**ВВЕДЕНИЕ**

Разрабатываемый продукт представляет собой экспертную систему ведения интернет торговли (ЭСИТ). ЭСИТ рассчитывает прогноз продаж путём использования уникального алгоритма, построенного в ходе разработки системы.

Алгоритм прогноза в ЭСИТ опирается на статистику продаж по прошедшим периодам. Для работы со статистическими данными был разработан модуль управления базой знаний (МУБЗ). Помимо таких задач как внесения информации в базу данных и ее чтение, МУБЗ выполняет анализ актуальности данной информации. Информация является актуальной в течение определенного времени, обычно такое время составляет три прошедших периода. Понятие период для разной групы товаров может быть разным, поэтому длительность одного периода выставляется индивидуально пользователем в настройках.

ЭСИТ так же предоставляет инструментарий, который автоматизирует управление бизнес прецессами. Опираясь на результаты прогноза, система сама формирует список рекомендуемых товаров для закупки, то есть те товары, спрос на которые растет или, как минимум, остается стабильным, выводит списки пользователю на экран. При отсутствии такой автоматизации владельцу интернет-бизнеса придется самому выполнять этот рутинный анализ. Для интернет магазинов с широким ассортиментом товаров ручной анализ обходится довольно дорого как по времени, так и по финансам.

Экспертная система ведения интернет торговли - это не только запрограммированный алгоритм, а действительно целая система, состоящяя из нескольких частей, каждая из которых выполняет свою функцию. На рисунке 1 изображена структура разрабатываемой системы.

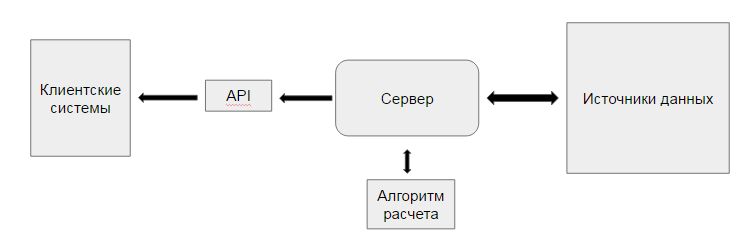


Рисунок 1 - Структура разрабатываемой системы

На сервере находится основная часть разработанного програмного обеспечения, включая алгоритм прогнозирования, модуль управления базой знаний и непосредственно базу данных.   
 Источник данных - это место хранения информации о товаре, это могут быть данные как со склада компани, занимающейся оптовой торговлей, так и данные с собственного склада.   
 API предназначен для того, чтобы наладить взаимодействие с интернет магазинами, которые решили подключить разработанную систему. Именно посредством API клиенты имеют возможность внедрять готовые функции от ЭСИТ в собственные системы, например вывод списка товара для интернет магазина. Гибкость разработанного API позволяет внедрять функциональность не только в веб-версии своих проектов, а так же в мобильные или настольные приложения.  
 Клиентские системы представляют собой такие системы, в которые будет внедряться функциональность ЭСИТ. К ним относятся веб-версии интернет-магазинов, мобильные а также настольные приложения разработанные под различные операционные системы.

**1 АКТУАЛЬНОСТЬ**

На сегодняшниий день рынок электронной коммерции в России активно развивается. Согласно данным исследовательского агентства Data Insight, объем российского рынка электронной коммерции в 2011 году составил 310 млрд. рублей, в 2012 – 392 млрд. рублей (рост на 26%).

Многим ведущим представителям розничной интернет-торговли удалось добиться увеличения продаж на 200-300%.  
 Важно отметить, что темпы роста данного сектора экономики значительно опережают темпы роста экономики страны: за 2012 год ВВП России вырос на 3,5%, а e-commerce – на 26%.  
 Согласно мнению экспертов компании   
J’son & Partners Consulting, в ближайшие годы ожидаются стабильные темпы роста на уровне 15-20% ежегодно, в то время как к 2020 году объем рынка электронной коммерции составит более 2180 млрд. руб. (рисунок 1).



**Рисунок 1 – Объем рынка электронной коммерции в России (млрд. руб.)**

Следует отметить тот факт, что сегмент электронной коммерции в экономике нашей страны стал заметен только за последние несколько лет. До 2011 года доля продаж посредством интернета составляла менее 1% от общего объема торговли России, в 2012 – около 2% (рисунок 2). Схожие показатели имели место в США и Великобритании в 2003 и 2005 годах, на сегодняшний день доля е-commerce в данных странах занимает более 10%, с чего мы можем сделать вывод о наличии определеннго потенциала для дальнейшего роста рынка электронной коммерции России.

Справедливо заметить, что на Российском рынке интернет-торговле присутствует дисбаланс в развитии (рисунок 3). Примерно 60% интернет-пректов ориентированы на большие город, такие как Москва или Санкт-Петербург , где проживают около 15% населения страны. Эксперты из Morgan Stanley прогнозируют что к 2019-2020 году объемы продаж через сеть интернет для вырастут на 30% для столичных городов , и примерно 55% для региональных городов.



Рисунок 3 – Прогноз распределения продаж через Интернет по регионам России

По результатам исследований за 2016 год, на рынке российской интернет коммерции преимущественно распростроняются цифровая продукция, доля которой 64% в рублях эта оценка состовляет 254,9 млрд. (рисунок 4). Большинство покупок в коммерческих онлайн сервисах это как правило бронь билетов, аудио, видео или любая другая цифровая продукция, бронь гостиниц или оплата путевок. Так же онлайн комерция часто используется для оплаты коммунальный услуг, мобильной связи или штрафов.



Рисунок 4 – Структура рынка электронной коммерции в России 2012 и 2017гг.

Что касается реально существующих товаров для которых необходима курьерская доставка их доля на рынке интернет торговли состовляет 36% от общей доли рынка, с особой интенсивностью растут продажи обуви, одежды, косметики а так же парфюмерии. Важно то что эта ниша активно развивается с того момента когда сформировался рынок, в конце 2011 - начале 2012 года общая сумма продаж состовляла 136,9 млрд. рублей, по результатам анализа за 2016 год, этот показатель вырос в уже двое. Эксперты прогнозируют что в ближайшие год-два доля продаж физических товаров закрепит свои позиции на 40 - 45% от общей суммы продаж на рынке интернет торговли. В Российской интернет коммерции присутствует ряд изьянов которые тормозят ее развитие. Одним из таких одним их таких изьянов является то что продвижением занимаются преимущественно, специалисты в области веб-разработки и ИТ, а не специалисты в области торговли и ведения бизнеса. Это приводит к большему количеству не жизнеспособных разработок, сложных в эксплуатации виртуальных магазинов. Из этого мы можем сделать вывод что на сегоднешний день все кто смог занять нишу российской электронной коммерции это мизерный процент от того количества предпренимателей которые пытались занять эту но их проекты оказались не жизни способными, из-за отсутствия опыта ведения интернет бизнеса. Согласно результатам опросса, многие придприниматели как начинающие так и опытные готовы вливать финансы на развитие собственного электронного предприятия. Однако в силу того что их опыт не позволяет сделать правильный выбор в сотовлении списков и растоновки им приоритетов для вывода на виртуальную ветрину, в каких обьемах выполнять закупки для владельцев собственных складов. А так же управлять другими процессами.

Проанализировав сложившуюся ситуацию, сделан вывод, учитывая количество только успешных владельцев электронного бизнеса это очень широая аудитория потенциальных потребителей разрабатываемого продукта.

Так же обязательно нужно принять во внимание приведенный выше факт о том что много людей готовых вливать финансы в интернет комерцию но опасаются это делать в силу отсутствия навыков ведения интернет бизнесса. Это так же очень широкая аудитория потенциальных потребителей. Процент сбыта среди такого рода аудитории гараздо выше, так как они получают экспертную систему которая выполняет все статистические анализы расчитывает прогноз, сортирует списки товаров которые стоит закупать в будующем а какие нет. Так же обладет другими функциями с которыми можно ознакомиться в документации.

Разрабатываемая система практически полностью автоматизирует управление бизнес процессом, что гораздо снизит порог входимости новых предпринимателей на рынок интернет комерции.

Разработка подобной экспертной системы действительно актуальна, более того экспертная система с такой функциональностью востребованна на рынке, по той причине что эта экспертная система позволяет делать расчеты на основе которых повышается точность принятых решений.

**2 РАЗРАБОТКА АЛГОРИТМА РАСЧЕТА ПРОГНОЗА ПРОДАЖ (РПП)**

Сегодня уже достаточно много разработано различных технологий в области прогнозирования.

Активно применяются алгоритмы нейросетевого прогнозирования. Так же довольно часто применяются методы нечёткой логики. Большинство таких задач решается, по средствам применения методов исследования операций, к ним относятся теория игр, имитационное моделирование, трендовый а так же регрессионный анализ. При разрабоке системы РПП был выбран один из возможных алгоритмов строящих прогноз объёма продаж для товароы с сезонным характером.

Важно понимать, такой величиной как “сезон” в области прогнозировании могут быть любые временные колебания - сезонные вариации, так например, в случае когда проводится исследование товарооборота в течение одной недели, то в этом случае в качестве “сезона” выступает один рабочий день. К тому же, длительность цикла колебаний не обязательно один год, эта величина может варьироваться. После того как выявлена величина цикла колебаний, следующим строится прогноз используя для этого временной ряд в котором была выявлена величина цикла, с применением мультипликативных а так же аддитивных моделей.

Аддитивная модель имеет вид:

(#)

где:

*F* – прогнозируемое значение;

*Т* – тренд;

*S* – сезонная компонента;

*Е* – ошибка прогноза.

В формуле все значения указанны в денежных еденицах.

Мультипликативные модели применяются, когда во временных рядах наблюдается значение сезонной компоненты которое являются частью трендового значения. Такие модели имеют вид:

(#)

Аддитивную модель отличить от мультипликативной возможно ориентируясь на величину сезонной вариации. Для Аддитивной модели характерна почти постоянная сезонная вариация, в то время как для мультипликативной модели сезонная вариация возрастает или убывает, на графиках это выражается в изменении амплитуды колебания сезонного фактора, пример изображен нарисунке 1.



Рисунок 1 − Аддитивная и мультипликативные модели прогнозирования

**1.1 Определение тренда**

Существует множество различных взаимозаменяемых методов определения тренда. Наиболее часто встречаимые и рекомендуемые в различных литературных источниках это логарифмический, линейный, степенной, экспонинцеальный и полиномиальный методы. Задача - выбрать метод который дает наиболее точную оценку. Выбор метода в итоге, по результатам расчета, даст наиболее точный прогноз. Прежде чем определиться с выбором метода, был изучен и опробован каждый из выше перечисленных.  
 **1.2 Логарифмический тренд**

Логарифмического тренд имеет следующие свойства:

• когда b>0, уровни тренда растут с замедлением

• когда b<0, уровни тренда уменьшаются с замедлением.

• абсолютные изменения уровней по модулю всегда уменьшаются со временем.

• ускорения абсолютных изменений обозначаются знаком противоположности самим абсолютным изменениям, в то время как по модулю постепенно уменьшаются.

• цепные темпы изменения при t>? плавно приближаются к 100 % .

Напрашивается вывод, о том что логарифмический тренд, как и описанный ниже гиперболический тренд, отражает постепенно затухающий процесс изменений. Различие заключается в том, что затухание по гиперболе происходит довольно быстро при приближении к конечному пределу, в случае с логарифмическом трендом затухающий процесс продолжается без ограничения значительно медленнее.

Система уравнений для определения коэффициентов уравнения регрессии имеет вид:

(#)

для функции вида:

(#)

(#)

При выполнении анализа рядов динамики большую роль играет выявление сезонных колебаний. Таким колебаниям свойственны практически постоянные изменения уровней ряда по внутригодовым периодам: месяцам, кварталам. Для того что бы выявить сезонные колебания используют специальные показатели – индексы сезонности (Is).

Для ряда внутригодовой динамики, в которой основная тенденция роста

незначительна (или она не наблюдается совсем), изучение сезонности базируется на методе постоянной средней: являющейся средней из всех рассматриваемых уровней. Один из способов получить индекс сезонности состоит в следующем: для каждого года отдельно рассчитывается средний уровень, после чего выполняется сопоставление(в процентах) уровня каждого месяца. Получаемое процентное отношение является индексом сезонности:

(#)

В большинстве случаев при выполнении моделирования рядов динамики с использованием полиномов или экспонециальной функции не получают необходимых результатов, это происходит по той пречине что эти ряды динамики содержат в себе периодические колебания вокруг общей тенденции.

В таких случаях как правило приходится использовать гармонический анализ.

Цель данного анализа заключается в нахождении и измерении периодических колебаний врядах динамики. Функцию, которая была заданна в каждой точке рассматриваемого интервала времени, представляют в виде бесконечного ряда синусоидальных и косинусоидальных функций.

Гармонический анализ - это операция выражения заданной периодической функции в виде ряда Фурье по гармоникам разных порядков.

Каждый член ряда - это слагаемое постоянной величины с функциями синусов и косинусов некоторого периода.

Аппроксимация динамики финансовых явлений по средствам ряда Фурье заключается в отборе таких гармонических колебаний, которые при наложении друг на друга отражают периодические колебания динамического ряда. Используя ряды Фурье представляют динамику явлений в виде некоторой функции во времени, в которой слагаемые расставлены по убыванию периодов:

Методом наименьших квадратов рассчитывают параметры уравнений:

На графиках изображены варианты зависимостей факторного признакаY от факторного Х, где Х – временной фактор.

По данным таблицы 1, был построен график который изображен на рисунке 1.

  
Рисунок 1 - Логарифмический тренд

Выполнив расчет коэффициента детерминации был получен результат.

**1.3 Линейный тренд**

Линейный тренд представляется как линейная зависимость анализируемой величины вида

где

y – исследуемая переменная (например, производительность);

x – значение, указывающее позицию (порядковый номер) года в периоде прогнозирования.

Когда выполняется линейная аппроксимация связи двух параметров, для того что бы найти эмпирические коэффициенты линейной функции, чаще всего используют метод наименьших квадратов. Суть этого метода заключается в том, что линейная функция «наивысшего соответствия» проходит через точки графика, которые соответствуют минимуму суммы квадратов отклонений измеряемого параметра. Имеет вид:

Уравнения расчитывающее параметры линейной парной регресии имеют следующий вид:

где n – объем совокупности (исследуемой).



Рисунок X − Построение тренда методом наименьших квадратов

Константы a и b или значение коэффициента при переменной Х а также свободного члена уравнения рассчитывается по следующей формуле:

а также:

В таблице # приведен пример вычисления линейного тренда по данным.

Таблица #. Вычисление линейного тренда

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Исходные данные | | XY |  |  |
| Период времени X | Обьем продажи Y |
| 1 | 10 | 10 | 1 | 14,68 |
| 2 | 12 | 24 | 4 | 15,42 |
| 3 | 13 | 39 | 9 | 16,16 |
| 4 | 16 | 64 | 16 | 16,9 |
| 5 | 19 | 95 | 25 | 17,64 |
| 6 | 23 | 138 | 36 | 18,36 |
| 7 | 26 | 182 | 49 | 19,12 |
| 8 | 30 | 240 | 64 | 19,86 |
| 9 | 28 | 252 | 81 | 20,6 |
| 10 | 18 | 180 | 100 | 21,34 |
| 11 | 16 | 176 | 121 | 22,08 |
| 12 | 14 | 168 | 144 | 22,82 |
|  |  |  |  |  |
| Среднее 6,5 | 18,75 |  |  |  |

, ; , ; ;

= =;

= 18,75 − 0,74 \* 6,5 = 18,25 − 4,81 = 13,94

уравнение тренда:

= 13,94 + 0,74

Методы сглаживания колебаний. При сильных расхождениях между соседними значениями тренд, полученный методом регрессии, трудно поддается анализу. При прогнозировании, когда ряд содержит данные с большим разбросом колебаний соседних значений, следует их сгладить по определенным правилам, а потом искать смысл в прогнозе. К методу сглаживания колебаний

относят: метод скользящих средних (рассчитывается n-точечное среднее), метод экспоненциального сглаживания. Рассмотрим их.

Метод «скользящих средних» (МСС). МСС позволяет сгладить ряд значений с тем, чтобы выделить тренд. При использовании этого метода берется среднее (обычно среднеарифметическое) фиксированного числа значений. Например, трехточечное скользящее среднее. Берется первая тройка значений, составленная из данных за январь, февраль и март (10 + 12 + 13), и определяется среднее, равное 35 : 3 = 11,67.

Полученное значение 11,67 ставится в центре диапазона, т.е. по строке февраля. Затем «скользим на один месяц» и берется вторая тройка чисел, начиная с февраля по апрель (12 + 13 + 16), и рассчитывается среднее, равное 41 : 3 = 13,67, и таким приемом обрабатываем данные по всему ряду. Полученные средние представляют новый ряд данных для построения тренда и его аппроксимации. Чем больше берется точек для вычисления скользящей средней, тем сильнее происходит сглаживание колебаний. Пример из МВА построения тренда дан в табл. 5.2 и на рис. 5.4.

Таблица 5.2 Расчет тренда методом трехточечного скользящего среднего

|  |  |  |
| --- | --- | --- |
| Месяц | Объем продаж, тыс. ед. | Трехточечное скользящее среднее объема продаж, тыс. ед. |
| Январь | 10 | (10 + 12 + 13) / 3 = 11,67 |
| Февраль | 12 | (12 + 13 + 16) / 3 = 13,67 |
| Март | 13 | (13 + 16 + 19) / 3 = 16 |
| Апрель | 16 | (16 + 19 + 23) / 3 = 19,33 |
| Май | 19 | (19 + 23 + 26) / 3 = 22,67 |
| Июнь | 23 | (23 + 26 + 30) / 3 = 26,33 |
| Июль | 26 | (23 + 26 + 30) / 3 = 26,33 |
| Август | 30 | (26 + 30 + 28) / 3 = 28 |
| Сентябрь | 28 | (30 + 28 + 18) / 3 = 25,33 |
| Октябрь | 18 | (28 + 18 + 16) / 3 = 20,67 |
| Ноябрь | 16 | (18 + 16 + 14) / 3 = 16 |
| Декабрь | 14 |  |

Характер колебаний исходных данных и данных, полученных методом скользящего среднего, иллюстрирован на рис. 5.4. Из сравнения графиков рядов исходных значений (ряд 3) и трехточечных скользящих средних (ряд 4), видно, что колебания удается сгладить. Чем большее число точек будет вовлекаться в диапазон вычисления скользящей средней, тем нагляднее будет вырисовываться тренд (ряд 1). Но процедура укрупнения диапазона приводит к сокращению числа конечных значений и это снижает точность прогноза.  
 Прогнозы следует делать исходя из оценок линии регрессии, составленной по значениям исходных данных или скользящих средних.



Рис. 5.4. Характер изменения объема продаж по месяцам года:

исходные данные (ряд 3); скользящие средние (ряд 4); экспоненциальное сглаживание (ряд 2); тренд, построенный методом регрессии (ряд 1)

Метод экспоненциального сглаживания. Альтернативный подход к сокращению разброса значений ряда состоит в использовании метода экспоненциального сглаживания. Метод получил название «экспоненциальное сглаживание» в связи с тем, что каждое значение периодов, уходящих в прошлое, уменьшается на множитель (1 – α).

Каждое сглаженное значение рассчитывается по формуле вида:

St =aYt +(1−α)St−1,

где St – текущее сглаженное значение;

Yt – текущее значение временного ряда; St – 1 – предыдущее сглаженное значение; α – сглаживающая константа, 0 ≤ α ≤ 1.

Чем меньше значение константы α , тем менее оно чувствительно к изменениям тренда в данном временном ряду.

Посчитав коэффициент детерминации был получен результат **R(квадрат) = 0,0166;**

**1.4 Полиномиальный тренд**

Описать полиномиальный тренд

**1.5 Выбор наиболее точного расчета тренда**

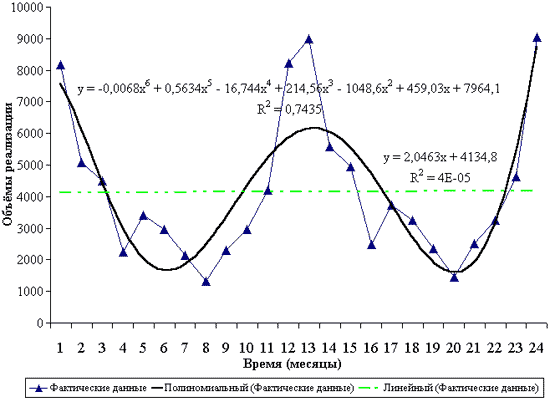
По резултатам исследования и выполненых эксперементов был выбран полиномиальный тренд. Результаты показали что данный тренд в сравнении с таким трендами как степенной а так же логарифмический или экспоненциальный оказался наиболее эффективным. Так как он наиболее удовлетворительно аппроксимирует фактические значения, и имеет самы высокий коэффицент детерминации:

· логарифмический R2 = 0,0166;

· степенной R2 =0,0197;

· экспоненциальный R2 =8Е-05.

Полниномиальный же R2 = 0,7435, что позволяет в разы уменьшить вероятность ошибки при прогнозировании.



**Рис. 2. Сравнительный анализ полиномиального и линейного тренда**

На рисунке показано, что полиномиальный тренд аппроксимирует фактические данные гораздо лучше, чем предлагаемый обычно в литературе линейный. Коэффициент детерминации полиномиального тренда гораздо выше, чем линейного.

**1.6 Алгоритм прогнозирования объема продаж по шагово**

Алгоритм прогнозирования объема продаж, имеющего сезонный характер, состоит из нескольких шагов и приведен ниже.

Первым шагом определяем тренд. Существует множество способов расчета тренда. Выше были описанны самые распространенные, с которыми был проведен эксперимант по результатам которого выбран полиномиальный тренд, так как именно он наилучшим образом позволяет сократить ошибку прогнозной модели.

Следующим шагом, вычетаем из фактических значений объёмов продаж значения тренда. Таким образом определяем величины сезонной компоненты и корректируем ее так, чтобы сумма фактических значений и сезонной компоненты была равна нулю.

После того как определены величины сезонной компоненты и откорректированны должным образом, рассчитываются ошибки модели как разности между фактическими значениями и значениями модели.

Далее строится модель прогнозирования по следующей формуле:

F = T + S ± E

где F– прогнозируемое значение, Т– тренд, S – сезонная компонента, Е - ошибка модели.

На основе модели строится окончательный прогноз объёма продаж. Для этого предлагается использовать методы экспоненциального сглаживания, что позволяет учесть возможное будущее изменение экономических тенденций, на основе которых построена трендовая модель. Сущность данной поправки заключается в том, что она нивелирует недостаток адаптивных моделей, а именно, позволяет быстро учесть наметившиеся новые экономические тенденции.

(#)

**где:**

– прогнозное значение объёма продаж;

– фактическое значение объёма продаж в предыдущем году;

– значение модели;

**–** константа сглаживания

Практическая реализация разработанного метода выявила следующие его особенности:

* для составления прогноза достаточно знать величину сезона. Исследования показывают, что множество продуктов имеют сезонный характер, величина сезона при этом может быть различной и колебаться от одной недели до десяти лет и более;
* применение полиномиального тренда вместо линейного позволяет значительно сократить ошибку модели;
* при наличии достаточного количества данных (в зависимости от величины сезона) метод даёт хорошую аппроксимацию и может быть эффективно использован при прогнозировании объема продаж в инвестиционном проектировании.

**3 ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИ РАЗРАБОТАННОГО АЛГОРИТМА В РЕАЛЬНЫХ УСЛОВИЯХ**

Для того проверить на сколько точный прогноз выдает данный алгоритм, был проведен эксперимент. Для построения прогноза, использовались данные об объёмах продаж мороженого “Белый пломбир” полученные из отчетов одного из предприятий которое находится в Нижнем Новгороде за 2013 и 2014 годы. Для данной статистики характерно то, что у значений объёма продаж явно наблюдается сезонный характер и возрастающий тренд. Задача заключалась в построении прогноза по продажам на 2015г. На момент написания диссертации, 2016 год, то есть нам доступны уже данные за прошедший 2015г. Был построен прогноз на 2015г. по результатам продажи за 2013 и 2014 годы, после чего выполнено сравнение с реальными цифрами продаж за 2015г. Исходная информация представлена в табл. 1.

**Таблица 1.**

**Фактические объёмы реализации продукции**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №п.п. | Месяц | Объем продаж (руб.) | №п.п. | Месяц | Объем продаж (руб.) |
| 1 | июль | 8174,40 | 13 | июль | 8991,84 |
| 2 | август | 5078,33 | 14 | август | 5586,16 |
| 3 | сентябрь | 4507,20 | 15 | сентябрь | 4957,92 |
| 4 | октябрь | 2257,19 | 16 | октябрь | 2482,91 |
| 5 | ноябрь | 3400,69 | 17 | ноябрь | 3740,76 |
| 6 | декабрь | 2968,71 | 18 | декабрь | 3265,58 |
| 7 | январь | 2147,14 | 19 | январь | 2361,85 |
| 8 | февраль | 1325,56 | 20 | февраль | 1458,12 |
| 9 | март | 2290,95 | 21 | март | 2520,05 |
| 10 | апрель | 2953,34 | 22 | апрель | 3248,67 |
| 11 | май | 4216,28 | 23 | май | 4637,91 |
| 12 | июнь | 8227,569 | 24 | июнь | 9050,3264 |

Была поставлена задача, построить прогноз по продажам товара на следующий год для каждого месяца.

Реализуем алгоритм расчета прогноза, который был описан выше. Для проведения эксперимента, и испытания алгоритма на практике была выбрана среда MS Excel, которая берет на себя выполнение рутинной работы что позволяет сосредоточиться именно на отладке алгоритма.

Определяем тренд, был выбран полиномиальный тренд, результаты выше описанных эксперементов показали что именно этот тренд выполняет апроксимацию данных значительно эффективнее, чем любые другие описываемые в большинстве литературных источниках. Его коэффициент детерминации составляет (0,74349) который гораздо выше, в отличие от линейного (4E-05) или любого другого тренда.

Далее определяется величина сезонной компоненты для этого, вычтим значения тренда из фактических значений объёмов продаж, в результате получим величины сезонной компоненты, пример результатов расчета изображен на рис.4.

**Таблица 2.**

**Расчёт значений сезонной компоненты**

|  |  |  |  |
| --- | --- | --- | --- |
| **Месяцы** | **Объём продаж** | **Значение тренда** | **Сезонная компонента** |
| 1 | 8174,4 | 7617,2674 | 557,1326 |
| 2 | 5078,3296 | 6104,0156 | -1025,686 |
| 3 | 4507,2061 | 4420,3206 | 86,885473 |
| 4 | 2257,1992 | 3004,1224 | -746,92323 |
| 5 | 3400,6974 | 2086,745 | 1313,95235 |
| 6 | 2968,7178 | 1741,0644 | 1227,65338 |
| 7 | 2147,1426 | 1924,9246 | 222,217979 |
| 8 | 1325,5674 | 2519,8016 | -1194,2342 |
| 9 | 2290,9561 | 3364,7154 | -1073,7593 |
| 10 | 2953,3411 | 4285,39 | -1332,0489 |
| 11 | 4216,2848 | 5118,6614 | -902,37664 |
| 12 | 8227,5695 | 5732,1336 | 2495,43589 |
| 1 | 8991,84 | 7617,2674 | 1374,5726 |
| 2 | 5586,1626 | 6104,0156 | -517,85304 |
| 3 | 4957,9267 | 4420,3206 | 537,60608 |
| 4 | 2482,9191 | 3004,1224 | -521,20332 |
| 5 | 3740,7671 | 2086,745 | 1654,02209 |
| 6 | 3265,5896 | 1741,0644 | 1524,52515 |
| 7 | 2361,8568 | 1924,9246 | 436,932237 |
| 8 | 1458,1241 | 2519,8016 | -1061,6775 |
| 9 | 2520,0517 | 3364,7154 | -844,6637 |
| 10 | 3248,6752 | 4285,39 | -1036,7148 |
| 11 | 4637,9132 | 5118,6614 | -480,74817 |
| 12 | 9050,3264 | 5732,1336 | 3318,19284 |

Откорректируем сезонную компоненту так, чтобы их сумма ровнялась нулю.

**Таблица 3.**

**Расчёт средних значений сезонной компоненты**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Месяцы** | **1-й сезон** | **2-й сезон** | **Итого** | **Среднее** | **Сезонная компонента** |
| 1 | 557,1326 | 1374,5726 | 1931,7052 | 965,8526 | 798,7176058 |
| 2 | -1025,686 | -517,853035 | -1543,539 | -771,7695155 | -938,90451 |
| 3 | 86,885473 | 537,60608 | 624,491553 | 312,2457765 | 145,1107823 |
| 4 | -746,92323 | -521,203316 | -1268,1265 | -634,0632745 | -801,198269 |
| 5 | 1313,9524 | 1654,022089 | 2967,97444 | 1483,987221 | 1316,852227 |
| 6 | 1227,6534 | 1524,525154 | 2752,17853 | 1376,089265 | 1208,954271 |
| 7 | 222,21798 | 436,932237 | 659,150216 | 329,575108 | 162,4401138 |
| 8 | -1194,2342 | -1061,677479 | -2255,9117 | -1127,955849 | -1295,09084 |
| 9 | -1073,7593 | -844,663701 | -1918,423 | -959,2115055 | -1126,3465 |
| 10 | -1332,0489 | -1036,714798 | -2368,7637 | -1184,381853 | -1351,51685 |
| 11 | -902,37664 | -480,748169 | -1383,1248 | -691,5624065 | -858,697401 |
| 12 | 2495,4359 | 3318,192838 | 5813,62873 | 2906,814363 | 2739,679369 |
|  |  |  | **Сумма** | **2005,61993** | **0** |

**3. Рассчитываем ошибки модели** как разности между фактическими значениями и значениями модели.

**Таблица 4.**

**Расчёт ошибок**

|  |  |  |  |
| --- | --- | --- | --- |
| **Месяц** | **Объём продаж** | **Значение модели** | **Отклонения** |
| 1 | 8174,4 | 8415,985006 | -241,585006 |
| 2 | 5078,3296 | 5165,11109 | -86,7814863 |
| 3 | 4507,2061 | 4565,431382 | -58,2253093 |
| 4 | 2257,1992 | 2202,924131 | 54,27503571 |
| 5 | 3400,6974 | 3403,597227 | -2,89987379 |
| 6 | 2968,7178 | 2950,018671 | 18,69910521 |
| 7 | 2147,1426 | 2087,364714 | 59,77786521 |
| 8 | 1325,5674 | 1224,710757 | 100,8566247 |
| 9 | 2290,9561 | 2238,3689 | 52,58718971 |
| 10 | 2953,3411 | 2933,873153 | 19,46793921 |
| 11 | 4216,2848 | 4259,963999 | -43,6792433 |
| 12 | 8227,5695 | 8471,812969 | -244,24348 |
| 13 | 8991,84 | 8415,985006 | 575,8549942 |
| 14 | 5586,1626 | 5165,11109 | 421,0514747 |
| 15 | 4957,9267 | 4565,431382 | 392,4952977 |
| 16 | 2482,9191 | 2202,924131 | 279,9949527 |
| 17 | 3740,7671 | 3403,597227 | 337,1698622 |
| 18 | 3265,5896 | 2950,018671 | 315,5708832 |
| 19 | 2361,8568 | 2087,364714 | 274,4921232 |
| 20 | 1458,1241 | 1224,710757 | 233,4133637 |
| 21 | 2520,0517 | 2238,3689 | 281,6827987 |
| 22 | 3248,6752 | 2933,873153 | 314,8020492 |
| 23 | 4637,9132 | 4259,963999 | 377,9492317 |
| 24 | 9050,3264 | 8471,812969 | 578,5134687 |

Находим среднеквадратическую ошибку модели (Е) по формуле:

**где:**

***Т*** − трендовое значение объёма продаж;

***S***– сезонная компонента;

***О −*** отклонения модели от фактических значений

***Е =*** *0,003739 или 0.37 %*

По результатам полученным при расчете среднеквадратичной ошибки можно сделать вывод, что построенная модель выполняет эффективную аппроксимацию фактических данных, то есть она действительно отображает экономические тенденции, позволяющие определить объём продаж, что свидетельствует о способности построить высокоточный прогноз.

Теперь строится модель прогнозирования :

***F = T + S ± E***

На рисунке 1 изображена построенная модель.

**5.** Используя построенную модель расчитывается непосредственно сам прогноз объёмов продаж. Для уменьшения влияния прошедших тенденций на точность результатов, помимо трендового анализа применяется еще и экспоненциальное сглаживание. Это позволит не упустить новые тенденции:

***где:***

− прогнозное значение объёма продаж;

– фактическое значение объёма продаж в предыдущем году;

− значение модели;

***а*** – константа сглаживания.

Для определения константы сглаживания был выбран методод экспертных оценок, как вероятность сохранения неизменной экономической ситуации, то есть в случае когда главные характеристики колеблются с той же амплитудой что и раньше, значит нет фактов говорящих о грядущих изменениях сложившейся экономической ситуации.



**Рис. 5. Модель прогноза объёма продаж**

Следующим шагом был построен прогноз, ниже описан пример построения за январь третьего сезона.

Расчет прогнозного значения модели:

Фактические продажи за предыдущий год (F\_ф t-1) состовляют 2 361руб. Коэффициент сглаживания равняется 0.8. Приняв все значения был построен прогноз:

Так же следует отметить что тенденции в экономике постоянно меняются, для того что бы учитывать эти изменения необходимо выполнять построение прогноза учитывая наиболее свежие данные о фактичеких обьемах продаж за прошедшие периоды. Результаты проведенных исследований показали что максимальный срок актуальности данных это три прошедших периода. Например если в качестве периода берется год, то самые давни данные которые можно использовать, это данные трех летней давности. Любые значения полученные еще раньше чем за три предыдущих периода теряют свою актуальность.

**2.1 Точность прогнозов**

Результаты сравнения подвердили высокую точность прогноза, результаты которого имели лиш небольшое отклонение от действительности. В таблици 2 Вы можете увидеть сравнение результатов прогноза с реальными цифрами результатов продаж.

|  |  |  |
| --- | --- | --- |
| **Месяц** | **Результаты прогноза** | **Реальные продажи** |
| июль |  |  |
| август |  |  |
| сентябрь |  |  |
| октябрь |  |  |
| ноябрь |  |  |
| декабрь |  |  |
| январь |  |  |
| февраль |  |  |
| март |  |  |
| апрель |  |  |
| май |  |  |
| июнь |  |  |

Справедливо заметить, что данные отклонения савершенно не значитильне для ведения продаж, так как он не ведет к убыткам, на основании этого можно сделать вывод что алгоритм действилеьно работает, и имеет полезность. Проанализировать продажи за прошедший определенный период и правильно принять решение о закупках на следующий период может только очень опытный предприниматель, который получил свой опыт "набивая шишки на голове", понеся убытки не однократные убытки за годы своей деятельности. Зачастую для начинающи предпренимателей это очень дорогие ошибки, которые неоднократно приводили к закрытию перспективного малого или крупного бизнесса, только из-за неправильных расчетов на сегон. Так же уменьшаются риски в случае если человек в качестве стартового капитала использует кредит в банке, правильно расчитав продажи на сезон предприниматель получит более высокую прибыль что поможет погасть кредит в срок. На основе этого можно сделать вывод что полезность алгоритма высока, и использование именно его в данной системе оправдано.

**4 ВЫБОР КОМПОНЕНТОВ СЕРВЕРА ДЛЯ РАБОТЫ С РАЗРАБОТАННОЙ ПРОГРАММОЙ**

Для применения алгоритма в разрабатываемой системе, он был запрограммирован, с использованием языка PHP. Алгоритм выполняется на удоленном сервере, взаимодействие с которым происходит по средствам API. В зависимости от задачи на удаленном сервере должно быть установлено то или иное ПО. В ходе работы были изучены различные пакеты программного обеспечения, среди которых был выбран позволяющий реализовать соответствующий требованиям сервер. Для того что бы было возможно решить поставленную задачу, и разработанная система могла развиваться а также расширять свой функционал, не требуя при этом много ресурсов, таких как время, финансы, с которых оплачивется труд разработчиков и т.д, удаленный сервер должен соостветствовать следующим требованиям. Возможность быстрого перехода с более слабых на более мощные сервера, легкость переноса всей серверной части, легкость адаптации в случае необходимости, рациональные и оправдывающие себя запроссы в финансовой поддержке и т.д.

**4.1 Компоненты сервера**

Выполнив исследования, различного серверного программного обеспечения, мною было принято решение в пользу FAMP, так как именно он идеально соответствовал требованиям. Далее будет подробно описанно что такое FAMP, и сравнение его каждой отдельно взятой компоненты с аналогами, где так же будут приведены более подробные пояснения, что именно побудило сделать выбор в пользу FAMP.

FAMP — комплекс серверного программного обеспечения. FAMP - это абривеатура составленная по первым буквам названий входящих в этот комплекс компонентов:

· FreeBSD — UNIX-подобная операционная система;

· Apache — веб-сервер;

· MySQL — система управления базой данных;

· PHP — язык программирования, ориентированный на веб разработку для серверной части.

Многим больше известен альтернативный набор LAMP, это тот же самый набор в котором вместо FreeBSD, используется операционная система Linux. Справедливо отметить что именно LAMP задал тон именования подобных наборов серверного ПО.

Все програмные продукты воходяшие в состав подобных комплексов разрабатывались отдельно как независимые проекты, некоторые из них разрабатывались разными компаниями. Такой комплекс стал популярным в ходе развития такой области как веб-разработка. Популярным он стал из-за своей гибкости а так же высокой производительности, так же на ряду с этим все компоненты распростроняются под открытой лицензией что приводит к снижению финансовых затрат для проектов использующих его. На сегоднешний день набор FAMP предостовляется многими хостинг-площадками.

**4.1.1 Apache**

HTTP-сервер Apache, Кроссплатформенное ПО, которое поддерживается практически всеми существующими операционными системами.

К главными преимуществами Apache относятся его надёжность и гибкость при конфигурировании. Все эти характеристики очень важны для разрабатывемой системы, кросс платформинность, обеспечивает гибкость в

проектировании сервера. Гибкость данного ПО заключается в том что если по каким либо причинам необходимо будет поменять операционную систему на сервере, например при необходимости использовать ПО под Linux сисемы а не под FreeBSD, которая была выбрана на момент написания работы, то у нас не возникнет проблем с переходом на другую ОС. В таблице 2.1 приведен список операционнх систем под которые адаптирован Apache.

**Таблица 2.1 ОС под которые адаптирован Apache.**

|  |  |  |
| --- | --- | --- |
| **№** | **Название** | **Поддержка** |
| 1 | OC на базе Linux | полная |
| 2 | ОС линии BSD | полная |
| 3 | Mac OS | полная |
| 4 | Microsoft Windows | полная |
| 5 | Novell NetWare | полная |
| 6 | BeOS | полная |

Так же часто бывает, когда проект растет, он от части меняет свою специфику, или же расширяет ее, что приводит к появлению ряда задач которые проще и быстрее и надежней решать используя другой язык программирования, благодоря поддержки не малого колличества языков программирования, мы можем разработать отдельные части или же полностью переписать систему на другом языке программирования, не затронув других настроек сервера. В таблице 2.2 приведен список подрживаемых языков.

**Таблица 2.2 Поддерживаемые языки Apache.**

|  |  |  |
| --- | --- | --- |
| № | Название | Поддержка |
| 1 | [PHP](https://ru.wikipedia.org/wiki/PHP) | Полная |
| 2 | Python | Полная |
| 3 | Ruby | Полная |
| 4 | Perl | Полная |
| 5 | ASP | Полная |
| 6 | TCL | Полная |

Apache предостовляет возможность подключения сторонних модулей, выполнять аутентификацию пользователей используя СУБД, гибкость работы с сообщениями об ошибках а так же множество других дополнительных опций. Включает поддержку IPv6.

Основное что реализовано в ядре Apache это: работа с файлами конфигураций, HTTP(протокол), интерфейс для интеграции модулей которые были разработаны сторонними разработчиками. Разработкой ядра и на сегоднешний день его развитием занимается компания Apache Software Foundation без превличения сторонних разработчиков, которые допускаются только к разработке модулей. Для написания ядра использовался язык Си.

Система конфигурации Apache основывается на фаилах в текстовом формате которые еще называются конфигурационными фаилами. Состоит из трех условных уровней конфигурации:

Конфигурация сервера (httpd.conf).

Конфигурация виртуального хоста (httpd.conf c версии 2.2, extra/httpd-vhosts.conf).

Конфигурация уровня директории (.htaccess).

Для управления конфигурационными фаилами использует собственный язык, который основывается на блоках директив. По средствам фаилов конфигураций можно изменить практически любые параметры ядра. Практически у всех модулей собственные параметры.

Так же есть модули которые в своей работе конфигурационные фаилы операционной системы такие как etc/pass или etc/host.

Система модулей. Apache HTTP Server инегрировать сторонние модули. На сегоднешний день разработанно более 500 модулей, предостовляющих различную функциональность. Некоторые из этих модулей были разработанны непосредственно самой компанией Apache Software Foundation. Но большая часть была разработанна сторонними open source-разработчиками.

Процесс интеграции модулей довольно гибкий, их можно интегрировать как в момент компиляции так и загрузить динмически по средствам конфигурационного фаила используя диррективы.

Создание новых модулей позволяет реализовать такие решения как:

· Поддержка языков программирования.

· Расширение функциональности.

· Доработка включенных в ядро функций.

· Работа с безопасностью.

Подводя итог по выше сказанному, возможность подключать модули, наши возможности становятся практически не ограниченными, вплоть до адаптации его до собственного языка программирования.

В качестве примера, часть веб-приложений, например панели управления ISPmanager и VDSmanager реализованы в виде модуля Apache.

Виртуальные хосты, в ядре Apache реализована технология виртуальных хостов. Такого рода технология дает возможность поддерживать на одном IP-адресе несколько разных доменных имён, выводя содержимое индивидуально для каждого. Так же он позволяет задавать индивидуально настройки ядра, модулей а так же доступов для каждого отдельного виртуального хоста.

Некоторые MPM, например Apache-ITK позволяют запускать процесс httpd для каждого виртуального хоста с отдельными идентификаторами uid и guid.

Также, есть модули, предостовляющие возможноть упровлять ресурсами сервера, такими как трафик, оперативная память, настройки процессора, для каждого из виртуальных хостов в отдельности.

Безопасность, В ядре Apache реализованы функции для обеспечения безопасности и управление доступом к данным. Среди основных можно отметить:

Настройка доступа для дирректорий и фаилов.

Авторизация пользователей для получения доступов используя HTTP-аутентификации (mod\_auth\_basic) и digest-аутентификации (mod\_auth\_digest).

Настрой доступов к непосредственно самому серверу, ориентируясь на IP-адреса.

Есть модули, которые выполняют авторизацию используя системы управления базой данных или подключаемые модули аутентификации.

Также, в Apache присутствует suexec, это механизм выполняющий запуск скриптов а так же CGI-приложений с данными об идентификационными и правами пользователя.

Для шифрования данных, которые передаются между клиентом и сервером используется алгоритм SSL, который был написан с использованием библиотеки OpenSSL. Для удостоверения что веб-сервер подлинный применяют сертификаты X.509.

Присутствуют внешние средства для работы с безопасностью, к таким относится mod\_security.

Интернационализация, в версии 2.0 была добавлена функциональность позволяющая определить локаль клиента на стороне сервера. С этого момента посылаемые браузеру системные сообщения, такие как сообщения об ошибках, могут быть отправленны на разных языках.

Еще один важный момент который стоит отметить, то что появилась возможность управлять на стороне сервера выводом содержимого страниц в зависимости от локали пользователя. Apache в состоянии работать с большим количеством кодировок, в том числе Юникод, это дает возможность обробатывать страницы, сохраненных в любых кодировках и написанных на любых языках.

Обработка событий, администратору предоставленна возможность самому задать собственные страницы и обработчики для возникающих HTTP ошибок или событий, например 404 (страница не найдена) или 403 (запрещен доступ). В том числе возможность запуска скриптов и отображения сообщений на разных языках.

Server Side Includes, начиная с версии 1.3 и выше добавлен механизм Server Side Includes, который дает возможность динамически работать с HTML-документами на серверной стороне.

За управление SSI отвечает модуль mod\_include, который входит в ядро Apache.

**4.1.2 База данных MySQL**

MySQL − свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации.

MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertrigoServ. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.

26 февраля 2008 года Sun Microsystems приобрела MySQL AB за 1 млрд долларов, 27 января 2010 года Oracle приобрела Sun Microsystems за 7,4 млрд долларов и включила MySQL в свою линейку СУБД.

Сообществом разработчиков MySQL созданы различные ответвления кода, такие как Drizzle, OurDelta, Percona Server и MariaDB. Все эти ответвления уже существовали на момент поглощения компании Sun корпорацией Oracle.

Языки программирования. MySQL имеет API для языков Delphi, C, C++, Эйфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk, Компонентный Паскаль и Tcl, библиотеки для языков платформы .NET, а также обеспечивает поддержку для ODBC посредством ODBC-драйвера MyODBC.

MyODBC представляет собой драйвер ODBC (2.50) уровня 0 (с некоторыми возможностями уровней 1 и 2) для подсоединения совместимого с ODBC приложения к **MySQL**. MyODBC работает на всех системах Microsoft Windows и на большинстве платформ Unix.

#### Тип таблиц Maria. Maria (начиная с версии 5.2.x — Aria) — расширенная версия хранилища MyISAM, с добавлением средств сохранения целостности данных после краха.

Основные достоинства Maria.

* В случае краха производится откат результатов выполнения текущей операции или возврат в состояние до команды LOCK TABLES. Реализация через ведение лога операций.
* Возможность восстановления состояния из любой точки в журнале операций, включая поддержку CREATE/DROP/RENAME/TRUNCATE. Может быть использовано для создания инкрементальных резервных копий, через периодическое копирование журнала операций.
* Поддержка всех форматов столбцов MyISAM, расширена новым форматом «rows-in-block», использующим страничный способ хранения данных, при котором данные в столбцах могут кэшироваться.
* В будущем будет реализовано два режима: транзакционный и без отражения в журнале транзакций, для некритичных данных.
* Размер страницы данных равен 8Кб (в MyISAM 1Кб), что позволяет достичь более высокой производительности для индексов по полям фиксированного размера, но медленнее в случае индексирования ключей переменной длины.

**4.1.3 Операционная система FreeBSD и почему именно она**

Если в вопросе Linux против Windows все преимущества и недостатки сравниваемых ОС более-менее ясны, то в вопросе «FreeBSD против Linux» все не так просто.

Linux — не операционная система, важно понимать различия между операционной системой и ядром операционной системы. Так вот, FreeBSD — это операционная система. Linux — это ядро операционной системы. Ubuntu, Gentoo, Arch Linux и SuSe — это дистрибутивы Linux, то есть операционные системы (плюс инсталятор, менеджер пакетов и тд), основанные на Linux. Всего существует несколько сотен дистрибутивов Linux и все они довольно сильно между собой различаются.

Допустим, мы установили Ubuntu и какое-то время пользовались этим дистрибутивом. Пользовались-пользовались, и в какой-то момент решили, что нам что-то не вполне в нем нравится. Что делать? Переходить на другую операционную систему, например Gentoo. И изучать, как там под ней устанавливается софт, где хранятся конфигурационные файлы и тд.

В отличие от Linux FreeBSD не ограничивает наш выбор ни в чем.

Ни в способе установки программ, ни в используемом оконном менеджере или фаерволе, ни в настройках ядра и способе его обновления. Потому что это однаоперационная система и никто не пытается заточить ее только под определенный круг пользователей. Хотим — устанавливаем софт из бинарных пакетов (пакеджей), хотим — компилируем из исходников (портов). Хотим оконный менеджер — выбираем любой от KDE до Awesome. Не нравится фаервол по умолчанию (ipfw), ставим pf. Нужен сервер — получаем сервер, нужна рабочая станция — получаем рабочую станцию. Насколько мне известно, ни один дистрибутив Linux не обладает такой гибкостью и не предоставляет такой свободы выбора, как FreeBSD.

Есть и другие проблемы, связанные с тем, что разработкой ядра Linux и операционных систем на его основе занимаются разные люди. Но на мой взгляд они менее серьезны, чем описанная выше.

FreeBSD намного безопаснее Linux,Была изучена статистика, из различных источников, по данным которых в опирационной системе FreeBSD в среднем 4-5 уязвимостей, справедливо отметить что есть источник который утверждает что за год было найдено 10 уязвимостей, но это не меняет картины. Для сравнения, в ядре за тот же период было найдено 123 уязвимости.

Выходит, что каждую неделю в ядре Linux находят по две ошибки. А если прибавить уязвимости в GNU и различных дистрибутивах? Ниже приведена гистограмма, отображающая количество ошибок, найденных во FreeBSD, Microsoft Windows и ядре Linux за последние 6 лет.



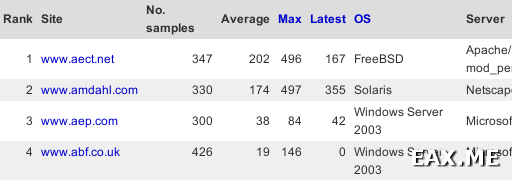
В одном только *ядре* Linux находят больше ошибок, чем в целых операционных системах.

Но хуже всего то, что количество уязвимостей, которые находят в Linux, год от года стабильно растет.

Можно было бы говорить о том, что в Linux находят больше ошибок, потому что им пользуется больше людей, чем FreeBSD. если бы ошибок было не в 12 раз больше, а, хотя бы, только в два. Я почти уверен, что именно благодаря своей безопасности, FreeBSD пользуется большой популярностью у таких крупных хостинг-провайдеров, как Masterhost, Majordomo и RU-CENTER.

Кстати, если внимательно посмотреть, *какие* ошибки находят во FreeBSD, несложно заметить, что, как правило, они могут использоваться злоумышленником только в *очень* редких случаях. Взять к примеру довольно свежую уязвимость в nfsclient.

Не удивительно, что именно серверы под управлением FreeBSD славятся большими значениями uptime. Железо, на котором работает ОС, выходит из строя быстрее, чем появляется необходимость обновиться и презагрузить машину. Высокий аптайм серверов на FreeBSD подтверждает (на момент написания этих строк) NetCraft:



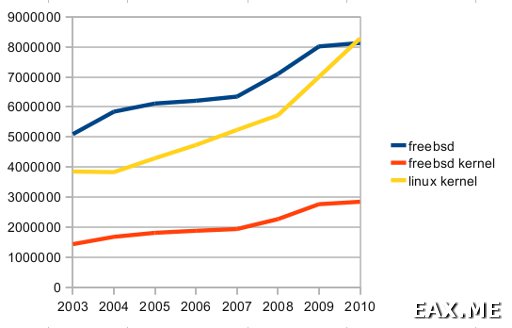
Размер имеет значение, сложно спорить с цифрами. Так что я решил провести еще одно маленькое исследование, на этот раз связанное с количеством строк в коде FreeBSD и Linux. Вы спросите, какое отношение имеет количество строк кода к качеству операционной системы? Прямое. Любому программисту известно, что количество ошибок в программе тем больше, чем больше ее код. Вполне очевидно, что при написании маленького калькулятора ошибиться намного сложнее, чем при разработке серьезного бухгалтерского приложения. Для кого это не очевидно, тот без труда найдет в сети множество исследований на соответствующую тему. Так давайте же посчитаем количество строк в коде FreeBSD и Linux!

Считать будем с помощью утилиты CLOC («Count Lines Of Code», в портах FreeBSD — /usr/ports/misc/cloc). Она не учитывает комментарии и пустые строки в коде, а также умеет отфильтровывать копии файлов. Считать количество *всех*строк в исходном коде не интересно, потому что комментарии, пустые строки и копии файлов, очевидно, никак не влияют на количество ошибок в программе.

Исходники Linux лежат в свободном доступе на kernel.org, исходники FreeBSD можно скачать из SVN репозитория.

Анализировались только *релизы* FreeBSD и Linux ветки 2.6, вышедшие в период с 2003 по 2010 год. Какой релиз в каком году вышел, можно прочитать на офсайтах проектов.

В результате получилась такая зависимость количества строк кода от времени:



Здесь «freebsd» — это *операционная система* FreeBSD (все исходники), а «freebsd kernel» — это только *ядро*. Следует отметить, что ядро FreeBSD собирается совершенно независимо от остальных частей системы.

Сравнивать, разумеется, следует ядро с ядром, а не ядро с системой. Внимательно посмотрим на полученную картинку, можно увидеть следующее. Во-первых, ядро Linux значительно больше ядра FreeBSD. Исходный код Linux версии 2.6.33 даже больше кода *операционной системы* FreeBSD! Во-вторых, размер ядра Linux увеличивается намного быстрее ядра FreeBSD — 4 млн новых строк кода за 7 лет против 1.5 млн. Кроме того, ядро Linux, похоже, растет экспоненциально, в то время, как ядро FreeBSD — линейно.

Итак, ядро Linux значительно больше ядра FreeBSD, и, что намного хуже, оно растет значительно быстрее.

Следовательно, Linux менее надежен и менее безопасен, чем FreeBSD. Это подтверждает и предыдущий пункт заметки. Тот, который про безопасность.

Недостатки, основных недостатков FreeBSD перед Linux, на мой взгляд, три.

Во-первых, Linux поддерживает больше различных экзотических устройств, которые в особенности любят производители ноутбуков.

Во-вторых, некоторые программы, например Skype, распространяются только в виде бинарников для Windows, Mac OS и нескольких дистрибутивов Linux. Конечно, это не вина разработчиков FreeBSD, но специально на этот случай они предусмотрели в своей ОС возможность запускать (именно запускать, а не эмулировать!) программы, скомпилированные под Linux.

Порезультатам изученных недостатков, можно сделать следующие выводы, во первых их совсем мало, а во вторых эти не достатки не остались без решения. Так же очень важно отметить, что с этими недостатками невозможно столкнуться при использовании операционной системы FreeBSD в качестве серверной ОС, а это значит, при разработке описываемой системы нужно принимать во внимание эти недостатки.

**4.1.4 Язык программирования PHP**

Главным фактором языка РНР является практичность. РНР должен предоставить программисту средства для быстрого и эффективного решения поставленных задач. Практический характер РНР обусловлен пятью важными характеристиками:

* традиционностью;
* простотой;
* эффективностью;
* безопасностью;
* гибкостью.

Существует еще одна «характеристика», которая делает РНР особенно привлекательным: он распространяется бесплатно! Причем, с открытыми исходными кодами ( Open Source ).

Традиционность. Язык РНР будет казаться знакомым программистам, работающим в разных областях. Многие конструкции языка позаимствованы из Си, Perl.

Код РНР очень похож на тот, который встречается в типичных программах на С или Pascal. Это заметно снижает начальные усилия при изучении РНР. PHP — язык, сочетающий достоинства Perl и Си и специально нацеленный на работу в Интернете, язык с универсальным (правда, за некоторыми оговорками) и ясным синтаксисом.

И хотя PHP является довольно молодым языком, он обрел такую популярность среди web-программистов, что на данный момент является чуть ли не самым популярным языком для создания web-приложений (скриптов).

Простота, сценарий РНР может состоять из 10 000 строк или из одной строки — все зависит от специфики вашей задачи. Вам не придется подгружать библиотеки, указывать специальные параметры компиляции или что-нибудь в этом роде. Механизм РНР просто начинает выполнять код после первой экранирующей последовательности (). Если код имеет правильный синтаксис, он исполняется в точности так, как указал программист.

PHP — язык, который может быть встроен непосредственно в html -код страниц, которые, в свою очередь будут корректно обрабатываться PHP -интерпретатором. Мы можем использовать PHP для написания CGI-сценариев и избавиться от множества неудобных операторов вывода текста. Мы можем привлекать PHP для формирования HTML-документов, избавившись от множества вызовов внешних сценариев.

Большое разнообразие функций PHP избавят вас от написания многострочных пользовательских функций на C или Pascal .

Эффективность, является исключительно важным фактором при программировании для многопользовательских сред, к числу которых относится и web .

Очень важное преимущество PHP заключается в его «движке». «Движок» PHP не является ни компилятором, ни интерпретатором. Он является транслирующим интерпретатором. Такое устройство «движка» PHP позволяет обрабатывать сценарии с достаточно высокой скоростью.

По некоторым оценкам, большинство PHP-сценариев (особенно не очень больших размеров) обрабатываются быстрее аналогичных им программ, написанных на Perl. Однако, чтобы не делали разработчики PHP, откомпилированные исполняемые файлы будут работать значительно быстрее – в десятки, а иногда и в сотни раз. Но производительность PHP вполне достаточна для создания вполне серьезных web-приложений. Подробно об устройстве и характеристиках «движка» PHP можно ознакомиться здесь.

Безопасность, РНР предоставляет в распоряжение разработчиков и администраторов гибкие и эффективные средства безопасности, которые условно делятся на две категории: средства системного уровня и средства уровня приложения.

Средства безопасности системного уровня. В РНР реализованы механизмы безопасности, находящиеся под управлением администраторов; при правильной настройке РНР это обеспечивает максимальную свободу действий и безопасность. РНР может работать в так называемом безопасном режиме (safe mode), который ограничивает возможности применения РНР пользователями по ряду важных показателей. Например, можно ограничить максимальное время выполнения и использование памяти (неконтролируемый расход памяти отрицательно влияет на быстродействие сервера). По аналогии с cgi-bin администратор также может устанавливать ограничения на каталоги, в которых пользователь может просматривать и исполнять сценарии РНР, а также использовать сценарии РНР для просмотра конфиденциальной информации на сервере (например, файла passwd).

Средства безопасности уровня приложения. В стандартный набор функций РНР входит ряд надежных механизмов шифрования. РНР также совместим с многими приложениями независимых фирм, что позволяет легко интегрировать его с защищенными технологиями электронной коммерции (e-commerce). Другое преимущество заключается в том, что исходный текст сценариев РНР нельзя просмотреть в браузере, поскольку сценарий компилируется до его отправки по запросу пользователя. Реализация РНР на стороне сервера предотвращает похищение нетривиальных сценариев пользователями, знаний которых хватает хотя бы для выполнения команды View Source.

Гибкость, поскольку РНР является встраиваемым (embedded) языком, он отличается исключительной гибкостью по отношению к потребностям разработчика. Хотя РНР обычно рекомендуется использовать в сочетании с HTML, он с таким же успехом интегрируется и в JavaScript, WML, XML и другие языки. Кроме того, хорошо структурированные приложения РНР легко расширяются по мере необходимости (впрочем, это относится ко всем основным языкам программирования).

Нет проблем и с зависимостью от браузеров, поскольку перед отправкой клиенту сценарии РНР полностью компилируются на стороне сервера. В сущности, сценарии РНР могут передаваться любым устройствам с браузерами, включая сотовые телефоны, электронные записные книжки, пейджеры и портативные компьютеры, не говоря уже о традиционных ПК. Программисты, занимающиеся вспомогательными утилитами, могут запускать РНР в режиме командной строки.

Поскольку РНР не содержит кода, ориентированного на конкретный web-сервер, пользователи не ограничиваются определенными серверами (возможно, незнакомыми для них). Apache, Microsoft IIS, Netscape Enterprise Server, Stronghold и Zeus — РНР работает на всех перечисленных серверах. Поскольку эти серверы работают на разных платформах, РНР в целом является платформенно-независимым языком и существует на таких платформах, как UNIX, Solaris, FreeBSD и Windows 95/98/NT/2000/XP/2003.

Наконец, средства РНР позволяют программисту работать с внешними компонентами, такими как Enterprise Java Beans или СОМ-объекты Win32. Благодаря этим новым возможностям РНР занимает достойное место среди современных технологий и обеспечивает масштабирование проектов до необходимых пределов.

Бесплатное распространение, стратегия Open Source, и распространение исходных текстов программ в массах, оказало несомненно благотворное влияние на многие проекты, в первую очередь — Linux, хотя и успех проекта Apache сильно подкрепил позиции сторонников Open Source. Сказанное относится и к истории создания РНР, поскольку поддержка пользователей со всего мира оказалась очень важным фактором в развитии проекта РНР.

Принятие стратегии Open Source и бесплатное распространение исходных текстов РНР оказало неоценимую услугу пользователям. Вдобавок, отзывчивое сообщество пользователей РНР является своего рода «коллективной службой поддержки», и в популярных электронных конференциях можно найти ответы даже на самые сложные вопросы.

**5 РАЗРАБОТКА ПРОГРАМНОГО ОБЕСПЕЧЕНИЕ**

После того как с алгоритмом были проведены все эксперименты, и отпали все сомнения в достоверности его прогонозов, а так же полностью определились с конфигурациями серверного оборудования, которое обеспечивает необходимый уровень защиты, производительности и рационален в фнансовых запроссах необходимых для его поддержки, началась разработка программного обеспечения.

**5.1 Расчет прогноза**

Для расчета прогноза продаж, необходимо выполнить несколько довольно сложных этапов каждый из которых представляет собой вычислительные операции. Главный метод в Класск "PrognosManager", это calcPrognose(), который и обьеденяет в себе все этапы расчета, и по результатам выполнения возвращает массив с прогнозами. На рисунке 3.1 изображена блок схема работы метода calcPrognose().

нет

да

data[] := loadFromDB()

trend := trend(data)

i := 0

size := getSize(data)

i < size

season[i] = data[i] - trend

i = i + 1

season[] := adaptSeason(season)

errors[] := errors(data)

prognosis[] := prognos(trend, season, errors)

Рисунок 3.1 Алгоритм расчитывающий прогноз

На выше описанной блок схеме можно увидеть что метод в свою очередь так же вызывает сторонние методы. Это обусловденно тем что некоторые шаги при расчете прогноза, представляют собой грамоздкие вычислительные операции, и по всем правилам грамотного программирования, эти расчеты нужно вынести в отдельные методы. Этим мы добиваеся, упрощение кода, код а соответственно и блок схемы становистя более читабельными. Это позволяет значительно уменьшить число потенциальных ошибок в разрабатываемом програмном обеспечении, а также в случае необходимости, упростить вливаемость новых разработчиков в проект. Как правило новые разработчики берутся за проект по нескольким причинам, либо если предыдущий прекратил по каким либо причинам сотрудничество, либо в случае когда проект развивается, количество и круг задач значительно увеличивается и в этом случае привлечение новых разработчиков становится неизбежным.  
 Метод loadFromDB(), Отвечает за загрузку из базы данных информацию о продажах за прошедшие годы. В силу того что это операция не имеет отношения к расчетам а просто достает данные логично было вынести ее в отдельный метод. На рисунке 3.2 изображена блоксхема работы метода loadFromDB().

login, pass

conn := connectDB(login, passs)

conn.isConn

data[] : = conn.req ("from data('sale')")

print("errorDB")

Рисунок 3.2 - Блоксхема загрузки данных из базы

В метод tend(data[]) был вынесен расчет тенда. Этот метод получает в качастве параметра массив с данными о продажах, которые загружаются при помощи метода loadData(), на рисунке 3.3 изображена блоксхема работы метода tend(data[]).

coef[]

size := data[].size

i := 0

i < size

coef[i] = trendCoef(i, data)

determ = determination(coef)

i = i +1

Да

Нет

**Рисунке 3.3 - блоксхема работы метода tend(data[])**

В методе trend(data), выполняются операции такие как расчет коэффициентов ренда для каждого периода, а также детерминация тренда. На сегоднешний день практически для всех языков программирования есть разработанные математические библиотеки, PHP не исключение. Для расчета всех операций было принято решение использовать стороннюю математическую библиотеку. Такое решение было принято по следующим причинам. Во первых это значительно сокращает время, во вторых любой грамотный специалист знает что "изобретение велосипеда" не лучшее решение, так как, библиотеки разрабатываются как правило командамы которые сосредоточенны на решения именно того круга задач для которых предназначается эта библиотека, что гарантирует уменьшение количества ошибок. Так важним фактором является что сторонняя библиотека сама по себе является отдельным, а поэтому активно развивается целой командой. Под развитеем подрзумевается, выявление и иправление ошибок, а так же ускорение работы написанных алгоритмов.

Дальше вызывается метод errors(data[]) который предназначен для того что бы выполнить расчет ошибки модели, на рисунке 3.4 отабражен алгоритм работы данного метода.

**Рисунок 3.4 - блоксхема работы метода errors(data[])**

errors[]

size := data.size

i := 0

i < size

errors[i] := data[i] - model[i]

i := i +1

да

нет

Алгоритм этого метода довольно прост. Для рассчета ошибки модели нам необходимо получить разность между фактическими значениями и значениями модели. Для этого мы в цикле обходим два массива, массив с данными и массив с моделью. На каждой итерации дастается эллементы из одного и другого массива по номеру текущей итерации, высчитывается разность между ними и результат заносится в массив errors[], который и возвращает описываемая функция.

**5.2 Разработанный API**

Система состоит из нескольких отдельных частей, несколько клиентских, таких как панель для владельцев складов, панель для пукупателей, которые закупают оптом товар после чего выстовляют его на своих ветринах, так же присутствует серверная часть которая отвечает за хранение и обработку информации. Именно на сервере хранится и выполняется главный алгоритм прогназирования. Для взаимодействия этих частей а так же для того чтоб у нас была возможность предоставить сторонним разработчикам внедрять наш функционал в свои программные продукты был разработан API. Для большей гибкости, было реализованно несколько вариантов API.

**5.2.1 Rest API** REST или Representational State Transfer — «передача состояния представления» — архитектурный стиль взаимодействия компонентов распределённого приложения в сети. REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой гипермедиа-системы. В определённых случаях (интернет-магазины, поисковые системы, прочие системы, основанные на данных) это приводит к повышению производительности и упрощению архитектуры. В широком смысле компоненты в REST взаимодействуют наподобие взаимодействия клиентов и серверов во Всемирной паутине. REST является альтернативой RPC.

В сети Интернет вызов удалённой процедуры может представлять собой обычный HTTP-запрос (обычно GET или POST; такой запрос называют *REST-запрос*), а необходимые данные передаются в качестве параметров запроса.

Для веб-сервисов, построенных с учётом REST, то есть не нарушающих накладываемых им ограничений, применяют термин «RESTful».

Преимущества, филдинг указывал, что приложения, не соответствующие приведённым условиям, не могут называться REST-приложениями. Если же все условия соблюдены, то, по его мнению, приложение получит следующие преимущества:

• надёжность (за счёт отсутствия необходимости сохранять информацию о состоянии клиента, которая может быть утеряна);

• производительность (за счёт использования кэша);

• масштабируемость;

• прозрачность системы взаимодействия (особенно необходимая для приложений обслуживания сети);

• простота интерфейсов;

• портативность компонентов;

• лёгкость внесения изменений;

• способность эволюционировать, приспосабливаясь к новым требованиям (на примере Всемирной паутины).

**5.2.2 Библиотеки для сторонних клиентских систем**

Хотя разработанный Rest API дает полную творческую свободу разработчикам практически на любой платформе, не лишним будет разработать библиотеку. При получении ответов с cервера в формате JSON или XML, программисту приходится самому обрабатывать эти данные, реализуя собственные парсеры, приходится писать много кода и проверок при прочтении данных в этих форматах. Преймущество готово библиотеки в том что все эти действия уже выполнены за разработчика который использует эту библиотеку.   
 На данный момент реализована библиотека для платформы Андроид, при разработке использовался стандартный инструментарий, и соответственно использовался язык программировния Java.

Для использования разработчику необходимо скачать библиотеку с соответствующих ресурсов, и выполнить подключение.

Так же четко поставлена задача в будущем разработать библиотеку под остальные распространенные платформы как мобильные так и компьютерные.  
 При при написании кода использовались известные паттерны. Этим самым решается сразу две задачи. Во первых паттерны разрабатывались и отлаживались не один год, проверялись и утверждались высоквалифицированными профессианалами мирового уровня в сфере программирования. Использоване паттернов гарантирует что написанный код стабилин, и обладает более высокой производительностью. Вторая задача которая решается, любой более или менее грамотный программист, знает эти паттерны, что поможет разработчику разобраться с библиотекой не только опираясь на документацию но еще понимать на интуитивном уровне, что значительно ускорит полное освоение разработаной библиотеки.