причем ожидаемое ускорение должно быть близко к линейному. Реализуем параллельную версию вычисления интеграла с разделением сетки интегрирования по столбцам.

Согласно схеме разбиения данных необходимо распараллеливать внешний цикл представленной программы. Это можно сделать с помощью добавления директивы *omp parallel for*. Но полученная, таким образом, параллельная версия будет содержать гонки данных. Это связано с четырьмя переменными:

- x и y – координаты точки, где необходимо вычислить функцию;
- j – переменная внутреннего цикла;
- *sum* – переменная для накопления суммы интеграла.

Для того, чтобы не возникали ошибки, первые три переменные необходимо сделать локальными для потоков, а по переменной *sum* организовать редукцию данных с операцией суммирования. Для локализации данных в директиве *parallel* можно использовать параметр *private (<список переменных>)*. Для редукции данных в директиве *omp for* используется параметр *reduction (<операция>:<список переменных>)*.

В результате код численного нахождения интеграла будет иметь вид:

```
n1=(int)((b1-a1)/h);
n2=(int)((b2-a2)/h);
sum=0.0;
#pragma omp parallel for private (x, y, j) reduction (+: sum)
for(i=0;i<n1;i++)
{x=a1+i*h+h/2;
  for(j=0;j<n2;j++)
   {y=a2+j*h+h/2;
```

```
        sum+=f(x,y)*h*h;
    }}
```

Результаты округления десятикратного выполнения функции представлены в таблице 2.

Таблица 2 Экспериментальные результаты

	$h=0.1,\ n=160$	$h=0.01,\ n=1600$	$h=0.001,\ n=16000$	Расчетное значение
Последовательная версия, время в *c*	0.003053	0.301972	30.088558	
Параллельная версия, время в *c*	0.002024	0.120572	11.015288	$p=2$
Ускорение, S_2	1.508399	2.504495	2.731527	2
Эффективность, E_2	0.754199	1.252248	1.365763	1
Параллельная версия, время в *c*	0.001212	0.113482	10.906492	$p=3$
Ускорение, S_3	2.518977	2.660968	2.758775	3
Эффективность, E_3	0.839659	0.886989	0.919592	1
Параллельная версия, время в *c*	0.00187	0.175753	8.432265	$p=4$
Ускорение, S_4	1.632620	1.718161	3.568265	4
Эффективность, E_4	0.408155	0.429540	0.892066	1

Анализ данных таблицы 2 позволяет сделать следующие выводы: при использовании нескольких ядер с увеличением числа точек интегрирования заметно явное улучшение ускорения и эффективности параллельной версии. Однако, с увеличением числа ядер уменьшается эффективность параллельного алгоритма, это может быть связано с накладными расходами при передаче данных во время выполнении программы. Можно, также увидеть превышение теоретических значений ускорения и эффективности при $p=2$ и больших значениях n, которое можно объяснить наличием внутреннего кеша у каждого ядра, что уменьшает вероятность промаха при выборе данных по сравнению с использованием одного ядра.

Литература

1. Бахвалов Н.С. Численные методы / Н.С. Бахвалов, Н.П. Жидков, Г.М. Кобельков. – М: Бином. Лаборатория знаний, 2008. – 640 с.
2. Качко Е.Г. Параллельное программирование: Учебное пособие / Е.Г. Качко. – Харьков: Изд-во “Форт”, 2011. – 528 с.
3. Воеводин В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин. – СПб: БХВ – Петербург, 2002. – 608 с.

Нескородев Р.Н., Белик Т.С., Галиахметова К.Р. Методика экспериментального определения показателей вычислительной сложности параллельных алгоритмов. *В работе рассмотрены современные проблемы технологий программирования, связанные с методами разработки параллельных вычислений. На примере численного вычисления двойного интеграла при помощи технологии OpenMP проведена сравнительная оценка теоретических и экспериментальных показателей вычислительной сложности последовательного и параллельного алгоритмов.*

Ключевые слова: *технологии параллельных вычислений, последовательный и параллельный алгоритм, вычислительная сложность алгоритма, технология OpenMP.*

Neskorodev Roman, Bielik Tetiana, Galiahmetova Kamilla Methods of experimental determination of indicators of the computational complexity of parallel algorithms. *The paper discusses the current problems of programming technologies associated with the methods of developing parallel computing. Using the example of the numerical computation of the double integral using the OpenMP technology, a comparative evaluation of the theoretical and experimental indicators of the computational complexity of the sequential and parallel algorithms is carried out.*

Key words: *parallel computing technologies, sequential and parallel algorithm, computational complexity of the algorithm, OpenMP technology.*

УДК 004.855.5

ИССЛЕДОВАНИЕ РАЗРАБОТКИ ОНТОЛОГИЧЕСКОГО ЧАТ-БОТА НА БАЗЕ ГЛУБИННОГО ОБУЧЕНИЯ НЕЙРОННЫХ СЕТЕЙ

Оверченко Я.Ю., Копытова О.М.
Донецкий национальный технический университет
yarikoverchenko@gmail.com, omkop@list.ru

Оверченко Я.Ю., Копытова О.М. Исследование разработки онтологического чат-бота на базе глубинного обучения нейронных сетей. *Рассмотрены основные подходы к построению систем искусственного интеллекта, изучены ключевые архитектуры нейронных сетей. Определён наиболее подходящий тип нейронной сети для практической реализации онтологического чат-бота. Исследованы основные методы обучения нейросетевых систем.*

Ключевые слова: *нейронные сети, глубинное обучение, чат-бот, искусственный интеллект.*

Общая постановка проблемы

В настоящее время интеграция компьютерных технологий в человеческую жизнь настолько обширна, что довольно сложно назвать сферы человеческой деятельности, в которых не используются различные автоматизированные компьютерные системы. Они могут служить для управления всевозможными технологическими процессами и оборудованием, для развлечения людей, а также для облегчения их повседневной жизни и помощи человеку в обучении, работе или быту.

Одним из видов таких систем являются чат-боты. Они позволяют автоматизировать различные работы, связанные с использованием сети Интернет, такие как, например, регистрация заявок пользователей или обеспечение предварительной технической поддержки, проведение опросов пользователей определенной сети. Помимо того, чат-боты могут выступать помощниками в обучении, автоматизируя процесс подачи различной информации.

Существуют несколько основных подходов к реализации подобных систем. Первый заключается в создании набора предустановленных шаблонов, в рамках которого чат-бот генерирует ответы с практически отсутствующими отклонениями. Второй подход предполагает использование нейронных сетей, что позволяет создать систему, которая способна генерировать импровизированные ответы на основе контекста вопроса и самостоятельно совершенствовать свой алгоритм генерации ответа с каждым новым диалогом.

Цель исследования

Целью данного исследования является изучение основных видов и архитектур нейронных сетей, методик их обучения, а также выбор наиболее подходящей архитектуры нейронной сети для реализации на её основе онтологического чат-бота, который сможет естественным образом имитировать человека-собеседника.

Архитектуры нейронных сетей

Основными структурными элементами нейронной сети являются нейрон и синапс (см.рис.1).

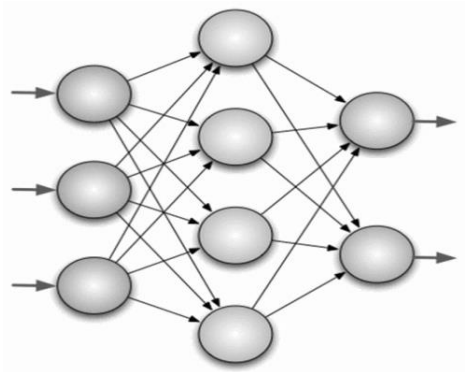


Рисунок 1 – Схема нейронной сети

Нейрон – это вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше. Они делятся на три основных типа: входной, скрытый и выходной. В том случае, когда нейросеть состоит из большого количества нейронов, вводят термин слоя. Соответственно, есть входной слой, который получает информацию, n скрытых слоев (обычно их не больше 3), которые ее обрабатывают и выходной слой, который выводит результат.

Синапс – это связь между двумя нейронами. У синапсов есть параметр – вес. Благодаря ему, входная информация изменяется, когда передается от одного нейрона к другому. Допустим, есть 3 нейрона, которые передают информацию следующему. Тогда у нас есть 3 веса, соответствующие каждому из этих нейронов. У того нейрона, у которого вес будет больше, та информация и будет доминирующей в следующем нейроне. На самом деле, совокупность весов нейронной сети или матрица весов – это своеобразный мозг всей системы. Именно благодаря этим весам, входная информация обрабатывается и превращается в результат. Важно помнить, что во время инициализации нейронной сети, веса расставляются в случайном порядке.

Согласно одной из классификаций нейронных сетей, которая была определена исходя из механизмов их функционирования и характера связей, существует четыре основных вида:

– сети прямого распространения (все связи направлены строго от входных нейронов к выходным);

– рекуррентные сети (сигнал с выходных нейронов или нейронов скрытого слоя частично передается обратно на входы нейронов входного слоя, образуя обратную связь);

– радиально-базисные функции (искусственные нейронные сети, использующие в качестве активационных функций радиально-базисные, также называются RBF-сетями);

– самоорганизующиеся карты (представляют собой соревновательную нейронную сеть с обучением без учителя, выполняющую задачу визуализации и кластеризации) [1][2].

В ходе исследования вышеуказанных архитектур нейронных сетей было определено, что наиболее подходящими для реализации онтологического чат-бота являются рекуррентные сети [2]. Они «фильтруют» входные данные, возвращаясь к устойчивому состоянию и, таким образом, позволяют решать задачи компрессии данных и построения ассоциативной памяти. Благодаря этому появляется возможность обрабатывать серии событий во времени или последовательные пространственные цепочки. В отличие от многослойных перцептронов, рекуррентные сети могут использовать свою внутреннюю память для обработки последовательностей произвольной длины. Поэтому они применимы в таких задачах, где нечто целостное разбито на сегменты, например: распознавание рукописного текста или распознавание речи [1]. Частным случаем рекуррентных сетей являются двунаправленные сети. В таких сетях между слоями существуют связи как в направлении от входного слоя к выходному, так и в обратном.

Трудность рекуррентной сети заключается в том, что если учитывать каждый шаг времени, то становится необходимым для каждого шага времени создавать свой слой нейронов, что вызывает серьёзные вычислительные сложности. Кроме того, многослойные реализации оказываются вычислительно неустойчивыми, так как в них, как правило, исчезают или зашкаливают веса. Если ограничить расчёт фиксированным временным окном, то полученные модели не будут отражать долгосрочных трендов. Различные подходы пытаются усовершенствовать модель исторической памяти и механизм запоминания и забывания.

Одна из главных областей применения рекуррентных нейронных сетей на сегодняшний день – работа с языковыми моделями, в частности – анализ контекста и общей связи слов в тексте. Для сети такого типа структура языка – это долгосрочная информация, которую надо запомнить. К ней относятся грамматика, а также стилистические особенности того корпуса текстов, на которых производится обучение. Фактически рекуррентная нейронная сеть запоминает, в каком порядке обычно следуют слова, и может дописать предложение, получив некоторую затравку. Если эта затравка случайная, может получиться совершенно бессмысленный текст, стилистически напоминающий шаблон, на котором училась нейронная сеть. Если же исходный текст был осмысленным, сеть поможет его стилизовать, однако в последнем случае одной рекуррентной сети будет мало, так как результат должен представлять собой «смесь» случайного, но стилизованного текста и осмысленной, но «неокрашенной» исходной части.

Исходя из этого рекуррентная нейронная сеть является наилучшей архитектурой для построения онтологического чат-бота, так как позволит реализовать возможность генерировать ответные реплики, используя контекст диалога и наибольшим образом их разнообразить, исключая шаблонность ответов.

Методы обучения нейронных сетей

Нейронные сети требуют обучения, в противном случае правильный результат вряд ли будет получен. Существует много различных методов обучения нейронных сетей. Среди них выделяют четыре наиболее эффективных способа:

– метод обратного распространения;

– метод упругого распространения;

– генетический анализ.

– глубинное обучение [3].

Метод обратного распространения использует алгоритм градиентного спуска. Данный алгоритм представляет собой способ нахождения локального минимума или максимума функции с помощью движения вдоль градиента [3]. Градиент представляет собой векторное значение, определяющее направление и крутизну склона. Градиент находится с помощью производной от функции в нужной точке. Оказавшись в определённой точке со значением веса, который распределяется в случайном порядке, вычисляется градиент и определяется направление движения спуска, и так в каждой следующей точке, пока не достигается локальный минимум, не позволяющий дальнейшего спуска. Чтобы справиться с этой неприятностью, необходимо установить нужное значение момента, которое позволит преодолеть часть графика и достигнуть нужной точки. Если это значение будет недостаточным, то преодолеть выпуклость не получится, в случае с установкой более высокого значения, чем нужно есть шансы проскочить глобальный минимум. Кроме момента ускорения, есть ещё понятие, определяющее общую скорость обучения сети. Это значение, как и предыдущее, представляет собой гиперпараметр и подбирается методом проб и ошибок. Оптимальный вариант заранее никогда не известен, узнать его можно только проведя несколько обучений и корректируя каждый раз значение в нужном направлении. В процессе поступления информации нейронная сеть последовательно передаёт её от одного нейрона к другому посредством синапсов, до того момента, пока информация не окажется на выходном слое и не будет выдана как результат. Такой способ называется «передачей вперёд» [3]. После того, как результат получен, вычисляется ошибка и на её сновании выполняется обратная передача, суть которой – последовательно изменить вес синапсов, начиная с выходного и продвигаясь к входному слою. При этом значение веса меняется в сторону лучшего результата. Для использования такого метода обучения подойдут только те функции активации, которые можно дифференцировать.

Описанный выше метод обратного распространения имеет недостаток в виде больших временных затрат на процесс обучения неуместных в случае необходимости получить быстрый результат. Для ускорения процесса было предложено немало дополнительных алгоритмов, ускоряющих процесс. Одним из которых и является текущий метод. Метод упругого распространения использует в качестве основы обучение по эпохам и применяет только знаки производных частного случая для корректировки весовых коэффициентов. Используется определённое правило, по которому производится расчёт величины коррекции весового коэффициента. Если на этом этапе расчётов производная меняет свой знак, значит, то нужно произвести откат, то есть вес вернуть в обратную позицию, а величину изменения уменьшить. Если знак производной не изменился, то величина изменения веса, наоборот, увеличивается для большей сходимости. Если основные параметры коррекции веса зафиксировать, то настройки глобальных параметров можно избежать[3]. И это является преимуществом текущего метода над предыдущим. Для этих параметров есть рекомендуемые значения, однако, никаких ограничений на их выбор не накладывается. Такой подход позволяет добиться сходимости нейросети быстрее в несколько раз в отличие от предыдущего варианта обучения.

Генетический анализ представляет собой упрощённую интерпретацию природного алгоритма, основанного на скрещивании результатов. То есть, по сути, происходит скрещивание результатов, выбор наилучших и формирование на их основе нового поколениях [3]. В случае, если результат не устраивает, алгоритм повторяется, пока поколение не становится идеальным. Алгоритм может завершиться без достижения нужного результата, если количество попыток будет исчерпано или же будет исчерпано время на мутацию. Этот алгоритм применим к процессу оптимизации веса нейронной сети, при заданной по умолчанию топологии [3]. При этом вес кодируется двоичным кодом, и каждый результат определяется полным набором веса. Оценка качества происходит методом вычисления ошибки на выходе [3].

Глубинное обучение – статистическая техника классификации закономерностей на основе пробных данных с использованием многослойных нейросетей. Нейронные сети, обучаемые глубинным методом, обычно состоят из набора модулей ввода, принимающих такие данные, как пиксели или слова, множества скрытых слоёв, содержащих скрытые модули, и набора модулей вывода, с учётом наличия связей между различными узлами [4]. Системы глубинного обучения чаще всего используются как классификационные, в том смысле, что миссия типичной сети – это решения по поводу того, к какому набору категорий принадлежит данный ввод [5]. Исходя из этого, данный метод является наиболее подходящим для обучения рекуррентной нейронной сети, выбранной для практической реализации онтологического чат-бота.

Кроме рассмотренных выше методов, есть ещё разновидности обучения нейросетей с учителем и без него, а также обучение с подкреплением.

Обучение без учителя носит несколько иной характер и встречается реже. При таком раскладе нейронная сеть не получает желаемого результата. Такая тренировка подходит сетям, задача которых кластеризация данных по заданным параметрам. То есть, проанализировав большой объем входных данных, сеть разделяет их на категории по определённым признакам.

Обучение с подкреплением применяется тогда, когда есть возможность оценить итоговый результат, выданный сетью. То есть путём определённого поощрения нейронной сети каждый раз, когда полученный результат максимально приближен к желаемому мы дадим ей возможность искать любые пути решения проблемы, пока она будет давать нужные результаты. Благодаря этому сеть будет искать наилучшие способы достижения цели без данных от тренера.

Обучение с учителем чаще всего применимо к регрессиям и классификациям. В этом случае тренер выступает в роли учителя, а созданная сеть – ученика. Тренер задаёт входные данные и требующийся результат, соответственно сеть понимает к какому именно результату необходимо стремиться при заданных параметрах (см.рис.2).

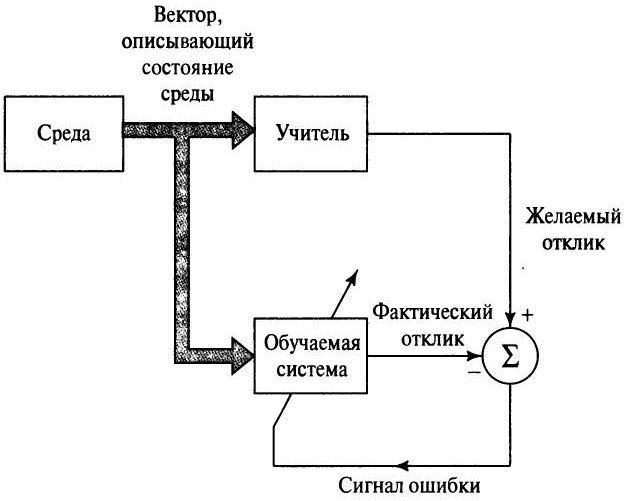


Рисунок 2 – Схема метода обучения нейронной сети с учителем

Выводы

Были изучены основные существующие архитектуры реализации нейронных сетей. В ходе их анализа был выбрана наиболее подходящая архитектура для реализации онтологического чат-бота. Также были рассмотрены ключевые методы обучения нейронных сетей, среди которых был выбран наиболее эффективный метод обучения нейронной сети, определенной ранее архитектуры.

Литература

1. Савельев, А. В. На пути к общей теории нейросетей. К вопросу о сложности // Нейрокомпьютеры: разработка, применение / А.В. Савельев. – 2006. – № 4–5. – С. 4–14.

2. Ясницкий, Л. Н. Введение в искусственный интеллект / Л. Н. Ясницкий. – М.: Издат. центр «Академия», 2005. – 176 с.

3. Вороновский, Г. К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г.К. Вороновский, К. В. Махотило, С. Н. Петрашев, С. А. Сергеев. – Харьков: Основа, 1997. – 112 с.

4. Николенко, С. Глубокое обучение / С. Николенко, А. Кадурин, Е. Архангельская. – СПб.: Питер, 2018. – 480 с.

5. Гудфеллоу, Я. Глубокое обучение / Я. Гудфеллоу, И. Бенджио, А. Курвилль. – М.: ДМК-Пресс, 2017. – 652 с.

Оверченко Я.Ю., Копытова О.М. Исследование разработки онтологического чат-бота на базе глубинного обучения нейронных сетей. *Рассмотрены основные подходы к построению систем искусственного интеллекта, изучены ключевые архитектуры нейронных сетей. Определён наиболее подходящий тип нейронной сети для практической реализации онтологического чат-бота. Исследованы основные методы обучения нейросетевых систем.*

Ключевые слова: *нейронные сети, глубинное обучение, чат-бот, искусственный интеллект.*

Overchenko Y.Y., Kopytova O.M. Research of the ontological chatbot based on the deep learning of neural networks development*. The main approaches to the construction of artificial intelligence systems are considered, the key architectures of neural networks are studied. The most suitable type of neural network for the practical implementation of an ontological chat bot has been determined. The basic methods of teaching neural network systems are investigated.*

Keywords: *neural networks, deep learning, chat bot, artificial intelligence.*

УДК 004.912

«Создание интеллектуальной системы стилистической оценки текста»

Столбунская А.С., Кравец Т.Н.
Донецкий национальный технический университет
nastassjanastya@gmail.com

Столбунская А.С., Кравец Т.Н. Создание интеллектуальной системы стилистической оценки текста. *Исследование обработки естественного языка – одно из направлений искусственного интеллекта и математической лингвистики, которое занимается изучением проблем компьютерного анализа и синтеза естественных языков. Сложностью оценки как процесса и результата познавательной деятельности является проблема статуса категории оценки на уровне слова, высказывания и текста. Разработка алгоритма анализа оценки решает ряд вопросов в области исследования текста.*

Ключевые слова: *статья, интеллектуальные системы, обработка текста, обработка естественного языка, чат-боты, машинное обучение.*

Введение

В настоящее время основными проблемами лингвистики является изучение лексики и семантики, быстрый автоматизированный перевод. В этих исследованиях невозможно обойтись без работы со словарями и архивами. Но у учёных не всегда существует возможность доступа к необходимым информационным ресурсам. Помочь в этом современным лингвистам может такая отрасль науки, как компьютерная прикладная лингвистика, которая занимается созданием разнообразных систем по обработке естественного языка.

Целью данной работы является исследование обработки естественного языка – одного из направлений искусственного интеллекта и математической лингвистики, которое занимается изучением проблем компьютерного анализа и синтеза естественных языков.

Данная цель обусловила следующие задачи:

– определить понятие «обработка естественного языка»;

– выявить основные задачи обработки естественного языка;

– выявить трудности, возникающие при выполнении задач естественного языка.

Обработка естественного языка

Обработка естественного языка – общее направление искусственного интеллекта и математической лингвистики. Оно изучает проблемы компьютерного анализа и синтеза естественных языков. Применительно к искусственному интеллекту анализ означает понимание языка, а синтез – генерацию грамотного текста. Решение этих проблем будет означать создание более удобной формы взаимодействия компьютера и человека.

Понимание, распознавание естественного языка – ключевая задача, поскольку узнавание и распознавание языка живого требует колоссальных знаний языковой системы, языкового строя, их особенностей и закономерностей.

Существует 10 основных и наиболее актуальных задач обработки естественного языка.

1. Распознавание речи – процесс, ведущий к преобразованию речевого сигнала человеческого голоса в цифровую информацию.

2. Синтез речи – формирование по печатному тексту сигналов речи, то есть искусственное производство человеческой речи.

3. Анализ текста – процесс извлечения содержательной, высокого качества информации из текста на естественном языке для автоматизации процесса извлечения и анализа данных.

4. Синтез текста – это объединение слов в предложения, предложений в текст по заданной на этапе анализа прагматической структуре. Задача синтеза может рассматриваться как обратная по отношению к анализу.

5. Машинный, или автоматический перевод – процесс перевода устных текстов, написанных на естественном языке, на другой, тоже естественный, язык при помощи электронно-вычислительных машин в предназначенных для данного типа задач компьютерных программах.

6. Создание вопросно-ответных систем – системы, которые способны принимать, распознавать, классифицировать вопросы и давать ответы на них на естественном языке.

7. Информационный поиск – процесс выявления информации в документах, содержащихся в доступных системе поиска базах данных, которые соответствуют заданному запросу по тематике.

8. Извлечение информации – задачи обработки естественного языка, выполняющая автоматическое извлечение необходимых данных из источника информации, текста.

9. Анализ тональности текста – анализ лексем текста, оценка их эмоциональной окрашенности и классификация по принадлежности к нейтральному, позитивному или негативному лексическому слою языка.

10. Реферирование – сокращение объёма текста за счёт выделения основных тезисов путём поиска соответствий заданным в поиске ключевым словам и его краткое изложение.

Трудности при выполнении задач

В процессе выполнения задач возникают препятствия, создаваемые теми или иными особенностями естественного языка. Например, на качество понимания текста могут повлиять такие факторы, как:

– отнесённость языка к той или иной языковой семье, группе;
– порядок речи (прямой, обратный или свободный);
– характерные особенности национальной культуры носителей естественного языка;
– логический строй речи;
– синтаксическое построение речи;
– грамматический строй речи;
– грамотность;
– фонетические особенности речи;
– полисемичность языка;
– наличие омонимов в данном естественном языке;
– способы словообразования, присущие определённому языку;
– неологизмы, окказионализмы;
– фразеологические обороты и устойчивые выражения.

Обработка естественного языка

Компьютеры замечательно работают со структурированной информацией, например таблицами в базах данных. Но люди общаются друг с другом не таблицами, а словами. Для компьютеров это слишком сложно.

Проблемой извлечения данных машиной из обычного текста занимается особое направление искусственного интеллекта: обработка естественного языка, или NLP (Natural Language Processing).

Компьютеры не могут в полной мере понимать живой человеческий язык, однако они на многое способны. NLP может делать по-настоящему волшебные вещи и экономить огромное количество времени.

Процесс чтения и понимания текста сам по себе очень сложен. Люди часто не соблюдают логику и последовательность повествования.

Реализация какой-либо сложной комплексной задачи в машинном обучении обычно означает построение конвейера. Смысл этого подхода в том, чтобы разбить проблему на очень маленькие части и решать их отдельно. Соединив несколько таких моделей, поставляющих друг другу данные, вы можете получать замечательные результаты.

Сначала нужно разбить процесс языкового анализа на стадии и понять, как они работают.

1. Выделение предложений. Можно предположить, что каждое предложение – это самостоятельная мысль или идея. Проще научить программу понимать единственное предложение, а не целый параграф.

Можно было бы просто разделять текст по определенным знакам препинания. Но современные NLP конвейеры имеют в запасе более сложные методы, подходящие даже для работы с неформатированными фрагментами.

2. Токенизация, или выделение слов. Выделение отдельных слов или токенов – токенизация. Отделение фрагмента текста, когда встречаем пробел. Знаки препинания тоже являются токенами, поскольку могут иметь важное значение.

3. Определение частей речи. Просматривает каждый токен и старается угадать, какой частью речи он является: существительным, глаголом, прилагательным или чем-то другим. Зная роль каждого слова в предложении, можно понять его общий смысл.

Анализирует каждое слово вместе с его ближайшим окружением с помощью предварительно подготовленной классификационной модели. Она была обучена на миллионе предложений с уже обозначенными частями речи для каждого слова и теперь способна их распознавать. Данный анализ основан на статистике – на самом деле модель не понимает смысла слов, вложенного в них человеком.

4. Лемматизация. В языках слова могут иметь различные формы. Если тексты обрабатывает компьютер, он должен знать основную форму каждого слова, чтобы понимать, что речь идет об одной и той же концепции. В NLP этот процесс называется лемматизацией – нахождением основной формы (леммы) каждого слова в предложении.

5. Определение стоп-слов. Определение важности каждого слова в предложении. Например, в английском языке очень много вспомогательных слов, таких как: «and», «the», «a». При статистическом анализе текста эти токены создают много шума, так как появляются чаще, чем остальные. Некоторые NLP пайплайны отмечают их как стоп-слова и отсеивают перед подсчетом количества. Для обнаружения стоп-слов обычно используются готовые таблицы.

6. Парсинг зависимостей. Установление взаимосвязей между словами в предложении. Это называется парсингом зависимостей. Конечная цель – построение дерева, в котором каждый токен имеет единственного родителя. Корнем может быть главный глагол. Модель получает слова и возвращает результат. Однако это более сложная задача.

6.1. Поиск групп существительных. Рассматривает каждое слово в нашем предложении как отдельную сущность. Но иногда нужно сгруппировать токены, которые относятся к одной и той же идее или вещи. Используется полученное дерево парсинга, чтобы автоматически объединить такие слова.

7. Распознавание именованных сущностей (Named Entity Recognition, NER). Обнаружение существительных и связей их с реальными концепциями. NER-системы не просто просматривают словари. Они анализируют контекст токена в предложении и используют статистические модели, чтобы угадать какой объект он представляет.

Шаг 8. Разрешение кореференции. Разрешением кореференции называется отслеживание местоимений в предложениях с целью выбрать все слова, относящиеся к одной сущности. Скомбинировав эту методику с деревом парсинга и информацией об именованных сущностях и получить возможность извлечь из документа огромное количество полезных данных.

Конвейер NLP на Python

На рисунке 1 изображены стандартные этапы обычного NLP-конвейера, но в зависимости от конечной цели проекта и особенностей реализации модели, некоторые из них можно пропускать или менять местами. Все перечисленные шаги уже написаны и готовы к использованию.

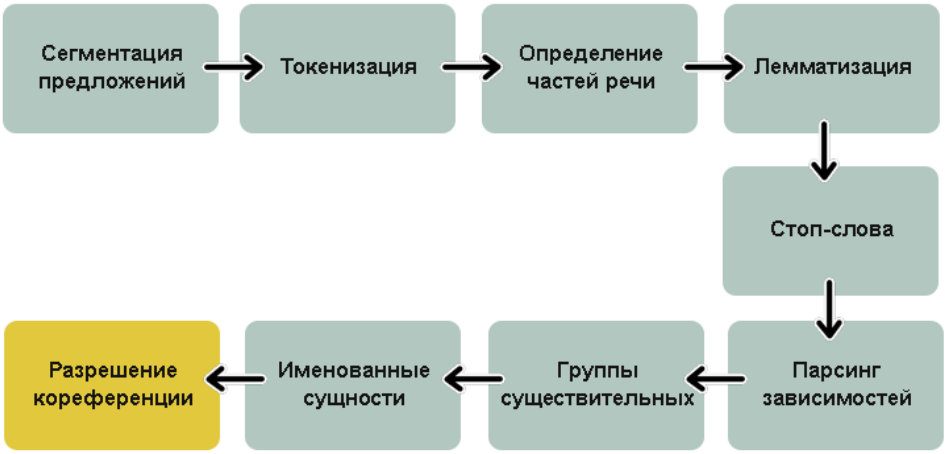


Рисунок 1 – Резюмирующая схема конвейер

Обработка естественного языка и машинное обучение

Благодаря обработке естественного языка и машинному обучению чат-боты могут интерпретировать данные, поступившие на естественном языке. Диалоговые системы помогают расшифровать эти данные в значимую информацию, и предоставляют ответ за запрос.

Многие компании пытаются разработать идеального чат-бота, который, ведёт диалог, неотличимый от обычного общения между людьми. Новые чат-боты используют глубокое обучение не только для анализа ввода человеческой речи, но и для генерирования ответов. Анализ и создание ответа достигается в результате использования алгоритма глубокого обучения, который применяется в декодировке ввода и генерировании ответа. NLP также переводит ввод и вывод в текстовый формат, понятный и компьютеру, и человеку.

Список задач, которые искусственная обработка языка должна решать. Многие из них могут быть связаны с распознаванием как текста, так и речи или даже картинок.

1. Реферирование. Задача состоит в том, чтобы создать реферат или резюме большого текста.

2. Открытые и закрытые вопросы. От современных чат-ботов ожидают готовности ответить на вопросы независимо от того, открытые они или закрытые.

3. Сопоставление. Бот должен сопоставлять объекты со словами, и понимать, когда разные слова относятся к одному объекту.

4. Двусмысленность. Двусмысленность, которая часто содержится в явлениях естественного языка, пока что представляют серьёзную проблему для ботов. Одна только омонимия требует, чтоб было выбранное правильное значение в зависимости от контекста.

5. Морфология. Чат-бот должен уметь разделить слова на морфемы.

6. Семантика. Собственно, это задача определения смысла предложений или слов в естественном языке, и генерация высказываний на естественном языке.

7. Структура текста. Связанность со структурой текста и пунктуацией.

8. Тональность. Чат-бот должен различать эмоциональную окраску высказываний человека, его отношение к предмету разговора. Должен распознавать по манере человека изъясняться, строению предложений и выбору слов, в каком настроении человек: зол, счастлив, печален.

Заключение

В статье было дано определение понятию «обработка естественного языка», а также выявлены основные задачи обработки естественного языка, и трудности, возникающие при выполнении задач. Не смотря на наличие большого количества научных публикаций и обучающих руководств на тему NLP в интернете, на сегодняшний день практически не существует полноценных рекомендаций и советов на тему того, как эффективно справляться с задачами NLP, при этом рассматривающих решения этих задач с самых основ.

Также в данной работе рассматривается понятие искусственного интеллекта, подходы к разработке и направления искусственного интеллекта.

Обработка естественного языка позволяет получать новые восхитительные результаты и является очень широкой областью.

Список использованных источников

1. Корн, Г. Справочник по математике для научных работников и инженеров / Г. Корн, Т. Корн. – М.: Наука. Главная редакция физико-математической литературы, 1974. – 832 с.

2. Джаратано, Джозеф. Экспертные системы: принципы разработки и программирование / Джозеф Джаратано. – М.: Высшая школа, 2002. – 1152 с.

3. Ясницкий, Л. Н. Введение в искусственный интеллект / Л. Н. Ясницкий. – М.: Издат. центр «Академия», 2005. – 176 с.

4. Джексон, Питер Введение в экспертные системы / Питер Джексон. – Харьков, 1997. – 112 с.

5. Большакова, Е.И. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика / Е.И. Большакова, Э.С. Клышинский, Д.Э. Ланде, А.А. Носков, О.В. Пескова. Е.В. Ягунова – М.: СССР-США СП «Параграф», 1990. – 160 с.

Столбунская А.С., Кравец Т.Н. Создание интеллектуальной системы стилистической оценки текста. *Исследование обработки естественного языка – одно из направлений искусственного интеллекта и математической лингвистики, которое занимается изучением проблем компьютерного анализа и синтеза естественных языков. Сложностью оценки как процесса и результата познавательной деятельности является проблема статуса категории оценки на уровне слова, высказывания и текста. Разработка алгоритма анализа оценки решает ряд вопросов в области исследования текста.*

Ключевые слова: *статья, интеллектуальные системы, обработка текста, обработка естественного языка, чат-боты, машинное обучение.*

Stolbunskaya A.S., Kravets T.N. Creating an intellectual system of stylistic evaluation of the text. *The study of natural language processing is one of the directions of artificial intelligence and mathematical linguistics, which is studying the problems of computer analysis and synthesis of natural languages. The complexity of evaluation as a process and result of cognitive activity is a problem of the state of categories, words and statements. The development of an evaluation analysis algorithm solves a number of questions in the field of text research.*

Keywords: *article, intelligent systems, word processing, natural language processing, chat bots, machine learning.*

УДК 519.683.8

Проектирование интеллектуального обучающего модуля «Динамические структуры данных»

Строкин В.С., Ольшевский А.И.
Донецкий национальный технический университет
vlad-strokin@yandex.ru, a_olshevskiy@mail.ru

Строкин В.С., Ольшевский А.И. Проектирование интеллектуального обучающего модуля «Динамические структуры данных». *Рассмотрена классификация основных динамических структур данных. Описаны способы представления и реализации различные динамических конструкций. Определена последовательность построения курса обучения и подходы формирования набора поддерживающих вопросов для обучения. Разработана модель обучаемого и процедура интеллектуального управления процессом обучения. Проанализирована эффективность обучения по набору ответов обучаемого.*

Ключевые слова*: динамические структуры данных, наборы операций, вычислительная сложность различных операций, построения курса обучения, модель обучаемого, интеллектуальный анализ.*

Общая постановка проблемы

В современных условиях требуется новый взгляд на формирование компетентностей – охват всего процесса приобретения знаний, умений и навыков. Передовые страны проводят новую образовательную политику, опирающуюся на инженерный подход, связанный с индивидуализацией процесса обучения. Реализация такой обучающей системы возможна на основе результатов следующих исследований в области искусственного интеллекта (ИИ): динамические интеллектуальные системы, много агентные системы, онтологии, эволюционирующие знания и некоторые другие [1].

Постановка задачи

Для написания компьютерных программ обычно используют метод, который уже был когда-то разработан для решения какой-либо задачи. Часто этот метод не зависит от конкретного используемого компьютера – вполне вероятно, что он будет одинаково пригодным для многих компьютеров и многих языков программирования [2]. Термин алгоритм используется в компьютерных науках для описания метода решения задачи, пригодного для реализации в виде компьютерной программы.

Основная побудительная причина изучения алгоритмов состоит в том, что это позволяет обеспечить огромную экономию ресурсов, вплоть до получения решений задач, которые в противном случае были бы невозможны.

Любая программа предназначена для обработки данных, от способа организации которых зависят алгоритмы работы, поэтому выбор структур данных должен предшествовать созданию алгоритмов [3]. Задача создания обучающей программы является актуальной, ведь прежде, чем выполнить какие-либо операции, нужно иметь объекты, к которым они применимы.

Цель проекта – накопление и систематизация информации по изучению динамических структур данных (ДСД). Обучающий модуль позволит уменьшить время обучения пользователей, а при использовании динамических структур в своих программах повысит быстродействие работы их при экономии памяти.

Разрабатываемый модуль является частью обучающей системы по курсу «Алгоритмизация и программирование».

Исследования

Память под данные выделяется либо на этапе компиляции (в этом случае необходимый объем должен быть известен до начала выполнения программы, то есть задан в виде константы), либо во время выполнения программы с помощью операции new или функции malloc (необходимый объем должен быть известен до распределения памяти). В обоих случаях выделяется непрерывный участок памяти.

Если до начала работы с данными невозможно определить, сколько памяти потребуется для их хранения, память выделяется по мере необходимости отдельными блоками, связанными друг с другом с помощью указателей. Такой способ организации данных называется динамическими структурами данных,

поскольку их размер изменяется во время выполнения программы. Из динамических структур в программах чаще всего используются линейные списки, стеки, очереди и бинарные деревья [2-3]. Они различаются способами связи отдельных элементов и допустимыми операциями. Динамическая структура может занимать несмежные участки оперативной памяти.

Динамические структуры широко применяют и для более эффективной работы с данными, размер которых известен, особенно для решения задач сортировки, поскольку упорядочивание динамических структур не требует перестановки элементов, а сводится к изменению указателей на эти элементы. Например, если в процессе выполнения программы требуется многократно упорядочивать большой массив данных, имеет смысл организовать его в виде линейного списка. При решении задач поиска элемента в тех случаях, когда важна скорость, данные лучше всего представить в виде бинарного дерева.

Элемент любой динамической структуры данных представляет собой структуру (в смысле struct), содержащую, по крайней мере, два поля: для хранения данных и для указателя. Полей данных и указателей может быть несколько. Поля данных могут быть любого типа: основного, составного или типа указатель.

Методология решения задачи на компьютере в общем случае рассматривает совместную «деятельность» человека и компьютера. Непосредственное решение задачи – это процесс автоматического преобразования исходных данных в искомый результат в соответствии с заданным алгоритмом [4-5].

Для создания обучающего модуля в общем случае, решаются следующие задачи:

− подготовки и систематизации учебного материала, адаптации материала по уровням сложности, разработки динамических иллюстраций, контрольных вопросов и других заданий (база знаний);

− регистрации пользователей (создания модели обучаемого), проверки уровня знаний и статистического сбора показателей усвоения учебного материала;

− оценки уровня знаний, умений и навыков у обучаемых до и после обучения, их индивидуальных способностей и мотиваций;

− администрирования системы, доставки учебного материала на рабочие станции и задачи обратной связи с обучаемым.

Для реализации общей обучающей системы по курсу «Алгоритмизация и программирование», необходимо результаты указанных исследований (разрабатываемый модуль) интегрировать в рамках одной системы для формирования компетентностей обучающегося в единой информационно-образовательной среде. Принцип модульности является средством упрощения задачи проектирования программного обеспечения и распределения процесса разработки между отдельными группами разработчиков. При разбиении программы на модули для каждого модуля указывается реализуемая им функциональность, а также связи с другими модулями [6]. Удобство использования модульной архитектуры заключается в возможности обновления (замены) модуля, без необходимости изменения остальной системы.

Программная реализация

Интеллектуализация компьютерного обучения предполагает использование методов и моделей представления знаний на базе систем, основанных на знаниях. При программировании комплекса необходимо сначала подготовить и систематизировать учебный материал, а затем адаптировать материала по уровням сложности.

Информационная база (учебный материал по ДСД) спроектирована по уровням сложности:

– классификация структур от простых (линейных: списки; стеки; очереди) к сложным (нелинейным: бинарные деревья; АВЛ сбалансированные деревья; графы);

– внутри каждой структуры по трем разделам: общие понятия и определения; примеры исходного кода программы и программы демонстрации; оценка вычислительной сложности для различных типов данных и разных операций.

Работа с базой знаний предполагает использование интернет-ориентированных систем и языков гипертекстовой разметки HTML, XML, SGML. Поэтому, после регистрации пользователя создается модель обучаемого и предлагается сначала просмотреть обучающий курс (презентацию), который имеет удобную систему навигации на основе гиперссылок и· включает в себя аудио- и видеофрагменты, а также поддерживает удобную систему поиска по ключевым словам и понятиям.

Модель деятельности обучаемого включать в себя информацию начиная с регистрации, сбора статистических показателей усвоения учебного материала: определение количества входов в систему; временные характеристики в том числе и время затраченное на решения тестов; определение общего числа ошибок и т.д. Компьютерные модели включают множество аспектов моделируемой реальности, обеспечивают большую гибкость при решении задач, позволяют управлять временем и пространством, повторять или изменять ситуацию, дополнять модель графикой, мультипликацией, звуковым сопровождением[7].

Система позволяет осуществлять углубленную индивидуализацию обучения на основе логико-оптимизационных методов автоматизации планирования действий с веб-интерфейсом для удалённых пользователей.

Для реализации принципов индивидуального обучения обучающая система должна включать модель

обучаемого и модель предметной области. На рисунке 1 представлена общая схема функционирования обучающего модуля.

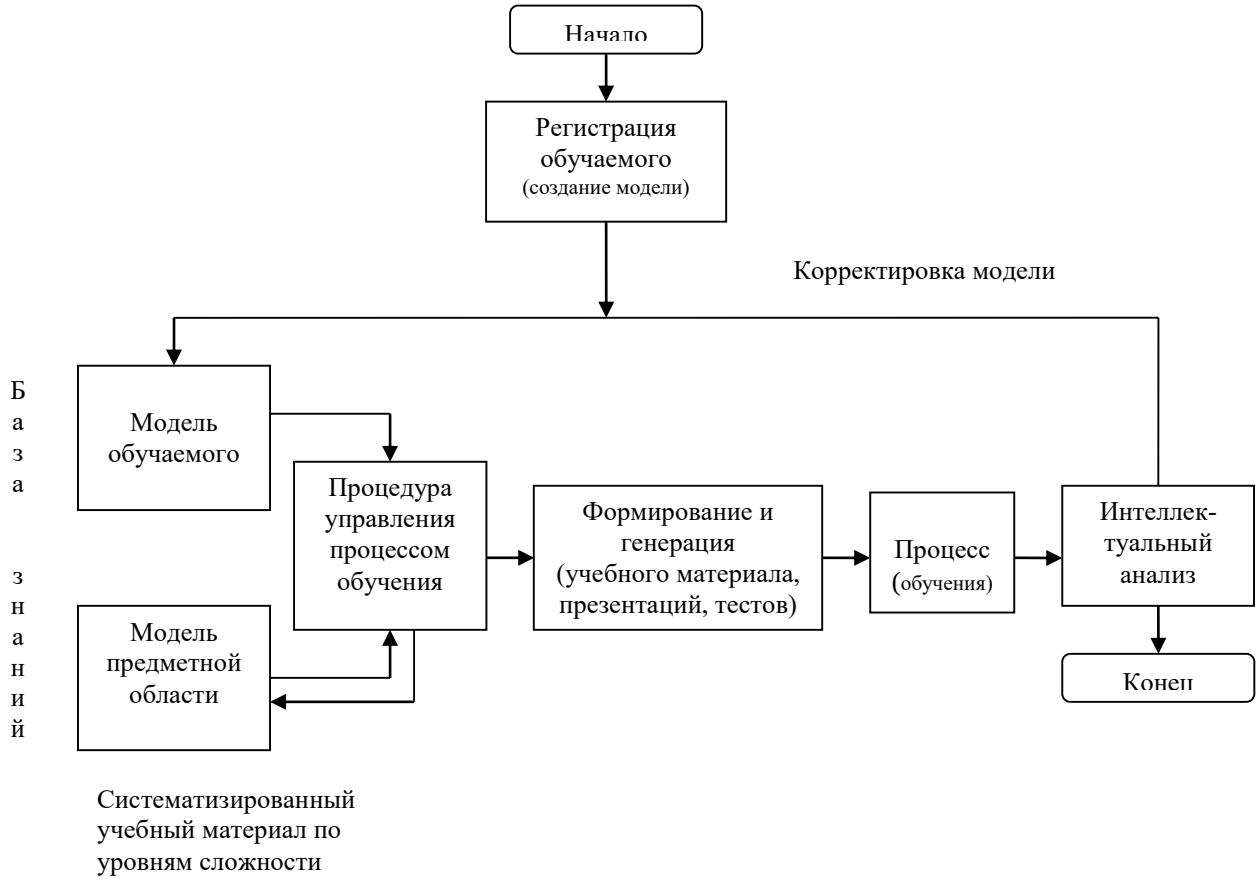


Рисунок 1 – Общая схема функционирования обучающего модуля

Для оценки уровня знаний, обучаемым предлагаются новый учебный материал и тестовые задания, которые генерируются автоматически с учетом их индивидуальных способностей и мотиваций (анализ модели обучаемого).

Центральное место в системе занимает процедура управления процессом обучения, которая используется для планирования и выбора дальнейших действий (логические «решатели», методы теории принятия решений). Для генерации новой порции учебного материала и помощи обучаемому задействуется база знания о предметной области и текущее состояние модели обучаемого. Сведения извлекаются по запросу анализатором действий системы управления. После удачного прохождения всех этапов обучаемому предлагается следующая тема из этого курса.

Интеллектуальный анализ результатов – это процедура проверки, при которой обучающая система способна проанализировать ответы студента на тестовые задания, указать, что именно неправильно или неполно освещено в ответе и, как следствие, какие знания недостаточно усвоены студентом.

Метазнания о компетенции обучаемого первоначально формируется путём диагностирования обучаемого с помощью тестов, а также на умение выполнять простейшие или более сложные упражнения. При этом осуществляется последующее итеративное уточнение компетентностной модели обучаемого по результатам его ответов в ходе изучения им очередной порции материала, предложенной ему с учётом достигнутой компетенции.

В результате применения этого метода к исходным данным создается иерархическая (древовидная) структура правил вида «Если … то …», а алгоритм анализа обеспечивает процесс вычисления на каждом этапе наиболее значимых условий и переходов между ними. Процесс обучения либо продолжается с учетом корректировки модели, либо заканчивается.

Выводы

Подготовлен и систематизирован учебный материал по уровням сложности, разработаны динамические иллюстрации и тестовые задания. Определена последовательность построения курса обучения, модель обучаемого и процедура управления процессом обучения, которая используется для планирования и

выбора дальнейших действий. Разработана процедура проверки уровня знаний по данным статистического сбора показателей усвоения учебного материала.

Для реализации общей обучающей системы по курсу «Алгоритмизация и программирование», необходимо данный модуль интегрировать в единую информационно-образовательную среду.

Литература

1. Юрков, Н. К. Интеллектуальные компьютерные обучающие системы [Электронный ресурс] / Н. К. Юрков //. – Режим доступа : http://www.mtas.ru/upload/library /MONOGRAFIYa_IKOS_2010.pdf
2. Павловская, Т. А. C/C++. Программирование на языке высокого уровня / Павловская Т. А. – СПб. : Питер, 2003. – 461 с.
3. Шилдт Г. C++: базовый курс / Шилдт Г. [пер. с англ.]. – [3-е изд.]. – М. : Вильямс, 2010. – 624 с.
4. Трембач, В.М. Основные этапы создания интеллектуальных обучающих систем / Трембач В.М. // Программные продукты и системы – 2012. – №3. – С. 147-151.
5. Интеллектуальные обучающие системы [Электронный реурс]. – Режим доступа : http://studbooks.net/1593672/informatika/intellektualnye_obuchayuschie_sistemy.
6. Ольшевский, А.И. Разработка обучающего пакета программ работы с контейнерными классами / А.И. Ольшевский // Информатика и кибернетика. – 2017. – № 1. – С. 110 – 116.
7. Интеллектуальное управление процессом обучения [Электронный реурс]. – Режим доступа : https://habr.com/post/194240/

Строкин В.С., Ольшевский А.И. Проектирование интеллектуального обучающего модуля «Динамические структуры данных». *Рассмотрена классификация основных динамических структур данных. Описаны способы представления и реализации различные динамических конструкций. Определена последовательность построения курса обучения и подходы формирования набора поддерживающих вопросов для обучения. Разработана модель обучаемого и процедура интеллектуального управления процессом обучения. Проанализирована эффективность обучения по набору ответов обучаемого.*

Ключевые слова: *динамические структуры данных, наборы операций, вычислительная сложность различных операций, построения курса обучения, модель обучаемого, интеллектуальный анализ.*

Strokin Vladislav, Olshevsky Andrey. Designing an intelligent learning module "Dynamic data structures". *The classification of the main dynamic data structures is considered. Describes how to represent and implement a variety of dynamic structures. The sequence of constructing a course of study and approaches to the formation of a set of supporting questions for training are determined. A model of a student and a procedure for intellectual management of the learning process have been developed. Analyzed the effectiveness of training on the set of responses of the student.*

Key words: *dynamic data structures, sets of operations, the computational complexity of various operations, the construction of a course of study, the model of the student, intellectual analysis.*

УДК 004.932

Параметрическое описание моделей глубоких нейронных сетей в библиотеке Keras

Ткачёв Н.М., Федяев О.И.
Донецкий национальный технический университет
m4dbrat@gmail.com, fedyaev@donntu.org

Ткачёв Н.М., Федяев О.И. Параметрическое описание моделей глубоких нейронных сетей в библиотеке Keras. *В статье представлен обзор различных видов глубоких нейронных сетей, их свойств, особенностей и предназначения. Рассмотрена высокоуровневая библиотека Keras, предназначенная для прототипирования нейронных сетей. На примерах рассмотрено описание элементов архитектуры нейронной сети на языке Python.*

Ключевые слова: *искусственная нейронная сеть, Keras, Python, глубокая свёрточная нейросеть, рекуррентная нейросеть .*

Введение

Нейросетевые технологии являются одним из ведущих направлений искусственного интеллекта. Построенные с задумкой имитации работы головного мозга человека, искусственные нейронные сети позволяют автоматизировать широкий спектр трудноформализуемых задач. Распознавание и генерация визуальных образов и речи, обработка естественных языков, различного рода прогнозирование и даже создание произведений искусства – лишь немногие из широкого спектра решаемых нейросетями задач.

Долгое время нейронные сети считались непрактичными и развивались лишь теоретически [1]. Однако, новые теоретические разработки в этой области в значительной мере способствовала их популярности. В 2006-2007 гг. в своих публикациях Джеффри Хинтон продемонстрировал возросший потенциал глубоких нейронных сетей созданием нового способа их обучения [2]. Он показал, что каждый слой глубокой сети можно обучать по отдельности на большом объеме данных, а затем провести остаточную тонкую настройку всей сети методом обратного распространения ошибки; полученная в результате сеть обладала высокой производительностью. Значительным фактором в распространении нейросетей стали возможности аппаратного ускорения вычислений при помощи таких библиотек, как Nvidia CUDA и OpenCL. Делегация матричных вычислений графическим процессорам позволила ускорить обучение и работу глубоких сетей в отдельных случаях на несколько порядков. В 2014 году была продемонстрирована модель глубокой свёрточной сети VGG-16, достигшая значительных результатов в точности распознавания изображений. Эта модель показала, что точность работы глубокой сети значительно вырастает при увеличении количества слоёв до 16-19 по сравнению со старыми моделями, использовавшими до 10 слоёв [3].

В наше время приложения с функционалом, построенным на глубоких нейронных сетях, широко распространены в потребительском секторе. Современные мобильные устройства могут с высокой точностью распознавать рукописный текст, активно улучшаются автономные системы распознавания речи, программы автоматической корректировки и дополнения набираемого текста способны подстраиваться под каждого отдельного пользователя. Функция распознавания лиц из видеопотока с камеры показывает, что нейронные сети с современными архитектурами способны работать в режиме реального времени даже на устройствах с ограниченными ресурсами.

Архитектуры глубоких нейронных сетей

Свёрточные нейронные сети (convolutional neural networks, CNN) [4] являются самой распространённой формой глубоких сетей. Главной особенностью CNN является то, что они способны обрабатывать входные данные сразу на нескольких уровнях абстракции путём поиска различного рода признаков, что позволяет более точно и с лучшей производительностью проводить классификацию входных данных. Также свёрточные сети частично устойчивы к различного рода искажениям (например, поворот изображения). Одним из главных достоинств CNN является простота распараллеливания работы и обучения, что даёт значительный прирост в производительности, в том числе благодаря возможности проведения вычислений на графических процессорах. Типичная свёрточная сеть состоит из входного слоя, нескольких последовательностей свёрточных слоёв, закрываемых подвыборочным (pooling) слоем, пары полносвязных (dense) слоёв и выходного слоя (см. рис. 1).

Рисунок 1 – Пример структуры свёрточной сети

Свёрточный слой генерирует матрицу выходных сигналов путём последовательного обхода матрицы входных данных матрицей весов. На каждом этапе матрице весов в соответствие ставится некоторый фрагмент исходной матрицы, соответствующие ячейки матриц умножаются, затем все результаты умножения суммируются и конечный результат записывается в ячейку матрицы выхода. Затем матрица весов смещается в сторону на одну ячейку (реже – на две или более) и процесс повторяется, пока матрица выходов не заполнится. Размеры матрицы выходов зависят от размеров исходной матрицы и матрицы весов. Количество возможных смещений в каждом направлении вычисляется по формуле: длина_матрицы_входа – длина_матрицы_весов + длина_смещения. Например, при обработке изображения размером 5x5 матрицей весов размером 2x2 матрица выходов будет иметь размер 4x4 (рис. 2), при матрице весов 3x3 – 3x3, 3x2 – 3x4. Каждая последовательность свёрточных слоёв решает задачу поиска признаков. Количество нейронов в каждом последующем слою уменьшается.

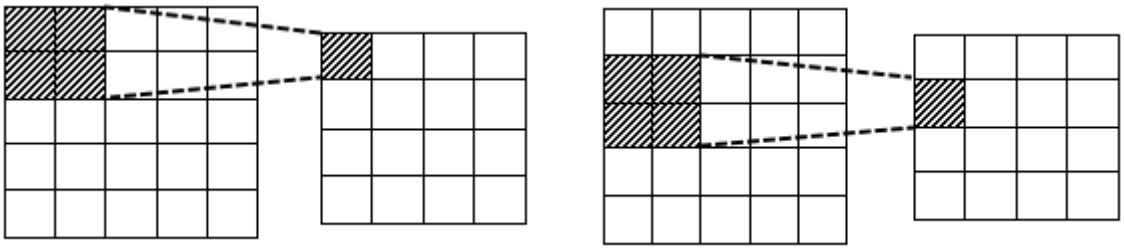


Рисунок 2 – Примеры шагов операции свёртки изображения 5x5 матрицей весов 2x2

Подвыборочный, или пулинговый слой используется для значительного уменьшения размерности матриц с целью повышения производительности (см. рис. 3). Пулинг применим тогда, когда предыдущие слои способны точно определить ярко выраженные признаки. После пулинга следующие слои пытаются найти признаки более высокого уровня абстракции. Аналогично свёрточному слою, подвыборочный слой генерирует свою выходную матрицу путём последовательного обхода матрицы входных данных, но с двумя различиями: 1) фрагменты входной матрицы не пересекаются (потому длина матрицы выхода будет равна длине матрицы входа, делённой на длину матрицы пулинга); 2) пулинговый слой не имеет весовых коэффициентов, и проводит операцию исключительно на входах. Существует множество видов операций пулинга, из них основные: max-pooling (выбор наибольшего значения) и усреднённый пулинг (вычисление среднего арифметического).

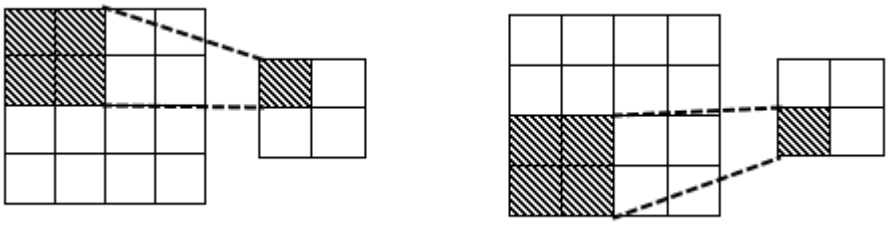


Рисунок 3 – Примеры шагов операции подвыборки изображения 4x4 матрицей 2x2

Полносвязные слои выполняют задачу классификации на основе данных о найденных признаках, полученных из предыдущих слоёв. Вместе с выходным слоем они работают по принципу многослойного перцептрона. Результатом работы полносвязных слоёв является вектор вероятностей принадлежности входного сигнала к каждому из классов.

Основной сферой применения свёрточных нейросетей является распознавание изображений (определения как одного, так и множества образов на сцене), но также они могут использоваться в других сферах, требующих комплексного распознавания признаков и классификации (таких, как распознавание текстов на естественных языках и определение диагнозов пациентов). Свёрточные сети могут объединяться с другими типами нейросетей для решения более сложных задач. Например, распознавание действий на видео или генерация голосового описания изображения решаются путём сложения свёрточной и рекуррентной сети.

Рекуррентные нейронные сети (recurrent neural networks, RNN) [5] содержат нейроны, которые способны сохранять входные данные для будущего использования. Такие сети способны определять зависимости между последовательными наборами входных сигналов, потому они идеально подходят для анализа любых сигналов, зависящих от времени. На практике они применяются для распознавания голоса, рукописного ввода, прогнозирования временных рядов, обработки естественных языков, перевода, распознавания эмоциональных оттенков и т.п.

Классические RNN анализируют каждый новый входной сигнал с учётом предыдущего. Такой подход достаточно прост и мало чем отличается от обыкновенных многослойных перцептронов в плане реализации, но при этом может происходить потеря контекста. Зачастую контекст, в котором необходимо распознавать текущий входной сигнал, может определяться множеством входных сигналов более ранних, чем предыдущий. Проблему запоминания долгосрочных зависимостей решают сети с долгой краткосрочной памятью (long short-term memory, LSTM). Существует множество вариаций LSTM-сетей, но в общем случае LSTM-нейроны содержат 3 вентиля: вентиль входа (определяет, насколько данные в памяти влияют на текущий вход), вентиль забытия (насколько вход влияет на данные в памяти) и вентиль выхода (насколько данные в памяти влияют на выход). Весовые коэффициенты этих вентилей настраиваются при обучении.

Нейросетевая библиотека Keras

Библиотека Keras [6] является высокоуровневым решением для создания нейронных сетей различных архитектур. Написанная на языке Python, Keras позиционируется как средство быстрого прототипирования и тестирования конфигураций нейросетей различной сложности, максимально упрощая процесс создания новых слоёв и их совмещения в единую модель.

При установке Keras необходимо установить back-end, который представляет собой более низкоуровневую реализацию всех используемых возможностей нейросетей. Основными back-end-ами являются TensorFlow и Theano, при этом разрабатывается поддержка других. По умолчанию Keras пытается использовать TensorFlow, но в файле `$HOME/.keras/keras.json` (`%USERPROFILE%` вместо `$HOME` при использовании ОС Windows) значение поля “backend” можно установить в нужное. Также можно задать новое значение переменной среды KERAS_BACKEND, тогда конфигурационный файл будет изменён автоматически. Также Keras может автоматически использовать графические процессоры для ускорения работы, если установлена поддерживающая это версия back-end-а. При написании кода можно как полностью абстрагироваться от back-end-а, используя общий API, так и использовать специализированные функции конкретного back-end-а.

Keras имеет две модели построения нейросети: последовательная (sequential) и функциональная (functional). Последовательная модель представлена классом Sequential (рис. 4). Эта функция в качестве аргументов принимает список слоёв нейронов. Также эту функцию можно вызвать без аргументов, а слои добавить с помощью функции add().

```
from keras.models import Sequential
from keras.layers import Dense, Activation

model = Sequential([
    Dense(32, input_shape=(1000,)),
    Activation('relu'),
    Dense(1),
    Activation(' sigmoid'),
])
```

Рисунок 4 – Создание последовательной модели в Keras

При использовании функциональной модели каждый последующий слой после создания должен получить в качестве аргумента модели, построенной из предыдущих слоёв.

Первый слой модели должен также определить форму входных данных аргументом input_shape (например, для изображений размером 50x50 пикселей с тремя цветовыми каналами input_shape = (50, 50, 3) [7]). Для последующих слоёв модели этот параметр определяется автоматически, основываясь на количестве выходов предыдущих слоёв.

После создания модели она компилируется при помощи метода compile(), принимающего в качестве аргументов функцию потерь и оптимизатор; среди дополнительных параметров можно указать измеряемые метрики (рис. 5).

```
model.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

Рисунок 5 – Компиляция модели

Затем полученная сеть обучается при помощи метода fit(). Данный метод принимает в качестве аргументов некоторый входной сигнал и поставленный ему в соответствие желаемый выходной сигнал. Данные должны быть представлены в форме массивов NumPy. Среди дополнительных параметров можно указать, например, количество эпох.

```
import numpy as np
data = np.random.random((1000, 100))
labels = np.random.randint(2, size=(1000, 1))

model.fit(data, labels, epochs=10, batch_size=32)
```

Рисунок 6 – Обучение сети на случайно сгенерированных данных

После обучения нейронную сеть можно проверить с использованием метода evaluate(), передав в него входной и желаемый выходной сигнал; на выходе будет получена потеря и желаемые метрики. Для вычисления выходного сигнала на основе входа используется метод predict().

Примеры параметрического описания типовых нейросетей с использованием Keras

Простые свёрточные слои в библиотеке Keras представлены классами Conv1D, Conv2D и Conv3D, соответствующим одно-, двух- и трёхмерным обрабатываемым данным. Обязательными аргументами при создании этих слоёв являются filters (количество фильтров – свёрточных нейронов) и kernel_size (размерности ядра – матрицы весов). Параметр stride указывает, насколько смещается ядро при обходе матрицы входа (по умолчанию 1).

Параметр padding расширяет матрицу входа. Установка его значения в “same” позволяет дополнять матрицу входа нулями, чтобы размеры матрицы выхода были равны размерам матрицы входа. По умолчанию его значение равно “valid”, что соответствует отсутствию расширения. Аналогичного поведения можно добиться, используя перед свёрточным слоем специальный слой ZeroPadding(1D, 2D, 3D), но при этом количества ячеек, на которые матрица расширяется в каждом измерении и в каждом направлении, можно задать вручную через аргумент padding.

Параметр activation задаёт функцию активации. По умолчанию функция активации в свёрточных слоях линейная (“linear”), т.е. выход равнозначен входу. В Keras уже встроено множество часто используемых функций активации. В свёрточных сетях чаще всего используется функция ReLU (rectified linear unit, выпрямленный линейный элемент), которая даёт достаточно точные результаты при своей низкой ресурсозатратности (при отрицательных входах она передаёт 0, при положительных – сам вход). Также часто используются гиперболический тангенс (“tanh”) и сигмоида (“sigmoid”). На практике вместо передачи функции активации в качестве параметра свёрточной сети её выделяют в отдельный слой Activation. Также существуют продвинутые слои активации, имеющие обучаемые элементы (LeakyReLU, PReLU, Softmax и пр.).

Слои подвыборки, аналогично свёрточным слоям, имеют 1D, 2D и 3D-версии. Классы MaxPooling при обходе матрицы входа выбирают максимальные значения, AveragePooling – вычисляют средние. Параметр pool_size определяет размеры окна подвыборки (по умолчанию каждое измерение равно 2). У каждого подвыборочного класса есть Global-аналог (например, GlobalMaxPooling2D), который устанавливает размерами окна размеры исходной матрицы (т.е. на выходе получается единственное значение).

Параметр strides, аналогично свёрточным слоям, определяет шаг смещения окна подвыборки в каждом измерении (по умолчанию оно равно размерам окна подвыборки). Параметр padding работает так же, как и в свёрточных слоях. Подвыборочные слои в качестве функции активации всегда используют ReLU, поэтому параметр, который задаёт функцию активации, отсутствует.

Полносвязные слои представлены классом Dense. Обязательный параметр units описывает количество нейронов (и, соответственно, выходных сигналов) в слое. Параметр useBias указывает, стоит ли использовать вектор смещений (по умолчанию - True). Слои Dense всегда принимают одномерный входной сигнал, поэтому в свёрточных сетях, использующих более одного измерения, перед первым полносвязным слоем добавляется слой Flatten, преобразующий данные в одномерную форму.

На практике иногда нейронные сети могут переобучаться и становиться слишком требовательными к входным данным, отказываясь распознавать некоторые признаки даже при малейших отличиях входных данных от тренировочных. В случае возникновения этой проблемы на некоторых этапах обучения сети можно случайно отбрасывать [8] выходы нейронов, устанавливая их в 0. В библиотеке Keras этим занимается слой Dropout. Параметр rate слоя Dropout задаёт, какая доля нейронов будет случайно отброшена. Отброс осуществляется только на этапе обучения, при работе с реальными данными слой полностью игнорируется.

Заключение

Библиотека Keras является мощным инструментом для моделирования нейронных сетей. В библиотеке реализован широкий набор видов слоёв глубоких нейронных сетей, каждый из которых обладает широким спектром параметров для тонкой настройки. При этом количество обязательных параметров при настройке каждого слоя сведено к минимуму, а необязательные устанавливаются в наиболее часто используемые либо рассчитываются автоматически. Композиция слоёв проста, нет необходимости писать какой-либо код для передачи или адаптации данных между слоями. Использование в качестве языка программирования Python, ввиду его простоты в использовании, ёмкости и распространенности, позволяет даже малоопытным программистам решать трудноформализуемые задачи наподобие распознавания изображений.

Литература

1. Искусственная нейронная сеть – Википедия [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Искусственная_нейронная_сеть
2. Deep Learning – Wikipedia [Электронный ресурс] – Режим доступа: https://en.wikipedia.org/wiki/Deep_learning
3. Джулли, А, Пал С. Библиотека Keras – инструмент глубокого обучения. Реализация нейронных сетей с помощью библиотек Theano и TensorFlow / пер. с англ. Слинкин А.А. – М.: ДМК Пресс, 2018 – с.101.
4. Сверточная нейронная сеть, часть 1: структура, топология, функции активации и обучающее множество – Хабр [Электронный ресурс] – Режим доступа: https://habr.com/post/348000/
5. Рекуррентные нейронные сети: типы, обучение, примеры и применение – Neurohive [Электронный ресурс] – Режим доступа: https://neurohive.io/ru/osnovy-data-science/rekurrentnye-nejronnye-seti/
6. Keras: The Python Deep Learning library [Электронный ресурс] – Режим доступа: https://keras.io/
7. Keras input explanation: input_shape, units, batch_size, dim, etc – Stack Overflow [Электронный ресурс] – Режим доступа: https://stackoverflow.com/questions/44747343/keras-input-explanation-input-shape-units-batch-size-dim-etc
8. Dropout: A Simple Way to Prevent Neural Networks from Overfitting [Электронный ресурс] – Режим доступа: https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf

Ткачёв Н.М., Федяев О.И.. Параметрическое описание моделей глубоких нейронных сетей в библиотеке Keras. *В статье представлен обзор различных видов глубоких нейронных сетей, их свойств, особенностей и предназначения. Рассмотрена высокоуровневая библиотека Keras, предназначенная для прототипирования нейронных сетей. На примерах рассмотрено описание элементов архитектуры нейронной сети на языке Python.*

Ключевые слова: *искусственная нейронная сеть, Keras, Python, глубокая свёрточная нейросеть, рекуррентная нейросеть .*

Tkachev Nickolay, Fedyaev Oleg. Parametric description of deep neural network models in the Keras library. *The article provides an overview of various deep neural network types, their capabilities and uses. A high-level neural network prototyping library Keras is showcased. Descriptions of neural network architecture elements are provided through examples using the Python programming language.*

Key words: *artificial neural network, Keras, Python, deep convolutional neural network, recurrent neural network.*