**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 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 каждой группы товаров может быть разным, поэтому длительность одного периода выставляется индивидуально пользователем в настройках.

ЭСИТ так же предоставляет инструментарий, который автоматизирует управление бизнес процессами. Опираясь на результаты прогноза, система самостоятельно 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 самостоятельно 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, состоящая из нескольких частей, каждая из которых выполняет свою функцию. На рисунке 1 изображена структура разрабатываемой системы.

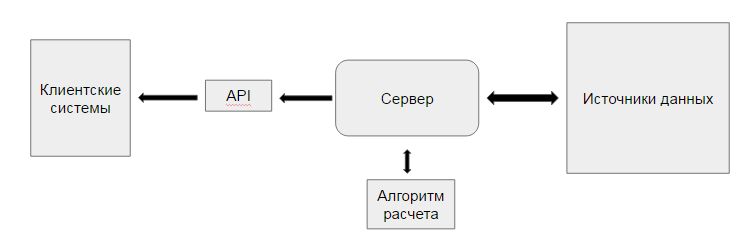


Рисунок 1 - Структура разрабатываемой системы

На сервере находится основная часть разработанного программного обеспечения, включая алгоритм прогнозирования, модуль управления базой знаний и непосредственно базу данных.   
 Источник данных представляет собой место хранения информации о товаре, это могут быть данные как со склада компании, 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, а так же мобильные и настольные приложения, разработанные под различные операционные системы.

**1 АКТУАЛЬНОСТЬ**

На сегодняшний 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 определенного потенциала для дальнейшего роста рынка электронной коммерции России.

Справедливо заметить, что на Российском рынке интернет-торговли присутствует дисбаланс в развитии (рисунок 3). Примерно 60% интернет-пректов ориентированы на крупные города, такие как Москва или Санкт-Петербург, где проживают около 15% населения страны. Эксперты из Morgan Stanley прогнозируют, что к 2019-2020 годам объемы продаж посредством сети интернет увеличатся на 30% для столичных городов, и примерно на 55% для региональных городов.



Рисунок 3 – Прогноз распределения продаж посредством сети Интернет по регионам России

По результатам исследований за 2016 год, на рынке российской интернет коммерции преимущественно распространяется цифровая продукция, доля которой составляет 64% в рублях - эта оценка составляет 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 интернет коммерция часто используется для оплаты коммунальных 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 составляет 36% от общей доли рынка, с высокой интенсивностью растут продажи обуви, одежды, косметики, а так же парфюмерии. Данная ниша активно развивается с момента формирования рынка (с того момента когда сформировался рынок), в конце 2011 - начале 2012 года общая сумма продаж составляла 136,9 млрд. рублей, по результатам анализа за 2016 год, этот показатель вырос уже вдвое. Эксперты прогнозируют, что в ближайшие два года доля продаж физических товаров закрепит свои позиции на 40 - 45% от общей суммы продаж на рынке интернет торговли. В Российской интернет коммерции присутствует ряд изъянов, которые препятствуют ее развитию. Одним из таких изъянов является тот факт, 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, которые являются сложными в эксплуатации виртуальных магазинов. Поэтому мы можем сделать вывод, что на сегодняшний день количество предпринимателей, которые смогли занять нишу российской электронной коммерции - это мизерный процент от того количества предпринимателей, которые пытались занять её, но их проекты оказались нежизнеспособными в силу отсутствия опыта ведения интернет бизнеса. Согласно результатам опроса многие предприниматели, как начинающие, так и опытные готовы инвестировать финансы в развитие собственного электронного предприятия. Однако их опыт не позволяет правильно составить списки товаров и расставить приоритеты для выхода (вывода?) на виртуальную витрину, а так же определить объемы необходимых закупок для владельцев собственных складов.

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

Кроме того, обязательно нужно принять во внимание приведенный выше факт о том, что достаточно большое количество людей готовы вкладывать средства в интернет коммерцию, но они опасаются делать это в силу отсутствия навыков ведения интернет-бизнеса. Это так же очень широкая аудитория потенциальных потребителей. Процент сбыта среди такого рода аудитории гораздо выше, так как они получают экспертную систему, которая выполняет все статистические анализы, рассчитывает прогноз, сортирует списки товаров, которые стоит закупать в будущем, а какие нет. Так же обладает 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 коммерции.

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

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

На сегодняшний день разработано достаточно много различных технологий в области прогнозирования.

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

Важно понимать, что такой величиной, как “сезон” в области прогнозирования могут быть любые временные колебания - сезонные вариации, например, в случае, когда проводится исследование товарооборота в течение одной недели, в качестве “сезона” выступает один рабочий день. К тому же, длительность цикла колебаний не всегда составляет один год - эта величина может варьироваться. После выявления величины цикла колебаний 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 единицах.

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

(#)

Ориентируясь на величину сезонной вариации возможно отличить аддитивную модель от мультипликативной. Для Аддитивной модели характерна практически 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 на рисунке 1.



Рисунок 1 − Аддитивная и мультипликативная модели прогнозирования

**1.1 Определение тренда**

Существует множество различных взаимозаменяемых методов определения тренда. Наиболее часто встречающиеся и рекомендуемые в различных литературных источниках методы - это логарифмический, линейный, степенной, экспоненциальный и полиномиальный. Задачей является выбор метода, 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 вышеперечисленных методов.  
 **1.2 Логарифмический тренд**

Логарифмический 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, чтобы 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 экспоненциальной функции не получают необходимых результатов, это происходит в силу того, что данные 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 в рядах динамики. Функцию, которая была задана 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 посредством 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 признака Y 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 представлен в виде линейной зависимости анализируемой величины вида

где

y – исследуемая переменная (например, производительность);

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

При выполнении линейной аппроксимации связи двух параметров, для нахождения эмпирических коэффициентов 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, рассчитывающие параметры линейной парной регрессии, 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, рассчитанный методом регрессии, сложно анализировать. В случае, когда при построении прогноза ряд состоит из данных с высокой амплитудой колебаний между ближайшими значениями, выполняется сглаживание с использованием специальных методов. Таких методов существует довольно много, самые известные среди них: метод экспоненциального сглаживания и метод скользящих средних.

Метод скользящих средних или МСС. МСС сглаживает ряд значений, что позволяет выявить тренд. Для проведения расчетов посредством данного метода используется среднее арифметическое значение определенного количества значений, например трехточечное скользящее среднее. Для этого берутся первые три значения из списка, например данные за январь, февраль и март, затем считается среднее арифметическое значение, которое ставится в центре диапазона, то есть в феврале. Далее выполняется сдвиг на один месяц и берутся три следующих значения, то есть февраль, март, апрель, затем считается среднее арифметическое значение уже между данными 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 представлен 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 получены методом скользящего среднего, изображена на рисунке #. Результаты сравнения графиков рядов исходных значений и трехточечных скользящих средних показывают наличие возможности сглаживания колебания. Чем больше точек содержится в диапазоне вычисления, тем отчетливее будет вырисовываться тренд. Стоит отметить тот факт, что операция укрупнения диапазона ведет к уменьшению количества конечных значений, что, в свою очередь, приводит к менее точному прогнозу.



Рис. 5.4.

На рисунке (#) изображена динамика изменения объема 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 в силу того, что каждое значение периодов, уходящих в прошлое, уменьшается на определенный 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) = 0,016

**1.4 Полиномиальный тренд**

Описать полиномиальный тренд

**1.5 Выбор наиболее точного расчета тренда**

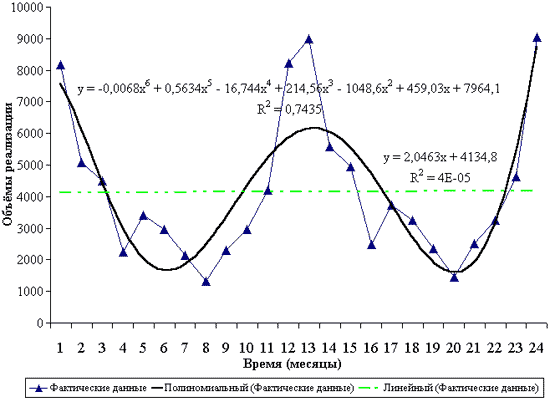
Согласно результатам исследований и проведенных экспериментов был выбран полиномиальный тренд. Результаты показали, что данный тренд в сравнении с таким трендами, как степенной, логарифмический или экспоненциальный, оказался наиболее эффективным, так как он наиболее удовлетворительно аппроксимирует фактические значения и имеет самый высокий коэффициент детерминации:

· логарифмический R2 = 0,0166;

· степенной R2 =0,0197;

· экспоненциальный R2 =8Е-05.

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



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

На рисунке # изображена диаграмма, изучив которую можно сделать вывод о том, что полиномиальный тренд выполняет аппроксимацию фактических данных гораздо эффективнее, нежели другие часто используемые тренды, например линейный.

**1.6 Алгоритм прогнозирования объема продаж пошагово**

Приведенный ниже алгоритм прогнозирования объема продаж имеет сезонный характер и состоит из нескольких шагов.

Первым шагом является определение тренда. Существует множество способов расчета тренда, самые распространенные из них описаны выше. С ними был проведен эксперимент, по результатам которого выбран полиномиальный тренд, так как именно он наилучшим образом позволяет сократить ошибку прогнозной модели.

Следующим шагом выполняется вычитание значения тренда из фактических значений объемов продаж. Это позволяет определить величину сезонной компоненты. После того, как сезонная компонента определена, ее необходимо откорректировать таким образом, чтобы сумма фактических значений и сезонной компоненты была равна нулю.

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

Когда посчитаны все значения, выполняется построение модели прогнозирования по формуле:

F = T + S ± E

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

На основе построенной прогнозной модели рассчитывается окончательный прогноз продаж. Для сглаживания ряда был выбран метод экспоненциального сглаживания, так как он позволяет учесть весьма вероятные изменения экономических тенденций в будущем.

(#)

**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 - год.

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

• При условии внесения достаточного количества данных (в зависимости от величины сезона), метод выполняет эффективную аппроксимацию и может быть полезен в построении стратегии ведения продаж.

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

С целью проверки точности прогноза, который предоставляет нам данный алгоритм, был проведен эксперимент. Для построения прогноза использовались данные об объёмах продаж мороженого “Белый пломбир”, полученные из отчетов одного из предприятий города Нижний Новгород за 2013 и 2014 годы. Для данной статистики характерно наблюдение явно выраженного сезонного характера и возрастающего тренда у значений объёма продаж. Задача заключалась в построении прогноза по продажам на 2015г. На момент написания диссертации (2016 год) нам доступны данные за прошедший 2015г. По результатам продаж за 2013 и 2014 годы был построен прогноз на 2015г., после чего выполнено сравнение с реальными цифрами продаж за 2015г. Исходная информация представлена в таблице 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.

Реализуется вышеописанный алгоритм расчета прогноза. 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.

Теперь определяется тренд. Был выбран полиномиальный тренд в силу того, что результаты вышеописанных экспериментов показали, что данный тренд выполняет аппроксимацию данных значительно эффективнее, чем любые другие, описываемые в большинстве литературных источниках. Его коэффициент детерминации составляет (0,74349) – данное значение гораздо выше значений линейного (4E-05) или любого другого тренда.

Далее определяется величина сезонной компоненты, для этого значение тренда вычитается из фактических значений объёмов продаж, в результате получены 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.

Сезонная компонента корректируется таким образом, 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. Расчет ошибки модели** как разницы 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 |

Находится среднеквадратическая ошибка 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 о том, что построенная модель выполняет эффективную аппроксимацию фактических данных, то есть она действительно отображает экономические тенденции, позволяющие определить объём продаж, что свидетельствует о способности построения высокоточного прогноза.

Теперь выполняется построение модели прогнозирования:

***F = T + S ± E***

На рисунке 1 изображена построенная модель.

**5.** Используя построенную модель, рассчитывается прогноз объёмов продаж. Для уменьшения влияния прошедших тенденций на точность результатов, помимо трендового анализа применяется 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 метод экспертных оценок, как вероятность сохранения неизменной экономической ситуации. То есть случай, когда главные характеристики колеблются с той же амплитудой, что и раньше, свидетельствует об отсутствии фактов, предвещающих грядущие изменения сложившейся экономической ситуации.



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

Далее был построен прогноз, ниже приведен пример построения за январь третьего сезона.

Расчет прогнозного значения модели:

Фактические продажи за предыдущий год (F\_ф t-1) составляют 2 361 руб. Коэффициент сглаживания равен 0.8. Приняв во внимание все значения, был построен следующий прогноз:

Так же следует отметить тот факт, что тенденции в экономике переменчивы. Для того чтобы учитывать такие изменения, необходимо выполнять построение прогноза с учетом наиболее свежих данных о фактических объемах продаж за прошедшие периоды. Результаты проведенных исследований показали, что максимальный срок актуальности данных составляет три прошедших периода. Например, если период равен году, то данные, которые можно использовать, имеют давность не более трех лет. Любые значения, полученные ранее, чем за три предыдущих периода, теряют свою актуальность.

**2.1 Точность прогнозов**

Итоги сравнения подтвердили высокую точность прогноза, результаты которого имели лишь небольшое отклонение от действительности. В таблице 2 приведено 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 совершенно незначительны для ведения продаж, так как оно не ведет к убыткам. Поэтому можно сделать вывод о том, что алгоритм действительно работает и имеет полезность. Проанализировать продажи за прошедший определенный период и правильно принять решение о закупках на следующий период может только опытный предприниматель, который ранее имел определенный опыт в данной сфере и неоднократно понес убытки за годы своей деятельности. Зачастую для начинающих предпринимателей такие ошибки обходятся довольно дорого, так как они неоднократно приводили к закрытию перспективного малого или крупного бизнеса только из-за неправильных расчетов. Так же риски уменьшаются в тех случаях, когда в качестве стартового капитала предприниматель использует кредит в банке. При условии грамотного расчета сезонных продаж, предприниматель получит более высокую прибыль, что поможет погасить кредит в установленный срок. Основываясь на вышесказанном можно сделать вывод о том, что полезность алгоритма высока и его использование в данной системе оправдано.

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

С целью применения алгоритма в разрабатываемой системе он был запрограммирован с использованием языка PHP. Алгоритм выполняется на удаленном сервере, взаимодействие с которым происходит посредством 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 тот, который позволяет реализовать соответствующий требованиям сервер. Для того чтобы было возможно решить поставленную задачу и разработанная система могла не только развиваться, а ещё и расширять свой функционал, не требуя при этом множество таких ресурсов, как время и финансы, которые выплачиваются разработчикам за их труд, удаленный сервер должен соответствовать 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 запросы в финансовой поддержке и т.д.

**4.1 Компоненты сервера**

В результате проведенного исследования различного серверного программного обеспечения, было принято решение в пользу FAMP в силу того, что он идеально соответствовал требованиям. Далее будет подробно описано значение FAMP, сравнение его каждой отдельно взятой компоненты с аналогами, где так же будут приведены более подробные пояснения того, что именно побудило к выбору непосредственно комплекса FAMP.

FAMP — комплекс серверного программного обеспечения. FAMP - это аббревиатура, составленная по первым буквам названий, входящих в данный 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.

Большинству людей известен альтернативный набор LAMP, который является тем же самым набором, в котором вместо FreeBSD используется операционная система Linux. Справедливо отметить, что именно LAMP задал тон именования подобных наборов серверного ПО.

Все программные продукты, входящие в состав подобных комплексов, разрабатывались отдельно в качестве независимых проектов, некоторые из них разрабатывались разными компаниями. Такой комплекс стал популярным в ходе развития такой области, как веб-разработка. Популярным он стал благодаря своей гибкости, а так же высокой производительности, наряду с этим все компоненты распространяются под открытой лицензией, что приводит к снижению финансовых затрат для проектов, использующих его. На сегодняшний день набор FAMP предоставляется многими хостинг площадками.

**4.1.1 Apache**

HTTP-сервер Apache – это кроссплатформенное ПО, которое поддерживается практически всеми существующими операционными системами. К главным преимуществам Apache относятся его надёжность и гибкость при конфигурировании. Все эти характеристики очень важны для разрабатываемой системы, кроссплатформенность обеспечивает гибкость в проектировании сервера. Гибкость данного ПО заключается в том, что при необходимости замены операционной системы на сервере не возникнет проблем с переходом на другую ОС (например, при необходимости использовать ПО под Linux системы, а не под FreeBSD, которая была выбрана на момент написания работы). В таблице 2.1 приведен список операционных 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 |

Зачастую в момент активного развития проект частично меняет свою специфику или расширяет ее, что приводит к появлению ряда задач, которые решаются проще и быстрее при условии использования другого языка программирования. Благодаря поддержке немалого количества языков программирования становится возможным разрабатывать отдельные части или полностью переписать систему на другом языке программирования, не затронув при этом другие настройки сервера. В таблице 2.2 приведен список поддерживаемых 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 предоставляет 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.

В ядре Apache реализовано следующее: работа с файлами конфигураций, HTTP(протокол), интерфейс для интеграции модулей, которые были разработаны сторонними разработчиками. На сегодняшний день разработкой ядра и его развитием занимается компания Apache Software Foundation без привлечения сторонних разработчиков, которые допускаются только к разработке модулей. Для написания ядра использовался язык Си.

Система конфигурации Apache основывается на файлах в текстовом формате, которые так же называются конфигурационными файлами. 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 файлами используется собственный язык, который основывается на блоках директив. Посредством файлов конфигураций возможно изменить практически любые параметры ядра. Большинство модулей имеют собственные параметры. Так же существуют модули, которые в своей работе такие конфигурационные файлы операционной системы как etc/pass или etc/host.

Система модулей. Apache HTTP Server интегрирует сторонние модули. На сегодняшний день разработано более 500 модулей, предоставляющих различную функциональность. Некоторые из этих модулей были разработаны непосредственно компанией Apache Software Foundation. Но большая часть была разработана сторонними open source-разработчиками.

Процесс интеграции модулей довольно гибкий. Модули можно интегрировать как в момент компиляции, так и загрузить динамически посредством конфигурационного файла с использованием директивы.

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 можно привести некоторые веб-приложения. 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. Он позволяет индивидуально задавать 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.

Существуют модули, предоставляющие возможность управлять такими ресурсами сервера 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 файлов.

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

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

Существуют модули, которые выполняют авторизацию, используя при этом системы управления базой данных или подключаемые модули аутентификации.

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

С целью шифрования данных, которые передаются между клиентом и сервером, используется алгоритм SSL, при написании которого использовалась библиотека OpenSSL. Для удостоверения в подлинности веб-сервера применяются сертификаты X.509.

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

Интернационализация. В версии 2.0 добавлена функциональность, позволяющая определить локаль клиента на стороне сервера. Поэтому теперь отправляемые браузеру системные сообщения (например, сообщения об ошибках) могут быть отправлены на разных языках.

Кроме того, важно отметить появление возможности управления на стороне сервера выводом содержимого страниц в зависимости от локали пользователя. Apache в состоянии работать с большим количеством кодировок, в том числе с Юникодом. Это дает возможность обрабатывать страницы, сохраненные в любых кодировках и написанные на любых языках.

Обработка событий. Администратору предоставляется возможность самостоятельно задать собственные страницы и обработчики для возникающих HTTP ошибок или событий, например 404 (страница не найдена) или 403 (запрещен доступ). В том числе предоставляется 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 сегодняшний день поддержкой и развитием занимается корпорация Oracle. Изначально MySQL ориентирована на работу в качестве отдельного сервера, с которым по принципу «запрос-ответ» будут взаимодействовать сторонние клиентские системы. Однако существует библиотека, позволяющая работать с базой автономно.

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

Языки программирования. Для большинства популярных языков разработано API, позволяющее взаимодействовать с базой данных. К таким 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.

Преимущества Maria:

• В случае аварийной ситуации, при которой рушатся данные, Maria позволяет выполнить откат операции, которая привела к аварийной ситуации. Так же предоставляется возможность возврата к состоянию, зафиксированному посредством команды LOCK TABLES.

• Позволяет восстановить состояние из любого места журнала ведения операций.

• Полностью поддерживает все типы данных информации, что и таблицы MyISAM.

• Размер страницы данных 8Кб, в отличие от MyISAM, где размер страницы данных составляет 1Кб. Благодаря этому достигается высокая производительность для индексов полей с установленным фиксированным размером.

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

При выборе операционной системы для сервера рассматривались 3 самые известные из них, а именно Windows Server, FreeBSD и операционные системы на базе Linux.

Производительность. Анализ результатов исследования показал, что все упомянутые ОС имеют примерно одинаковую производительность. Однако Windows Server является платной ОС, а ее использование действительно оправдано только в том случае, если для разработки серверного программного обеспечения используются технологии от компании Microsoft, например ASP, FrontPage или .NET. Использование данных технологий на других платформах будет 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 данной ОС.

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

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

Стоит отметить, что FreeBSD является гибкой операционной системой, которая позволяет задавать настройки и с помощью которой при необходимости можно получить рабочую станицу и настроить сервер. Администрирование так же является важным аспектом в силу того, что для его осуществления нанимаются и оплачиваются специалисты. Все дистрибутивы Linux отличаются в администрировании друг от друга способом установки программ, возможностью установки настроек, работой с библиотеками, местом хранения конфигурационных файлов и множеством других тонкостей. Нет никакой гарантии того, что нанятый специалист, который долгое время администрирует текущий дистрибутив, сможет администрировать другой в случае необходимости. FreeBSD в этом случае выигрывает в силу того, что способы ее администрирования гораздо более гибкие, нежели в дистрибутивах Linux. FreeBSD позволяет устанавливать программное обеспечение, как из бинарных пакетов, так и скомпилировать из портов, предоставляет полную свободу в выборе оконных менеджеров, гибкость при настройке безопасности, а так же предоставляет администратора возможность не ограничиваться стандартным фаерволом.

Важно отметить тот факт, что разработкой Linux занимается не одна сложившаяся команда, а множество сторонних разработчиков, что не может отразиться на качестве.

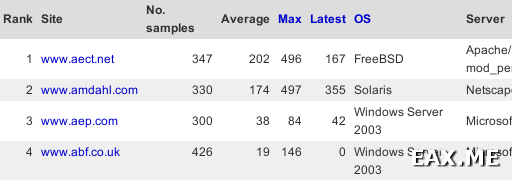
В сравнении Linux и FreeBSD по уровню безопасности, FreeBSD превосходит Linux согласно результатом анализа статистики. За год в FreeBSD выявлены примерно 4-5 уязвимостей. Справедливо заметить, что существуют источники, которые утверждают, что количество выявленных ошибок составляет в среднем 10, для сравнения в Linux за этот период выявлены около 120 уязвимостей. На рисунке # изображена гистограмма, которая показывает число ошибок, обнаруженных в операционных системах FreeBSD, Windows и ядре Linux за прошедшие шесть лет.



Изучив рисунок # можно заметить, что в ядре Linux содержится большее число ошибок, чем в других операционных системах. Важно понимать, что Linux - это ядро, а не полноценная ОС, но, не смотря на это, у нее значительно большее количество ошибок, чем у полноценных операционных систем. Эксперты утверждают, что это связанно с тем, что ОС на базе Linux пользуется большее количество людей. Учитывая широкий выбор дистрибутивов Linux, ним действительно пользуется большее количество людей. Но этот факт не оправдывает такое количество ошибок, более того, сейчас наблюдается их заметный рост.

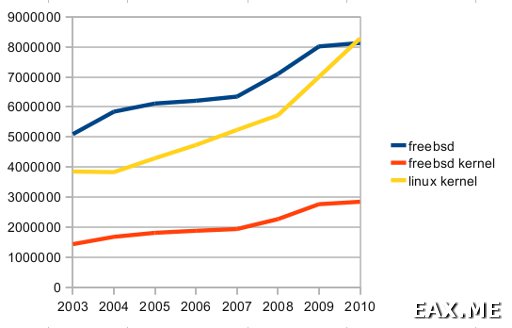
Выполнив анализ списка найденных ошибок в FreeBSD, можно сделать вывод, что такие уязвимости могут редко могут быть полезными для злоумышленников.

Благодаря своей 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 объема кода для сравниваемых операционных систем. Любому опытному программисту известен факт о том, что при большом количестве строк кода потенциальное количество ошибок возрастает.

Подсчет выполнялся при помощи инструмента CLOC (Count Lines Of Code). Данная утилита строки с комментариями, 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 о датах выхода релизов находится на официальных сайтах. На рисунке # изображен результат анализа.



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

Для получения объективных результатов проводилось сравнение ядра с ядром. На графике видно, что размеры ядра Linux значительно превышают размер ядра FreeBSD. Учитывая тот факт, что Linux не обеспечивает более широкую функциональность, можно сделать вывод, что такие большие размеры ничем не оправданы и объясняются лишь меньшей профессиональностью участвовавших разработчиков. Это говорит о том, что ядро операционной системы FreeBSD написано более качественно.

Нельзя оставить без внимания тот факт, что размер всей операционной системы FreeBSD меньше размера ядра Linux в чистом виде.

Важно отметить, что объем кода Linux стремительно растет, а принципиально новые преимущества по сравнению с FreeBSD не выявлены. В ядре Linux за прошедшие семь лет добавилось 4 миллиона новых строк кода в то время, как в FreeBSD за тот же период добавилось лишь 1.5 миллиона строк.

В данном случае FreeBSD превосходит Linux по большинству параметров, но прежде, чем окончательно определиться с выбором, был проведен анализ недостатков данной ОС.

Выявлены три основных недостатка:

• Linux по сравнению с FreeBSD поддерживает значительно большее количество устройств. Однако такие устройства чаще всего используются в ноутбуках. Что касается устройств, входящих в список комплектующих сервера, все они поддерживаются данной операционной системой, а значит этот факт никак не отразится при решении поставленной задачи при проектировании ЭСИТ.

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

• Справедливо отметить, что в FreeBSD предусмотрена возможность запуска программ, скомпилированных под Linux, это означает, что в случае острой необходимости использовать подобное ПО есть возможность его запуска.

По результатам рассмотренных недостатков можно сделать вывод о том, что они никак не помешают решить поставленные задачи при разработке ЭСИТ, а значит операционная система FreeBSD подходит по всем параметрам, в силу чего была выбрана именно она.

**4.1.4 Язык программирования PHP**

Главной особенностью PHP является его практичность, которая заключается в следующем:

• распространенный синтаксис

• высокая популярность в web-разработке

• простота в использовании

• обладание высокой эффективностью

• гибкость

• безопасность

• возможность распространения под свободной лицензией

Распространенный синтаксис. Практически 90% синтаксиса языка PHP были заимствованы из языка Си, так же наблюдается сходство с языком Perl. Справедливо отметить, что синтаксис языка Си признан наиболее удобным для изучения и понимания. Именно поэтому большинство начинающих специалистов в области разработки программного обеспечения изначально пробуют работать с языком Си. Важно отметить, что большинство распространенных языков программирования имеют Си подобный синтаксис. Из вышесказанного можно сделать вывод о том, что найти специалиста, работающего с данным языком значительно проще, нежели специалиста, пишущего на таких языках, как Python или Rubby, которые в отличие от PHP имеют нераспространенный синтаксис.

PHP наиболее популярен в web-разработке, это означает, что с данным языком программирования работают много высококвалифицированных специалистов, которые создают свои решения и делятся ими в таких источниках, как форумы, статьи, социальные сети и прочие. Такие решения отличаются своей эффективностью и практичностью. Благодаря тому, что большинство из них свободно распространяются, другие разработчики имеют возможность изучить их и внедрить в свой проект, что сэкономит время и уменьшит потенциальное количество ошибок.

Простота в использовании. При работе с PHP не придется тратить большое количество времени на выставление различных параметров компиляции, решать проблемы несовместимости библиотек и т.п. Код начинает исполняться сразу же. В случае отсутствия ошибок, код начинает исполняться именно так, как указал программист, в отличие от других языков, которые после компиляции могут менять свои механизмы. В большинстве случаев на работе программы это не отражаются, однако, иногда возникают трудности из-за такого механизма работы.

Программист, работающий на PHP, имеет возможность встроить его в html код, который без проблем обрабатывается PHP интерпретатором.

Его эффективность является одним из наиболее важных факторов. Язык PHP развивается уже достаточно длительное время, за которое приобрел достаточно широкий инструментарий, входящий в ядро PHP. В арсенале данного языка находится большое количество функций, позволяющих быстро и эффективно решать задачи различного, например: 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, несмотря на его простоту использования.

Безопасность. В PHP предусмотрена возможность гибкой настройки системы безопасности. Средства, обеспечивающие безопасность, делятся на две группы: средства системного уровня и средства уровня приложения.

Средства системного уровня предназначены для администраторов сервера. Используя данные средства, администратор имеет возможность выставить ряд таких настроек, как время выполнения скрипта, лимит на использование памяти, а так же может настроить доступы к каталогам, позволять или запрещать использование отдельных php сценариев.

Средства уровня приложения. К ним относятся механизмы надежного шифрования, совместимость с большинством приложений сторонних фирм, что позволяет интегрировать готовый инструментарий, выполняющий защиту информации. Такого рода приложения часто используются в электронной коммерции, например для безопасного совершения платежей, где пользователь проходит авторизацию, не выдавая свои секретные данные.

Сценарии PHP выполняются на удаленном сервере, поэтому нет возможности посмотреть скрипт в браузере. Это так же является средством безопасности в силу отсутствия возможности скачивания скрипта из браузера, так как это могло бы выгодно послужить злоумышленникам.

Гибкость. PHP - это встраиваемый язык, который обладает довольно высоким уровнем гибкости. Как правило, PHP используется совместно с HTML, однако на самом деле его возможности этим не ограничиваются. Существует возможность использовать PHP совместно 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 языке PHP, обрабатывается непосредственно на удаленном сервере. Только после обработки результаты возвращаются клиентскому ПО.

Кроме того, гибкость языка PHP заключается в том, что данный язык не ориентирован на какой-либо 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, благодаря этому данный язык является платформо-независимым. Независимость языка от платформы при необходимости позволяет поменять операционную систему на сервере без возникновения трудностей и необходимости каких - либо адаптаций.

Еще одним важным фактом, подтверждающим гибкость PHP, является возможность взаимодействия со сторонними компонентами, например Enterprise Java Beans или COM-объекты Win32. Благодаря способности взаимодействия с подобными сторонними компонентами, PHP предоставляет возможность масштабировать проекты до необходимых размеров, что позволяет ему занимать высокие позиции среди современных технологий.

Открытая лицензия. В дополнение ко всем своим преимуществам и мощности, PHP имеет открытую лицензию, что, несомненно, является большим плюсом, так как это позволяет сократить финансовые расходы и сделать проект экономически выгодным без ущерба качеству работы.

**5 РАЗРАБОТКА ПРОГРАМНОГО ОБЕСПЕЧЕНИЯ**

После того, как с алгоритмом были проведены все эксперименты и отпали все сомнения в достоверности его прогнозов, а так же полностью определены конфигурации серверного оборудования, которое обеспечивает необходимый уровень защиты, производительности и рациональности в финансовых запросах, 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 классе "PrognosManager" - это calcPrognose(), который и объединяет 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 рассчитывающий прогноз

На вышеописанной блок схеме можно увидеть, что данный метод, в свою очередь, вызывает сторонние методы. Это обусловлено тем, что некоторые шаги при расчете прогноза представляют собой громоздкие вычислительные операции, а по всем правилам грамотного программирования данные расчеты нужно вынести в отдельные методы. Таким образом, код становится более упрощенным, и, соответственно, блок - схемы становятся более читабельными. Это позволяет значительно уменьшить число потенциальных ошибок в разрабатываемом программном обеспечении, а также, в случае необходимости, упростить привлечение новых разработчиков в проект. Как правило, новые разработчики начинают работать над проектом по нескольким причинам: если предыдущий разработчик по каким - либо причинам прекратил сотрудничество, либо в случае, когда проект развивается, количество и круг задач значительно увеличивается, в таком случае привлечение новых разработчиков становится неизбежным.  
 Метод loadFromDB() отвечает за загрузку информации о продажах из базы данных за прошедшие годы. В силу того, что данная операция не имеет отношения к расчетам, а просто берет необходимые данные, логично было вынести ее в отдельный метод. На рисунке 3.2 изображена блок - схема работы метода loadFromDB().

login, pass

conn := connectDB(login, passs)

conn.isConn

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

print("errorDB")

Рисунок 3.2 – Блок - схема загрузки данных из базы

В метод tend(data[]) был вынесен расчет тенда. Данный метод в качестве параметра получает массив с данными о продажах, которые загружаются при помощи метода loadData(), на рисунке 3.3 изображена блок - схема 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

**Рисунок 3.3 – блок - схема работы метода tend(data[])**

В методе trend(data) выполняются такие операции, как расчет коэффициентов тренда для каждого периода, а также детерминация тренда. На сегодняшний день для большинства языков программирования существуют разработанные математические библиотеки, PHP не является исключением. Для расчета всех операций было принято решение использовать стороннюю математическую библиотеку. Такое решение было принято по причине того, что это значительно сокращает время. Кроме того, любой грамотный специалист знает, что "изобретение велосипеда" не является лучшим решением, так как зачастую библиотеки разрабатываются командами, которые сосредоточены на решении непосредственно тех задач, для которых предназначается данная библиотека, что гарантирует уменьшение количества ошибок. Так же важным фактором является то, что сторонняя библиотека сама по себе является отдельным, в силу чего активно развивается целой командой. Под развитием подразумевается выявление и исправление ошибок, а так же ускорение работы написанных алгоритмов.

Далее вызывается метод errors(data[]), который предназначен для того, чтобы выполнить расчет ошибки модели. На рисунке 3.4 отображен алгоритм работы данного метода.

**Рисунок 3.4 – блок - схема работы метода errors(data[])**

errors[]

size := data.size

i := 0

i < size

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

i := i +1

да

нет

Алгоритм данного метода довольно прост. Для расчета ошибки модели  необходимо получить разницу между фактическими значениями и значениями модели. Для этого в цикле находятся два массива: массив с данными и массив с моделью. На каждой итерации берутся элементы из одного и другого массива по номеру текущей итерации, высчитывается разница между ними и результат заносится в массив errors[], который возвращается описываемой функцией.

**5.2 Разработанный API**

Система состоит из нескольких отдельных частей: клиентских - панель для владельцев складов, панель для покупателей, которые закупают оптом товар и выставляют его на своих витринах, так же присутствует серверная часть, которая отвечает за хранение и обработку информации. Именно на сервере хранится и выполняется главный алгоритм прогнозирования. Для взаимодействия этих частей, а так же для наличия возможности предоставить сторонним разработчикам возможность внедрять данный функционал в свои программные продукты был разработан API. Для большей гибкости было реализовано 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.

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:

− нименьшие временные затраты на выполнение работы, оптиместический прогноз;

− наибольшие временные затраты, пессимистический прогноз;

− самые вероятные временные затраты на выполнение работ, реалистический прогноз;

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 из всех поставленных задач.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер задачи | Оптиместический прогноз , *tmin* | Реалистический прогноз , *t*нв | Пессимистический прогноз , *tmax* | Ожидаемые временнвые затраты на выполнение задач, *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 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 аварийных ситуаций в момент их возникновения.

Подробное описание всех стадий работ приведено в таблице #1

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 так же включаются затраты на разработку алгоритмов.

Для определения общей суммы вложения на проектирование системы использовалась формула:

(#)

где:

m – количество привлеченных на проект специалистов, m = 3;

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 соц отчислений, в определенных долях от оплаты труда нанятых специалистов. Wс = 0.262.В таблице #6.1 приведено подробное описание состовляющих данного коэффициента.

Таблица #6.1 состовлющии коэффициента соц отчислений

|  |  |
| --- | --- |
| Наименование | Доля от з/п специалиста |
| Персионный фонд | 0.2 |
| Страховые взносы в ФСС | 0.029 |
| Страховые взносы в ФОМС | 0.031 |
| Страховые взносы на производственный травмотизм | 0.002 |
| Итого | 0.262 |

Wн – коэффициент накладных расходов, в определенных долях от оплаты труда нанятых специалистов. Значение взято из фактических данных 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 | 22000 | 800 | 26 | 20800 |
| Программист | 18000 | 680 | 39 | 26520 |
| Системный администратор | 14000 | 500 | 10 | 5000 |
| Итого | | | | 52320 |

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 =600 час.;

− стоимость одного часа машинного времени. ≈ 20 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 | Необходимое  количество | Стоимость за единицу, руб. | Сумма, руб. |
| Журнал для записей | шт. | 1 | 60 | 60 |
| Краска для печати | шт. | 1 | 1100 | 1100 |
| Бумага для принтера | упаковка | 1 | 284 | 284 |
| Итого | | | | 1444 |

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

|  |  |
| --- | --- |
| Описание расходов | Сумма, руб. |
| Затраты на оплату труда | 16817,96 |
| Дополнительная оплата труда, премия, бонусы и т.д. | 6727,18 |
| Соц отчисления | 6168,83 |
| Финансовые вложения на оплату материалов | 1200 |
| Финансовые вложения на оплату машинного времени | 2688 |
| Накладные финансовые затраты | 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.