**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.

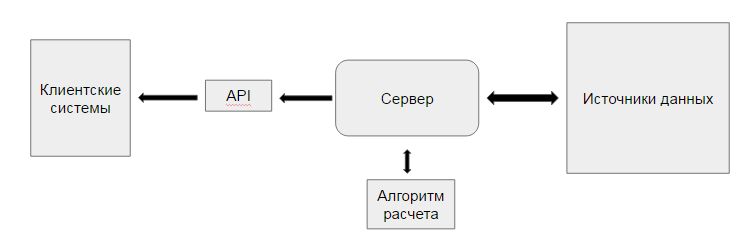


Рисунок 1 - Структура разрабатываемой системы

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 2014 году составил 310 млрд. рублей, 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 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. В случае когда наблюдаются большие расхождения между соседними значениями, тренд расчитанный методом регресии, сложно анализировать. В случае когда при построении прогноза ряд состоит из данных с высокой амплитудой колебаний между ближайшими значениями, выполняется сглаживание используя специальные методы. Существует довольно много таких методов самые известные среди них: метод экспоненциального сглаживания иметод скользящих средних.

Метод скользящих средних или МСС. МСС сглаживает ряд значений что позволяет выявить тренд. Для проведения расчетов данным методом используется среднее арифметическое значение определенного количества значений. Например трехточечное скользящее среднее, для это берутся первые три значения из списка например данные за январь февраль и март и считается среднее арифметическое значение которое ставится в центре диапазона то есть в феврале. Следующим выполняется сдвиг на один месяц и берется три следующих значения, то есть февраль, март, апрель, и считается среднее арифметическое значение уже между этими тремя значениями. По такому принципу обрабатывается весь ряд. Новые значения которые были получены в результате полной обработки ряда составляют новый ряд. Чем больше ряд содержит значений тем сильнее выполняется сглаживание. Пример построения тренда предоставлен в таблице # а так же изображен на рисунке #

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 |  |

Динамика колебаний исходных значений, которые были полученны методом скользящего среднего изображен на рисунке #. Результаты сравнения графиков рядов исходных значений и трехточечных скользящих средних показывают что удается сгладить колебания. Чем больше точек содержится в диапозоне вычисления тем четче будт вырисовываться тренд. Стоит отметить что операция укрупнения диапозона ведет к уменьшению количества конечных значений что приводит к менее точному прогнозу.



Рис. 5.4.

На рисунке(#) Избражена динамика изменения обьема продаж: ряд №3 - исходные данные; ряд №4 - скользящее среднее; ряд №2 - экспоненциальное среднее; ряд №1 - тренд, построенный методом регрессии.

Метод экспоненциального сглаживания. Еще один подход к уменьшению разброса значений ряда. Такое значение ему было присвоено из-за того что каждое значение периодов, уходящих в прошлое, уменьшается на множитель то есть 1 – α. Для расчета каждого значения используется формуа:

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**

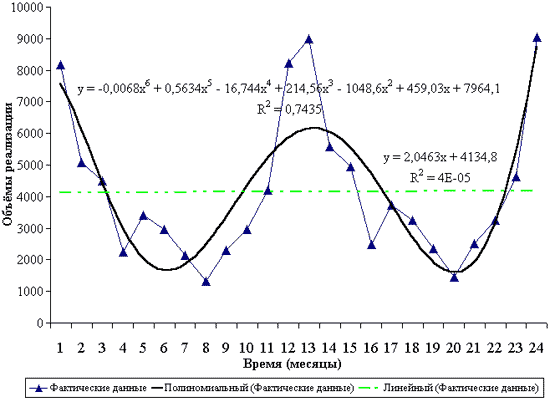
По резултатам исследования и выполненых эксперементов был выбран полиномиальный тренд. Результаты показали что данный тренд в сравнении с таким трендами как степенной а так же логарифмический или экспоненциальный оказался наиболее эффективным. Так как он наиболее удовлетворительно аппроксимирует фактические 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 р На рисунке # изображенна диаграмма, посмотрев на которую можно сделать вывод о том что полиномиальный тренд выполняет апроксимацию фактических данных гораздо эффективнее нежели другие часто используемые, например такой как линейный.

**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 выполняется вычетание значения тренда из фактических значений обьемов продаж. Это позволяет определить величину сезонной компоненты. После того как сезонная компонента определена ее необходимо откорректировать таким образом что бы сумма фактических значчеий и сезонной компоненты была равна нулю.

После определения сезонной компоненты и ее корректировки, выполняется расчет ошибки модели. Для этого нужно посчитать разность между фактическими значениями и значениями модели.

Когда посчитаны все значения, строится модель прогнозирования по формуле:

F = T + S ± E

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

После того как построена прогнозная модель, на ее основе расчитывается окончательный прогноз по продажам.Для сглаживания ряда был выбран метод эспонинциального сглаживания так как он позволяет учесть весьма вероятные будующие изменения экономических тенденций.

(#)

**где:**

– прогнозное значение объёма продаж;

– фактическое значение объёма продаж в предыдущем году;

– значение модели;

**–** константа сглаживания

Результаты применения метода на практике показали ряд его особенностей:

• Для построения прогноза достаточно указать величину сезона. Исследования показывают что у большинства продуктов присутствует именно сезонный характер, при этом величина сезона для каждого вида товара индивидуальна. Для одной группы товаров это может быть неделя, для другой группы это может быть год.

• Использование именно полиномиального тренда позволяет значительно сократить ошибку по сравнению с другими трендами.

• Если внести достаточное количество данных(в зависимости от велечины сезона) методот выполняет эффективную апроксимацию, и может быть полезен в построении стратегии ведения продаж.

**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 - система управления базой данных или СУБД. На сегоднешний день поддержкой и развитием занимается корпорация Oracle. Изначально MySQL ориентированна на работу в качестве отдельного сервера с которым по принципу запрос ответ будут взаимодействовать сторонние клиентские системы, однако есть библиотека позволяющая работать с базой автономно.

MySQL поддерживает различные типы таблиц, такие как MyIASM, InnoDB и другие.

Языки программирования. Практически для всех более или мении популярных языков разработанно API позволяющее взаимодействовать с базой данных.К этим языкам относятся Delphi, C, C++, Эйфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk, Компонентный Паскаль и Tcl, так же разработан инструментарий для платформы .NET.

Поддержка таблиц Maria или начиная с версии 5.2 и выше Aria.

Преймущества Maria:

•Если происходит аврийная ситуация при которой рушатся данные, Maria позволяет выполнить откат операции которая привела к аврийной ситуации. Так же еще есть возможность вернуться к состоянию зафиксированному по средствам команды LOCK TABLES.

• Позволяет восстановить состояние из любого места журнала ведения операций.

• Полностью поддерживает все типы данных информации что и таблици MyISAM.

• Размер страници данных 8Кб, для сравнения в MyISAM 1Кб, благодоря этому достигается высокая производительность для индексов полей с установленным фиксированным размером.

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

Если в При выборе операционной системы для сервера рассматривались 3 самых известных, а именно Windows Server, FreeBSD и операционные системы на базе Linux.

Производительность, анализ результатов исследования показал все упомянутые ОС имеют примерно одинаковую производительность. Однако Windows Server являетсяплатной ОС, а ее использование действительно оправдано только в случае если для разработки серверного программного обеспечения используются технологии от компании Microsoft, например, ASP, FrontPage или .NET. Использование этих технологий на других платформах выйдит дорогостоящим.

При разработке ЭСИТ не используются выше упомянутые технологии от Microsoft, поэтому использование платной Windows Server экономически не оправданно в связи с чем принято решение отказаться от использования этоЙ ОС.

При выборе между FreeBSD и Linux рассмотрен ряд приймущств и недостатков.

Linux это не операционная система а ядро, на базе которого на сегоднешний день создано много операционных систем, к ним относятся: Ubuntu, Red Hat, Mandriva, Gentoo и множество других. Среди этих ОС есть и серверные, однако все дистрибутивы Linux это действительно разные операционные системы, а учитывая их обширность при смене сервера порой трудно найти сервер с необходимым дистрибутивом, это ведет к отсутствию гибкости.

Что касается FreeBSD, то эта операционная система дейсствительно гибкая, она позволяет задать настройки, при помощи которм можно молучть если необходимо рабочую станицию, а если необходим сервер можно настроить как сервер. Что касается администрирования, важно отметит что это тоже не маловажный факт так как для этого нанимаются оплачиваются специалисты. Все дистрибутивы Linux сильно отличаются в администрировании то есть, способ установки програм, возможности установки настроек, работы с библиотеками, места хранения конфигурационных фаилов и множество других тонкостей. Очень невелеки гарантии того что нанятый специалист который долгое время администрирует текущий диструбутив сможет администрировать другой в случае необходимости. FreeBSD в этом случае выигрыват, по той причине что способы ее администрирования гараздо более гибкие нежели в дистрибутивах Linux. FreeBSD позволяет устанавливать программное обеспечение как из бинарных пакетов так и скомпилировать из портов, полная свободы в выборе оконных менеджеров, гибкость при настройке безопатности, у администратора есть возможность не ограничеваться стандартным фаерволом.

Со стороны Linux присутствует один очень важный факт который так же нельзя не отметить. Разработкой Linux не одна и та же сложившаяся команда а множество сторонних разработчиков, что не может отразиться на качестве.

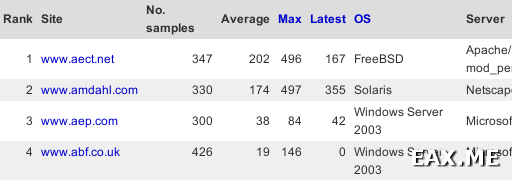
Если сравнивать Linux и FreeBSD по безопастности, то по результатом анализа статистики FreeBSD выигрывает. За год в FreeBSD находят примерно 4-5 уязвимостей, справедливо заметить что есть источники которые утверждают что ошибок находят в среднем 10, но это ни коим образом не портит картину, для сравнения в Linux за этот период находя примерно 120 уязвимостей. На рисунке# изображена гистограмма показывающая число ошибок которые были обнаружены в операционных системах FreeBSD, Windows, и ядре Linux за прошедшие 6 лет.



Изучив внимательно рисунок # можно заметить что в ядре Linux содержится большее число ошибок в отличае от других операционных систем. Важно понимать что Linux это только лиш ядро а не полноценная ОС, и не смотря на это у нее значительно большее количество ошибок чем у полноценных оперционных систем. Некоторые эксперты утверждают что это связанно с тем что ОС на базе Linux пользуется большее количество людей. Действительно учитывая широкий выбор дистрибутивов Linux, ним действительно пользуется большее количество людей. Но этот факт не оправдывает такого количества ошибок, так как их больше более чем в 10 раз, и это действительно очень большой размах. Более того наблюдается заметный рост ошибок в ядре Linux.

Выполнив анализ списка найденных ошибок в FreeBSD, сделан вывод что такие уязвимости могут редко могут быть полезными для злоумышленников.

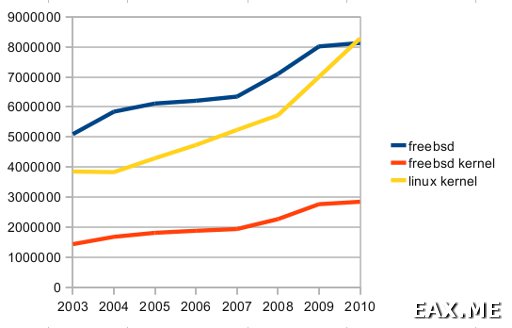
Именно благодоря безопасности FreeBSD пользуется популярностью у многих крупных хостинг-провайдеров таких как Masterhost, Majordomo и RU-CENTER.



При выборе операционной системы проведено еще одно исследование. Суть исследования заключалось в подсчете обьема кода для сравниваемых операционных систем. Любому более или менее опытному программисту известно что чем больше строк кода тем больше потенциальное количество

ошибок.

Подсчет выполнялся при помомщи инструмента CLOC (Count Lines Of Code). Эта утилита строки с кометариями а так же пустые строки не берет во внимание при подсчете. Исходники Linux находятся в свободном доступе и были скачаны с kernel.org. Исходники FreeBSD так же находятся в свободном доступе и были скачаны из SVN репозитория. Анализу подвергались только релизные версии вышедшие в период с 2008 по 2016 год. Информация датах выхода релизов находится на официальных сайтах. Н рисунке # изображен результат анализа.



На графике можно заметить нименования freebsd и freebsd kernel. Freebsd - это непосредственно сама операционная система, а freebsd kernel - это ядро выше упомянутой операционной системы.

Для получения обьективных результатов проводилось сравнение ядра с ядром. Как видно из графика размеры ядра линукс значительно привышают размер ядра FreeBSD. Учитывая тот факт что Linux не обеспечивает более широкую функциональность, был сделан вывод что такие большие размеры ничем не оправданы, и обьяняются лиш меньшей професианальностью учавствовавших разработчиков, что говорит о том что ядро операционной системы FreeBSD написанно более качественно.

Нельзя оставить без внимания тот факт что размер полностью всей операцонной системы FreeBSD всеравно меньше нежели размер только ядра Linux в чистом виде.

Еще одним очень важным моментом является то что обьем кода Linux стремительно растет, а пр нципиально новых преимуществ по сравнению с FreeBSD не появляется. Для сравнения в ядое Linux за прошедшие 7 лел добавилось 4 миллиона новых строк кода в FreeBSD за тот же период добавилось лиш 1.5 миллиона строк.

В двнном случае FreeBSD выигрывает по большенству параметров, но прежде чем окончательно определиться с выбором, проведен анализ недостатков этой ОС. Можно выделить 3 основных недостатка:

• Linux по сравнению с FreeBSD поддерживает значительно больше устройст. Однако эти устройства чаще всего используются в ноутбуках. Что косается устройств входящие в список комплектующих сервера, все они поддерживаются данной операционной системой, а значит ни как не отразится при решении поставленой задачи при проектировании ЭСИТ.

•Некоторое программное обеспечение например Skype распростроняется в виде бинарных фаилов только для Windows, Mac OS и для нескольких дистрибутивов Linux. При разработке ЭСИТ, FreeBSD используется в качестве серверной ОС а значит отсутствие поддержки подобного ПО никак не скажется на решении пирвичной задачи.

Спарведливо отметить что, в FreeBSD предусмотрена возможность запускать программы скомпилированные под Linux, а значить что в случае острой необходимости использовать подобное ПО, есть возможность его запустить.

По результатам рассмотренных недостатков сделан вывод, что они не коем образом не помешают решить поставленные здачи при разработке ЭСИТ, а значит операционная система FreeBSD подходит по всем параметрам, из-за чего и была выбрана именно она.

**4.1.4 Язык программирования PHP**

Главным Главная особенносто PHP это его практичность,

которая заключается в следующем:

• распространенный синтаксис

• наиболее популярен в web разработке

• прост в использовании

• при своей простоте обладает высокой эффективностью

• гибкость

• безопастность

• распростроняется под свободной лецензией

Распространенный синтаксис, практически 90% синтаксиса язык PHP были позаимствованы из языка Си, так же в некоторых моментах наблюдается сходство с языком Perl. Справедливо отметить что синтаксис языка Си признан наиболее одубным как для изучения так и для понимания. Именно по этому большенство начинающих специалистов в областе разработки программного обоспечения, первое с чем они пробуют работать это язык Си. Есть еще один факт который важно отметить, почти все самые распространенные языки программирования имеют Си подобный синтаксис. Из всего выше сказанного можно сделать вывод о том что найти специалиста работающего на данном языке будет значительно проще нежели специалистов пишуших на таких языках как Python или Rubby, которые в отличае от PHP имют далеко не распространенный синтаксис.

Наиболее популярен в web разработке, это немаловажный момент, так как это означает что с PHP работает очень много высоко квалефецированных спецеалистов, которы которые создают свои решения и делятся ими в различных источниках, таких как форумы, статьи соц сети и прочии. Эти решения отличаются своей эффективностью и практичностью, а благодоря тому что большенство из них свободно распростроняются, другий разработчики могут их изучить и внедрить в свой проект, что сэкономит время и уменьшит потенциальное количество ошибок.

Прост в использовании, при работе с PHP не прийдется тратить много времени на выставлении различных параметров компиляции, решать проблемы не совместимости библиотек и т.д. Код начинае исполняться сразу же. В случае отсутствия ошибок, код начинает исполняться именно так как указал программист, в отличае от других языков которые после компиляции могут менять свои механизмы. В большенстве случаев на работе программы это не отрожаются, однако бываю трудности из-за такого механизма работы как в этих языках.

У программиста работающего на PHP есть возможность встроить его в html код, который без проблем обрабатывается PHP интерпретатором.

Эффективность это бесспорно один из самых важных факторов. Язык PHP развивается уже достаточно длительное время, за которое "оброс" достаточно широким инструментарием входяшим в ядро PHP. В арсенале этого языка большое количество функций позволющих быстро и эффективно решеть различного рода задачи, такие как сложные математические вычисления, быстрая обработка большого количества данных, задачи шифрования и дешифрования, а так же множество других задач. Все упомянутые функции делают PHP действительно высокоэффективным языком, не смотря на его простоту использования.

Безопастность, в PHP прелусмотренна возможность гибкой настройки системы безопастности. Средства обеспечивающие безопастнося делятся на две группы: средства системного уровня и средства уровня приложения.

Средства системного уровня, предназначенны в первую очередь для администраторов сервера. Используя данные средства администратор может выставить ряд настроек, таких как, время выполнения скрипта, лимит на использование памяти, настраивать доступы к каталогам, а так же позволять или запрещать использовать отдельные php сценарии.

Средства уровня приложения, сюда входят механизмы надежного шифрования, совместимость с большинством приложений сторонних фирм что позволяет интегрировать готовый инструментарий выполняющий защиту информации. Такого рода приложения часто используютс в электронной коммерции, например для безопастного совершения платежей, где пользователь проходит авторизацию не выдавая свои скретные данные.

Сценарии PHP выполняются на удалунном сервере поэтому скрипт невозможно посмотреть в браузере. Это так же является средством безопастности, по скольку не возможно из браузера скачать скрипт который может выгодно послужить для злоумышлинеков.

Гибкость, PHP - это встраиваемый язык, который обладает очень высокой гибкостью. Как правило PHP используется совместно с HTML, однако на самом деле его возможности на этом не заканчиваются. PHP возможно так же совместно использовать с JavaScript, XML, WML а так же с рядом других языков.

При работе с PHP не возникает проблем с зависимостью от браузеров, так как скрипт написанный на PHP обрабатывается непосредственно на удаленном сервере И только после обработки результаты возвращаются клиентскому ПО.

Так же гибкость языка PHP еще заключается в том что этот язык не ореинтирован на какой-то один сервер. PHP работает на следующих серверах: Apache, Microsoft IIS, Netscape Enterprise Server, Stronghold а так же Zeus. Все выше перечисленные сервера работают практически на всех распространенных платформах таких как Linux, BSD, Windows, UNIX благодоря чему этот язык является платфомо независимым. Независимость языка от платформы позволяет в случае необходимости поменять опреционную систему на сервере без возникновения трудностей и необходимости каких либо адаптаций.

Еще одним очень важным фактом говорящем о гибкости PHP является возможность взаимодействовать со сторонними компонентами, например Enterprise Java Beans или COM-объекты Win32. Благодоря способности взаимодействовать с подобными сторонними компонентами PHP позволяет масштабировать проэкты до необходимых рамеров что позволяет ему занимать высокии позиции среди современных технологий.

Открытая лицензия, при всех своих преимуществах и мощности PHP еще в добавок имеет открытую лицензию что так же не сомненно является большим плюсом, так как позволяет сократить финансовые расходы и сделать проэкт экономически более выгодным не потеряв при этом качества работы.

**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 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.

6. ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ

Данная работа позволяет повысить прыбили в области эллектронной коммерции.

На сегоднешний день российский рынок эллектронной активно развивается, все больше и болше появляется успешных проектов. Выполняющих сбыт товаров а так же услуг по средствам сети интернет. Однако в обществе присутствует боязнь риска, так как для большинства людей это в новинку, а опытные предпрениматели которые на протяжении долгих лет вудет физическую торговлю и вовсе нехотят пробовать что либо не понятное.

Разрабатываемый продукт позволяет внести ястность для людей пробующих себя в этом бизнессе.

**6.1 Обоснование целесообразности**

**6.2 Оценка конкуренто способности**

**6.3 Планирования комплекса работ по разработке ЭСИТ и оценка трудоемкости**

При определении трудоемкости работ учитвабтся следующие факторы: срок сдачи, обьем поставленных задач, необходимое время на решение каждой задачи в отдельности.

Ожидаемая длительность выполнения работ расчитывается по формуле:

где − кратчайшая продолжительность данной работы (оптимистическая оценка);

− самая большая продолжительность работы (пессиместическая оценка);

− наиболее вероятная продолжительность работ (реалистическая оценка);

Оценка трудоемкости каждой отдельной задачи из всех поставленных приведена в таблице #. В разработке системы учавствует 3 человека, руководитель, инженер-программист, и системный администратор.

Таблица # – Оценка трудоемкости каждой отдельной задачи из всех поставленных.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Виды  работ | Оптимистическая оценка, *tmin* | Реалистическая оценка, *t*нв | Пессимистическая оценка, *tmax* | Ожидаемая  продолжительность  работы, *Т*ож |
| 1.1 | 15 | 17 | 18 | 17 |
| 1.2 | 7 | 8 | 9 | 8 |
| 1.3 | 12 | 14 | 16 | 14 |
| 2.1 | 2 | 3 | 5 | 3 |
| 2.2 | 5 | 7 | 8 | 7 |
| 3.1 | 8 | 14 | 16 | 13 |
| 3.2 | 19 | 21 | 23 | 21 |
| 3.3 | 2 | 5 | 7 | 5 |
| 3.4 | 3 | 4 | 8 | 5 |
| 4.1 | 2 | 3 | 4 | 3 |
| 4.2 | 3 | 4 | 5 | 4 |
| 4.3 | 11 | 16 | 17 | 15 |

Задача руководителя изучить проект, ознакомиться с требованиями которым должна соответствовать разрабатываемая система, в данном случае ЭСИТ, поставить ряд задач, определить какие именно технологии будут использоваться для решения этих задач. После того как определены задачи и технологии которые будут использоваться, руководитель распределяет задачи между специалистами соответствующих направлений.

Инжинер-программист занимется разработкой програмного обеспечения. В его задачи входит кодинг на соответствубщем языке программирования который был выбран руководителем, отладка разработанных им модулей с целью достижения наиболее высокой эффективности их работы, а так же, исправление ошибок которые были выявленны на более поздних этапах разработки.

Системный администратор. Весь принцип работы ЭСИТ бозируется на том что система должна быть постоянно дступна через сеть интернет, для этого необходим удаленный сервер на которм и будет размещен ЭСИТ. Задача системного администратора, установка и настройка на сервере необходимого программного обеспечения. Под настройкой програмного обеспечения подразумевается распределения прав доступа, выделение допустимой памяти. К задачам администратора так же относится работа с базой данных, а именно, обеспечение безопастности, хранение и восстановление разевных копий в случае аварийных ситуаций, а так же разрешение самих аварийных ситуаций.

Подробное описание всех стадий работ приведено в таблице #

Таблица #1 – Стадии выполняемых работ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Содержание работ | Исполнители | | | | Длительнось, дни | | | Загрузка | | | | | |
| дни | | | | % | |
| **1. Подготовка процесса разработки и анализ требований** | | | | | | | | | | | | | |
| 1.1 Исследование и обоснование разработки | | | | | | | | | | | | | |
| 1.1.1 Постановка задачи | | | | Руководитель | | | 3 | 1 | | | | 33 | |
| Программист | | | 3 | | | | 100 | |
| 1.1.2 Сбор исходных данных | | | | Руководитель | | | 14 | 5 | | | | 35 | |
| Программист | | | 14 | | | | 100 | |
| 1.2 Поиск аналогов и прототипов | | | | | | | | | | | | | |
| 1.2.1 Анализ существующих методов решения задачи и программных средств | | | Руководитель | | | | 6 |  | | | |  | |
| Программист | | | | 6 | | | | 100 | |
| 1.2.2 Обоснование принципиальной необходимости разработки | | | Руководитель  Программист | | | | 2 | 1  2 | | | | 50  100 | |
| 1.3 Анализ требований | | | | | | | | | | | | | |
| 1.3.1 Определение и анализ требований к проектируемой программе | | | Руководитель | | | | 3 | 1 | | | 33 | | |
| Программист | | | | 3 | | | 100 | | |
| 1.3.2 Определение структуры входных и выходных данных | | | Руководитель | | | | 5 | 1 | | | 20 | | |
| Программист | | | | 5 | | | 100 | | |
| 1.3.3 Выбор технических и программных средств реализации | | | Руководитель | | | | 3 | 1 | | | 33 | | |
| Программист | | | | 3 | | | 100 | | |
| 1.3.4 Согласование и утверждение технического задания | | | Руководитель | | | | 3 | 1 | | | 33 | | |
| Программист | | | | 3 | | | 100 | | |
| **Итого по этапу 1** | | | Руководитель | | | | **39** | **11** | | |  | | |
| Программист | | | | **39** | | |  | | |
| **2. Проектирование** | | | | | | | | | | | | | |
| 2.1 Проектирование программной архитектуры | | Руководитель | | | | 3 | | | |  |  | | |
| Программист | | | | 3 | 100 | | |
| 2.2 Техническое проектирование компонентов программы | | Руководитель | | | | 7 | | | |  |  | | |
| Программист | | | | 7 | 100 | | |
| **Итого по этапу 2** | | Руководитель | | | | **10** | | | |  |  | | |
| Программист | | | | **10** |  | | |
| **3. Программирование и тестирование программных модулей** | | | | | | | | | | | | | |
| 3.1 Программирование модулей в выбранной среде программирования | | Руководитель | | | | 13 | | |  | | | |  |
| Программист | | | | 13 | | | | 100 |
| 3.2 Тестирование программных модулей | | Программист | | | | 21 | | |  | | | |  |
| Программист | | | | 21 | | | | 100 |
| 3.3 Сборка и испытание программы | | Руководитель | | | | 5 | | | 2 | | | | 40 |
| Программист | | | | 5 | | | | 100 |
| 3.4 Анализ результатов испытаний | | Руководитель | | | | 5 | | | 1 | | | | 20 |
| Программист | | | | 5 | | | | 100 |
| **Итого по этапу 3** | | Руководитель | | | | **44** | | | **3** | | | |  |
| Программист | | | | **44** | | | |  |
| **4. Оформление рабочей документации** | | | | | | | | | | | | | |
| 4.1 Проведение расчетов показателей безопасности жизнедеятельности | | Руководитель | | | | 3 | | |  | | | |  |
| Программист | | | | 3 | | | | 100 |
| 4.2 Проведение экономических расчетов | | Руководитель | | | | 4 | | |  | | | |  |
| Программист | | | | 4 | | | | 100 |
| 4.3 Оформление пояснительной записки | | Руководитель | | | | 15 | | | 5 | | | | 33 |
| Программист | | | | 15 | | | | 100 |
| **Итого по этапу 4** | | Руководитель | | | | **22** | | | **5** | | | |  |
| Программист | | | | **22** | | | |  |
| **Итого по проекту** | | Руководитель | | | | **115** | | | **19** | | | |  |
| Программист | | | | **115** | | | |  |

По данным таблици #1 построен календарный план рабод который приведен в таблице #2.4. В плане последовательно описывается каждый отдельно взятый этап работы, а так же, взаимосвязь этих этапов.

Таблица #2.4 − Календарный план работ

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание работы | Исполнители | Длительность, дни | График работ |
| 1 Постановка задачи | Руководитель Программист | 1  3 | 20.01.07-20.01.07 20.01.07-22.01.07 |
| 2 Сбор исходных данных | Руководитель  Программист | 5  14 | 23.01.07-27.01.07  23.01.07-05.02.07 |
| 3 Анализ существующих методов решения задачи и программных средств | Программист | 6 | 06.01.07-10.02.07 |
| 4 Обоснование принципиальной необходимости разработки | Руководитель Программист | 1  2 | 11.02.07-11.02.07  11.02.07-12.02.07 |
| 5 Определение и анализ требований к программе | Руководитель Программист | 1  3 | 13.02.07-13.02.07 03.02.07- 15.02.07 |
| 6 Определение структуры входных и выходных данных | Руководитель Программист | 1  5 | 16.02.07-16.02.07  16.02.07-20.02.07 |
| 7 Выбор технических средств и программных средств реализации | Руководитель Программист | 1  3 | 13.03.07-13.03.07 13.03.07-15.03.07 |
| 8 Согласование и утверждение технического задания | Руководитель Программист | 1  3 | 16.03.07 – 16.03.07  16.03.07 –18.03.07 |
| 9 Проектирование программной архитектуры | Программист | 3 | 19.03.07 –21.03.07 |
| 10 Техническое проектирование компонентов программы | Программист | 7 | 22.03.07– 28.03.07 |
| 11 Программирование модулей в выбранной среде программирования | Программист | 13 | 29.03.07 – 10.07.07 |
| 12 Тестирование программных модулей | Программист | 21 | 11.04.07 – 01.05.07 |
| 13  Сборка и испытание программы | Руководитель  Программист | 2  5 | 02.05.07 – 03.05.07  02.05.07 – 06.05.07 |
| 14 Анализ результатов испытаний | Руководитель  Программист | 1  5 | 07.05.07 – 07.05.07  07.05.07 – 11.05.07 |
| 15 Проведение расчетов  показателей безопасности  жизнедеятельности | Программист | 3 | 12.05.07 – 14.05.07 |
| 16 Проведение экономических расчетов | Программист | 4 | 15.05.07 – 18.05.07 |
| 17 Оформление пояснительной записки | Руководитель Программист | 5  15 | 19.05.07 – 23.05.07  19.05.07 – 02.06.07 |

Используя данные из таблици #2.4 был построен график этапов выполнения работ который отображен на рисунке #2.1.



Рисунок #2.1 - График этапов выполнения работ

**6.4 Расчет затрат на разработку проекта**

Для расчета капитальных вложений в разработку ЭСИТ использовалась формула:

(#)

где:

−капитальные вложения на проектирование, руб;

−капитальные вложения на реализацию проекта, руб.

Проэктирование предстовляет собой предпроизводственную часть. На этом этапе происходит исследование общей задачи, изучение технологий необходимых для решения задач, разработка технического задания. Сюда включаются затраты на разработку алгоритмов.

Для определения общей суммы вложения на проектирование системы использовалась формула

(#)

где:

m – количество привлеченных на проект специалистов;

Зoi – финансовые затраты на оплату труда специалиста i-й категории, руб.;

Wd – Коэффициент финансовых затрат на дополнительную оплату труда. Сюда входит оплата отпускных коэффициент которых равен 0.1 и районого коэффициента в среднем равен 0.3. Учитывая значения всех входящих коэффициентов Wd ≈ 0.4;

Wс – коэффициент, учитывающий отчисления на социальные нужды, в долях к сумме основной и дополнительной заработной платы разработчиков (=0,262: страховые взносы в Пенсионный фонд в долях единицы – 0,2, страховые взносы в ФСС – 0,029, страховые взносы в ФОМС – 0,031, страховые взносы на производственный травматизм – 0,002);

Wн – коэффициент, учитывающий накладные расходы организации, в долях к основной заработной плате разработчиков (принимается по фактическим данным, Wн = 0,6);

СM – стоимость материалов;

Мв – оплата машинного времени.

Расчет финансовых затрат на оплату труда специалиста i-й категории выполняется по формуле:

(#)

где:

− средняя стоимость работы специалиста *i*-й категории за день, руб./дн.;

− отработанное количество дней специалистом *i*-й категории.

Время необходимое для разработки системы по каждому специалисту утверждается, ориентируясь на загрузку по календарному плану работ приведенного в таблици #2.4

Расчет оплаты работы специалиста приведен в таблице #2.5, при расчете учтено что в месяце примерно 21 рабочий день.

Таблица #2.5 – Основная заработная плата разработчиков

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Должность | Должностной оклад, руб. | Средняя дневная ставка, руб. | Затраты времени на разработку, человеко-дней | ОЗП, руб. |
| Руководитель Программист | 8189,74  1717,95 | 389,99  81,81 | 19  115 | 7409,81  9408,15 |
| Итого | | | | 16817,96 |

Учитывая то что специфика данной работы требует работу на компьютере, при расчете суммарных затрат обязательно нужно учитывать затраты на использования машинного времени, которые расчитываются по формуле:

(#)

где:

− время в течение которого будет использоваться компьютер для разработки системы =224 час.;

− стоимость одного часа машинного времени. ≈ 12 руб./час.;

− коэффициент мультипорадигменности, то есть кожффициент отображающий долю машинного времени выделенного именно на разработку проекта. = 1.

Использованные в ходе работы материалы, и финансовые затраты на их преобритение приведены в таблице #2.6.

Таблица 2.6 - Затраты на материалы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Материалы | Единица измерения | Требуемое  количество | Цена за единицу, руб. | Сумма, руб. |
| Тетрадь общая | шт. | 1 | 10 | 10 |
| Компакт-диск CD-RW | шт. | 2 | 35 | 70 |
| Тонер для лазерного принтера | шт. | 1 | 1000 | 1000 |
| Бумага офисная | пачка | 1 | 120 | 120 |
| Итого | | | | 1200 |

Из выше описанного, финансовые затраты на проктирование системы состовляют:

= (7409,81 + 9408,15)\*((1+0,4)\*(1 + 0,262) + 0,6) + 224\*12\*1+ 1200 руб. =

= 43692,75 руб.

В таблице #2.7 представлена смета финансовых затрат на разработку данного проекта.

Таблица 2.7 – Смета на разработку

|  |  |
| --- | --- |
| Статьи затрат | Сумма, руб. |
| Основная заработная плата | 16817,96 |
| Дополнительная зарплата | 6727,18 |
| Отчисления на социальные нужды | 6168,83 |
| Затраты на материалы | 1200 |
| Затраты на машинное время | 2688 |
| Накладные расходы организации | 10090,78 |
| ИТОГО | 43692,75 |

Общая сумма финансовых вложений на реализацию проекта определяется по формуле:

где:

− финансовые вложения на покупку обородывания;

− финансовые вложения на годовую аренду сервера ;

− финансовые вложения на создание информационной базы;

− финансовые вложения на подготовку или переподготовку кадров.

При разработки ЭСИТ, финансовые вложения на ее реализацию определяются финансовыми вложениями на покупку оборудывания для специалистов разрабытывающих проект, аренду сервера и т.д.

В оборудование входит три компьютера на базе процессора Intel Celeron, цена каждого такого компьютера состовляет ≈ 12000 руб.

Аренда сервера состовляет 5000 руб. в год.

Необходимая сумма финансовых вложений на закупку оборудования расчитывается по формуле:

(#)

где:

− стоимость j-го оборудования, руб. При n=1 = 12000руб;

− количество едениц j-го оборудования;

− коэффициент загрузки j-го оборудования при работе специалиста над поставленными ему задачами. Данный коэффициент расчитывается по формуле:

где:

− годовой резерв времени работы одной еденици оборудования j-го вида час ./год;

− количество времени отработанного j-ой еденицей оборудования над решением S задач, час/год:

где:

− количество часов однократной обработки инфомации при решении k-ой задачи на j-ой еденице оборудования. =6.

− частота решения k-й задачи, дней/год =264;

Таким образом финанасовые вложения на реализацию ЭСИТ состовляют:

= 22500\*1\*6\*264/(264\*8)руб = 16875 руб.

Таким образом, общая сумма финансовых вложений на создание проекта состовляет:

= 43692.75 + 16875 руб = 60567.75 руб.

Итого сумарные финансовые вложения на разработку проекта состовляют 61000 руб.