АКАДЕМИЯ МАРКЕТИНГА И СОЦИАЛЬНО-ИНФОРМАЦИОННЫХ

ТЕХНОЛОГИЙ – ИМСИТ (г. Краснодар)

Факультет информатики и вычислительной техники

Кафедра математики и вычислительной техники

Допустить к защите

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_ Н.С. Нестерова

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему: «Разработка системы электронного расписания для учебно-методического управления образовательных организаций» (по материалам ЧУ ПОО «Краснодарский техникум управления, информатизации и сервиса» (г. Краснодар), г. Краснодар)

Направление подготовки 230100.62 «Информатика и

вычислительная техника»

Работу выполнил студент

4 г. заочной формы обучения,

группы 14-ЗИВТ-СПО-02

Скляр Дмитрий Эдуардович

Научный руководитель, к.т.н., профессор Н. С. Нестерова

Краснодар

2017

АКАДЕМИЯ МАРКЕТИНГА И СОЦИАЛЬНО-ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ — ИМСИТ (г. Краснодар)

Факультет информатики и вычислительной техники

Кафедра математики и вычислительной техники

Утверждаю

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_20\_\_г.

**ЗАДАНИЕ**

на выпускную квалификационную работу

Студенту Скляр Дмитрию Эдуардовичу

Тема выпускной квалификационной работы:

«Разработка системы электронного расписания для учебно-методического управления образовательных организаций» (по материалам ЧУ ПОО «Краснодарский техникум управления, информатизации и сервиса» (г. Краснодар), г. Краснодар)

Утверждена приказом ректора от « 22 » июля 2014 г. №1001

Целевая установка: улучшение качества образовательного процесса, повышение интереса студентов к изучаемой дисциплине.

Основные вопросы, подлежащие разработке: анализ предметной области, создание интерактивного и мультимедийного контента, разработка программного кода.

Основная литература:

1. Тжаскалик Тадеуш - Введение в исследование операций с применением компьютера. - М.: Горячая Линия-Телеком, 2010. — 310 с.

2. В. В. Фаронов - Delphi. Программирование на языке высокого уровня. Учебник для вузов. – Спб.: Питер, 2012. — 348с.

Срок представления законченной работы « 20 » декабря 2014 г.

Дата выдачи задания: «2» сентября 2014г.

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Нестеров Г.Д. /

(подпись) (Ф.И.О.)

Задание получил: «2» сентября 2014г.

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Коначев В.С. /

(подпись) (Ф.И.О.)

РЕФЕРАТ

Выпускная квалификационная работа 90с.,23рис., 7 табл., 25 источников, 1 прил.

ЧУ ПОО КТУИС, РАСПИСАНИЕ, БАЗА ДАННЫХ,

Объектом исследования является работа подразделения учебно-методического управления образовательной организации, в которую входит составление расписания для обучающихся среднего профессионального образования.

Цель работы состоит в автоматизации процесса составления расписания занятий.

К полученным результатам относится разработанная система электронного расписания. Она позволяет автоматизировать работу подразделения учебно-методического управления образовательной организации, предоставив ему автоматически составленное расписание занятий образовательной организации, с последующим, при необходимости, его модифицированием, для достижения исключительно необходимого результата, вывод необходимых отчётов.

Новизна результатов заключается в разработке эффективной системы создания электронного расписания для организации среднего профессионального образования в условиях федеральных государственных образовательных стандартов.

Внедрение результатов предусмотрено в ЧУ ПОО «Краснодарский техникум управления, информатизации и сервиса».

Эффективность работы характеризуется автоматизацией работы, скоростью получения необходимого результата, упрощением, за счёт исключения второстепенных данных.

Определения, обозначения и сокращения

ОО – образовательная организация

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 8](#_Toc503217392)

[1 Исследование предметной области 11](#_Toc503217393)

[1.1 ЧУ ПОО «Краснодарский техникум управления, информатизации и сервиса» 11](#_Toc503217394)

[1.2 Система электронного расписания занятий 12](#_Toc503217395)

[2 Обзор существующих продуктов 17](#_Toc503217396)

[2.1 «Экспресс-расписание Колледж» 17](#_Toc503217397)

[2.2 «Ректор-Колледж» 18](#_Toc503217398)

[2.3 Конфигурация «1С: Автоматизированное составление расписания. Колледж» 19](#_Toc503217399)

[3 Требования к программному продукту 21](#_Toc503217400)

[3.1 Постановка задачи 21](#_Toc503217401)

[3.2 Анализ требований 22](#_Toc503217402)

[4 Техническое задание на разработку 23](#_Toc503217403)

[4.1 Введение 23](#_Toc503217404)

[4.1.1 Наименование программы 23](#_Toc503217405)

[4.1.2 Краткая характеристика области применения 23](#_Toc503217406)

[4.2 Назначения разработки 23](#_Toc503217407)

[4.2.1 Функциональное назначение 23](#_Toc503217408)

[4.2.2 Эксплуатационное назначение 23](#_Toc503217409)

[4.3 Требования к программе или программному изделию 24](#_Toc503217410)

[4.3.1 Требования к функциональным характеристикам 24](#_Toc503217411)

[4.3.1.1 Требования к составу выполняемых функций 24](#_Toc503217412)

[4.3.1.2 Требования к организации входных данных 24](#_Toc503217413)

[4.3.2.3 Требования к организации выходных данных 25](#_Toc503217414)

[4.3.1.4 Требования к временным характеристика 25](#_Toc503217415)

[4.3.2 Требования к надёжности 25](#_Toc503217416)

[4.3.2.1 Требования к обеспечению надежного (устойчивого) функционирования программы 25](#_Toc503217417)

[4.3.2.2 Время восстановления после отказа 26](#_Toc503217418)

[4.3.2.3 Отказы из-за некорректных действий оператора 26](#_Toc503217419)

[4.3.3 Условия эксплуатации 26](#_Toc503217420)

[4.3.3.1 Климатические условия эксплуатации 26](#_Toc503217421)

[4.3.3.3 Требования к численности и квалификации персонала 27](#_Toc503217422)

[4.3.4 Требования к составу и параметрам технических средств 27](#_Toc503217423)

[4.3.5 Требования к информационной и программной совместимости 27](#_Toc503217424)

[4.3.5.1 Требования к информационным структурам и методам решения 28](#_Toc503217425)

[4.3.5.2 Требования к исходным кодам и языкам программирования 28](#_Toc503217426)

[4.3.5.3 Требования к программным средствам, используемым программой 28](#_Toc503217427)

[4.3.5.4 Требования к защите информации и программ 28](#_Toc503217428)

[4.3.6 Требования к маркировке и упаковке 28](#_Toc503217429)

[4.3.6.1 Требования к маркировке 29](#_Toc503217430)

[4.3.6.2 требования к упаковке 29](#_Toc503217431)

[4.3.6.3 Условия упаковывания 29](#_Toc503217432)

[4.4 Требования к программной документации 29](#_Toc503217433)

[4.4.1 Предварительный состав программной документации 29](#_Toc503217434)

[4.5 Технико-экономические показатели 30](#_Toc503217435)

[4.5.1 Экономические преимущества разработки 30](#_Toc503217436)

[4.6 Порядок контроля и приемки 30](#_Toc503217437)

[4.6.1 Виды испытаний 30](#_Toc503217438)

[5 Модель предметной области 31](#_Toc503217439)

[6 Выбор метода и инструментов разработки 37](#_Toc503217440)

[6.1 Выбор среды программирования 37](#_Toc503217441)

[6.2 Выбор языка программирования 40](#_Toc503217442)

[7 Реализация разработки 45](#_Toc503217443)

ВВЕДЕНИЕ

В современном информационном обществе значительное внимание отводится информатизации в сфере образования. Информатизация образования — это процесс обеспечения системы образования теорией и практикой разработки и использования новых информационных технологий, ориентированных на реализацию целей обучения и воспитания. Процесс информатизации образования невозможен без оснащения учреждений образования и органов управления образованием аппаратными и программными средствами информационных технологий.

В настоящее время задача правильного формирования расписания учебных занятий является значимой частью в управлении учебным процессом. Таким образом, проблема автоматизации составления расписания учебных занятий в учебных заведениях является одной из преимущественно значимых проблем в организации учебного процесса. От успешности составленного расписания зависит:

- качество знаний;

- результативность обучения;

- удобство обучения студентов и работы преподавателей.

Автоматизация процесса составления расписания занятий позволит значительно сократить временные затраты на составление расписаний и облегчит работу сотрудникам учебного отдела образовательного учреждения.

Совсем не сложно составлять расписания для маленьких групп с несколькими дисциплинами. Более сложным является составление расписания для крупных вузов, для которых специфично большое число групп на каждом курсе обучения, существенное большое количество различных дисциплин, наличие различных типов аудиторий (лекционные, практические, лабораторные), различные направления обучения, большое количество преподавателей.

Автоматизация процесса формирования учебных занятий позволяет:

* учитывать различные требования и условия, предъявляемых к расписанию;
* правильно формировать расписание для получения наилучшего и гибкого результата;
* осуществить критериальный или оптимизационный подход к составлению расписания;
* уменьшить временные затраты на формирование расписания.

Качество подготовки специалистов в образовательной организации и эффективное использование научно-педагогического потенциала напрямую зависит от уровня организации учебного процесса. Одно из частей этого процесса является расписание учебных занятий – регламентирует трудовой ритм, напрямую влияет на творческую отдачу преподавателя, исходя из этого, расписание занятий можно расценивать как фактор оптимизации использования ограниченных ресурсов – преподавательского состава, его времени, а также аудиторного фонда.

Расписание учебных занятий является основным календарным планом системы планирования учебно-воспитательной работы образовательной организации, от которой зависит своевременное и качественное выполнение учебных планов и образовательных программ.

Проблему составления расписания следует воспринимать, не только как трудоёмкий процесс, объект автоматизации с использованием ЭВМ, требующий особого внимания, но и как механизм оптимального управления.

Поскольку при составлении расписания возникает большое количество факторов, практически невозможно учесть их все. Интересы участников учебного процесса достаточно многообразны, поэтому задача, которой является составление расписания занятий учебного процесса, является многокритериальной с варьирующимся множеством факторов, влияющих на его конечный результат.

Решение таких задач, как правильно, осуществляется в два этапа:

* получение оптимального варианта, с точки зрения используемых критерием;
* последующая доработка человеком, с целью учёта неформализованных факторов.

Расписание занятий составляет учебная часть под руководством заместителя директора по учебной работе. В основе составления расписания лежат учебные планы, наличие материально-технической базы, количество групп и преподавателей.

На первом этапе задачи, при составлении расписания можно выделить шесть атрибутов:

* группа, в которой проводятся занятия;
* преподаватель, который проводит занятия;
* количество часов на изучение предмета;
* предмет, по которому проводится занятие;
* аудитория;
* календарное время проведения занятия.

Данные атрибуты должны учитываться при разработке системы электронного расписания для учебно-методического управления образовательных организаций.

Целью данной работы является разработка программного продукта, позволяющего составить расписание учебных занятий, учитывая множественное количество факторов, при необходимости иметь возможность внесения изменений в конечный результат, предоставленный программным продуктом. Автоматизация рабочего места сотрудника учебно-методического управления, за счёт ускорение процесса составления расписания, предоставления необходимых для работы отчётов.

Для достижения данной цели потребуется решить ряд задач: проанализировать предметную область – процесс составления расписания занятий в образовательной организации; выработать требования к программному продукту и построить модель предметной области; на основании требований и модели, выбрать подходящее для решения задачи языки программирования и инструменты; реализовать разработку и составить необходимую документацию.

Объектом исследования работы является процесс работы сотрудника учебно-методического управления в обязанности которого входит составления расписания учебных занятий ЧУ ПОО «Краснодарский техникум управления, информатизации и сервиса».

Предметом исследования выступают процессы составления расписания, факторы влияющие на эти процессы, программные продукты, обеспечивающие составление расписания учебных занятий.

Структурно работа состоит из введения, много глав, заключения и двух приложений.

Краткое описание глав.

1 Исследование предметной области

* 1. ЧУ ПОО «Краснодарский техникум управления, информатизации и сервиса»

Частное учреждение – профессиональная образовательная организация «Краснодарский техникум управления информатизации и сервиса» был учрежден в 1998 году по решению собрания учредителей Института маркетинга и социально-информационных технологий как негосударственное образовательное учреждение «Финансово-экономический колледж ИМСИТ» с целью создания системы непрерывного профессионального образования.

В 2002 году колледж был переименован в Негосударственное образовательное учреждение среднего профессионального образования «Колледж управления, информатики и сервиса Академии маркетинга и социально-информационных технологий» (сокращенное наименование «НОУ СПО КУИС Академии ИМСИТ»).

В настоящее время в связи с принятием Федерального закона от 29.12.2014 №273-ФЗ «Об образовании в Российской Федерации» колледж был переименован в Частное учреждение — профессиональная образовательная организация «Краснодарский техникум управления информатизации и сервиса».

Как и в любом учебном заведении, в условиях реализации ФГОС СПО при модульной разработке учебного процесса рекомендуется использовать метод построения занятий по типу – семестровое, то есть планирования отдельно каждой недели семестра. Семестровое расписание реализует следующие задачи:

* установить оптимальную последовательность в изучении дисциплин, профессиональных модулей без учёта кратности часов количеству недель в семестре;
* рационально планировать в течение семестра проведение зачетов, экзаменом, контрольных работ, практических и лабораторных занятий, учебных и производственных практик, а также сдачу курсовых работ.
  1. Система электронного расписания занятий

Проблема в области решения задачи по составлению расписания является предметом научных исследований с середины прошлого века. Область применения обширна и включает в себя различные сферы человеческой деятельности: транспортные перевозки, обслуживание, промышленность, образование и т.д. На практике возникает множество задач, которые невозможно эффективно разрешить путём полного перебора. Для большинства моделей теории расписаний, получение оптимального расписания является трудноразрешимой задачей, а решение на практике показывают ещё большую сложность, так как данные решения должны удовлетворять многочисленным, а зачастую и конфликтующим между собою, условиям, как организационного, так и психологического характера. Выходом из данного положения является отказ от подхода, когда пригодным считается идеальное решение, так как на практике добиться его к сожалению, невозможно.

Качественный и количественный рост сферы среднего профессионального образования требует нового подхода к решению задач в учебно-методической, научной и хозяйственной деятельности. Этот подход с недавних пор находит свою реализацию в применение современных средств вычислительной техники и математических методов в управлении профессиональным образованием.

В современном мире всё большее распространение получают различные системы автоматизации процессов профессиональной деятельности, которые всего выполнялись вручную. Например, системы приятия решений в маркетинге, экспертные системы, системы тестирования, прогнозирующие системы в различных областях науки, техники и торговли. К таким же процессам относится и составление расписания занятий, которое до сих пор во многих учебных заведениях разрабатывается вручную на основе многолетнего опыта. Современные IT-технологии предлагают средства, позволяющие наилучшим образом организовать и автоматизировать любой процесс, в том числе и учебный.

Планирование расписания учебных занятий – это задача разработку расписания комбинаторного характера, особенностью которой является огромная размерность и наличие большого числа ограничений в сложной форме. Фактический, в настоящее время, не существует универсальных методов, которые могли бы решить такие задачи. Прямое применение математической (классической) теории расписания к задаче составления учебных занятий не представляется возможным. Тем не менее, есть ряд эвристических и переборных методов, которые поддаются программированию.

Существует мнение, что опытный специалист отвечающий за составление расписание сможет составить его таким образом, что оно будет отвечать всем интересам учебного процесса и научно-методической жизни образовательной организации. Однако с этим нельзя согласиться. Ручное решение данной задачи требует больших затрат времени, высокой квалификации специалиста, в тоже время результат такого решения часто получается далёк от оптимального. После ввода исходной информации требуется её согласование, в то время как на этапе анализа этих данных, может сложится ситуация, когда получение требуемого расписания может быть невозможна. Во время разработки расписания также может возникнуть тупиковая ситуация. Все эти факторы могут потребовать изменения исходных данных и ослабления ограничений, и в данной ситуации без человека не обойтись.

Без внесения изменений в расписание составленной авторизированной системой, данное расписание не будет иметь практической ценности. В этом плане очень важной является поддержка данного процесса автоматизированными методами и процедурами. Основное преимущество в автоматизации составления расписания занятий заключается в устранение массы рутинной работы, такой как:

* поиск вариантов внесения очередных данных в расписание;
* проверка выполнения требований;
* поиск случайных ошибок в готовом расписание;
* оформление расписания на бумаге в виде различных таблиц (для обучающихся, для преподавателей, групп, покабинетного);
* уменьшение затрат времени связанных с составлением расписания занятий.

Компьютер в данном случае выступает инструментом, существенно увеличивающим возможности человека в данном рабочем процессе, так как человек не в состоянии переработать и проанализировать такое же большое количество вариантов и факторов, влияющих на них, как компьютер.

В последние годы осуществляются множественные попытки совершенствования учебно-методических и научных процессов в образовательной организации путём построения алгоритмов оптимизации задач планирования учебной работы с использованием вычислительной техники и использования программного обеспечения от Корпорации Microsoft, в данном случае Microsoft Excel. Анализ состояния этих разработок позволяет сделать следующие выводы:

* Разработка и внедрение образовательными организациями автоматизированных систем управления осуществляется в инициативном, свободном порядке и эти работы, как правило, направлены на решение конкретных проблем. Разобщенность групп исследователей, разработчиков программах продуктов привело к большому множеству систем, направленных на разработку расписания занятий, рассчитанных на обслуживание только конкретной образовательной организации.
* Многие системы возлагают на специалиста отвечающего за разработку расписания всю ответственность за учёт реальных требований. В частности, учёт требований преподавателей, ограничения на количество проводимых занятий в день, в неделю, а также остальных факторов, влияющих на оптимальное расписание занятий – все эти и многие другие рутинные задачи в таких системах приходится решать специалисту самостоятельно, чаще всего методами перебора, что конечно же сказывается на качестве и временных затрат.
* Имеющиеся программы имеют весьма неудобный интерфейс для ввода исходных данных и редактирования полученного результата.

При разработке алгоритмов автоматизированной системы управления составлением расписания занятий остро стоит проблема в создании универсальных алгоритмов, учитывающих специфику условий всех факторов и каждой конкретной задачи. Такие алгоритмы должны быть достаточно гибкими, то есть без существенного их изменения можно было бы включать и исключать, а также добавлять новые требования для системы разработки расписания. Однако попытки решать такую задачу каким-либо одним универсальным алгоритмом на данный момент не представляется возможным. Алгоритмы, решающие широкий класс задач, не в состоянии дать той эффективности, которую обеспечивают конкретные, адаптированные с учётом конкретных условий алгоритмы.

Для систем разработки расписания занятий характера сильная зависимость от специфики конкретной образовательной организации уже на уровне математических моделей и исходных данных, что в значительной мере затрудняет процесс внедрения и использования типовых систем. Систему, созданную в одной образовательной организации, обычно без изменений и каких-либо доработок невозможно эффективно использовать в другой. К тому же многие системы создавались достаточно давно и не могут затронуть весь спектр изменений в образовательном процессе изменившийся в связи с нововведениями в образовательные стандарты РФ, поэтому не в состоянии эффективно решать поставленные перед ними задачи.

На данный момент в исследуемой образовательной организации используются процессы составления расписания, которые можно представить, как полуавтоматические, так как большую часть этих процессов специалист выполняет в ручном режиме. Данный процесс можно сопоставить следующему алгоритму действий: происходит обработка большого количества входной информации, что способствует внушительном затратам времени, а также не исключает человеческий фактор, после анализа этих данных, и составления, так называемых набросков, данные вводятся в заранее подготовленные таблицы Microsoft Excel, после чего происходит вывод на печать.

2 Обзор существующих продуктов

К настоящему времени разработан ряд программных продуктов выполняющие автоматизацию процесса разработки расписания занятий образовательной организации.

Из ряда разработанных программных продуктов стоит выделить наиболее популярные и которые в той или иной мере заслужили признания в учебных процессах образовательных организаций:

* «Экспресс-расписание Колледж»;
* «Ректор-Колледж»;
* Конфигурация «1С: Автоматизированное составление расписания. Колледж».

2.1 «Экспресс-расписание Колледж»

Экспресс-расписание Колледж – программа для автоматизации составления расписания учебных занятий в колледжах, техникумах и профессиональных лицеях. Программа автоматически составляет основное расписание, позволяет вести ежедневные изменения расписания, учет выполненных часов, формирует разнообразные отчёты.

Из описания к данному программному продукту можно выделить следующее:

* автоматическое составление основного расписание на 1 (2, 3 или 4) недели;
* ведение текущего расписания и возможность ежедневной замены уроков, преподавателей, аудиторий;
* составление расписания для нескольких смен, а также потоков с разными сроками начала и окончания обучения;
* работа с подгруппами;
* планирование отсутствия и занятости учителей, аудиторий, групп;
* печать основного и текущего расписания, листа замены;
* выборка занятий для ведения журнала;
* учет норм СанПиН.

Также стоит выделить что данный программный продукт зарегистрирован в Министерстве образования, получил рекомендации Департамента образования, и внедрён более чем в 450 учебных заведений страны.

К плюсам данного программного продукта можно отнести большой функционал, отзывчивая техническая поддержка, обновления, большое количество обучающих видеоуроков.

К минусам данного программного продукта стоит отнести, сложность интеграции продукта в учебный процесс, удовлетворительное, с точки зрения восприятия, юзабилити, сложность и неудобство в управление.

Цена программного продукта на данный момент составляет 7990 рублей, что можно так же отнести к плюсу.

2.2 «Ректор-Колледж»

«Ректор-Колледж» - программа предназначена для составления расписания занятий в колледжах. Программа состоит из четырёх разделов: «Списки», «Нагрузки», «Расписание» и «Замены».

Из описания к данному программному продукту можно выделить следующее:

* составление расписание для любого учебного заведения;
* ручной и автоматический режим работы;
* расписание занятий можно сохранить в форматах Microsoft Word, Excel или HTML;
* лёгкость (работа практический на любом персональном компьютере под управлением системы Windows старше XP);

К плюсам данного программного продукта можно отнести лёгкость и простота освоение, небольшая требовательность к железу персонального компьютера.

К минуса данного программного продукта можно отнести меленький функционал, не затрагивающий как заявлено в описание всех аспектов учебного процесса, из этого следует, что данный программный продукт не подходит для любого учебного заведения, отсутствие обновлений, технической поддержки.

Представленный программный продукт не поддерживается с 2012 года, но из-за своей простоты и лёгкость всё же используется в некоторых образовательных организация. На данный момент данный программный продукт можно купить по цене от 7000 рублей.

2.3 Конфигурация «1С: Автоматизированное составление расписания. Колледж»

«1С: Автоматизированное составление расписания. Колледж» относится к самостоятельному программному продукту, который обладает возможностью решать такие задачи, как: автоматизированное составление учебного расписания, оперативное управление кабинетами в общеобразовательных учреждениях среднего профессионального образования.

Из описания к данному программному продукту можно выделить следующее:

* удобная форма «шахматка» для быстрой ручной модификации расписания перетаскиванием;
* составление расписаний в разрезе:
  + семестров;
  + отделений;
  + сценариев («пессимистичный», оптимистичный);
  + составление нескольких расписаний и выбор лучшего.
* учет пожеланий и возможностей преподавателей, групп студентов, помещений;
* загрузка справочников и учебных планов из Excel;
* оперативное изменение расписаний и составление «текущих» расписаний;
* компоновка расписаний по отделениям в расписания по колледжу;
* разграничение доступа подокументно.

К плюсам данного программного продукта можно отнести большой функционал, отличная техническая поддержка, обновления, простая интеграция в учебный процесс.

К минусам можно отнести цену на данный программный продукт которая составляет около 22 000 рублей, а также сложность в обучении и эксплуатации данного программного продукта.

3 Требования к программному продукту

3.1 Постановка задачи

Требуется обеспечить возможность ввода и редактирования необходимых для составления расписания занятий данных. Выходными данными должны служить необходимые для работы учебно-методического управления образовательной организации отчёты, а также расписания занятий.

Входными данными должно служить следующее:

* наименование специальностей;
* названия групп;
* наименование кабинетов;
* названия дисциплин;
* ФИО преподавателя;
* время работы преподавателя;
* сводная выписка из учебного плана.

Выходными данными должно служить следующее:

* расписание занятий;
* ещё отчёты;

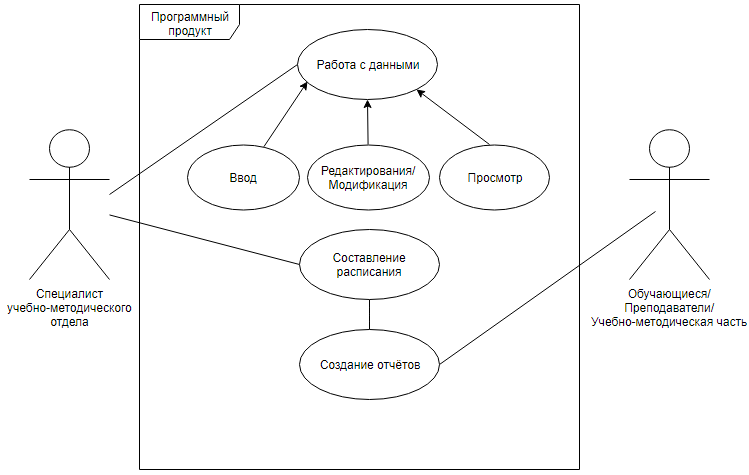


Рисунок 1 – Варианты использования программного продукта

Интерфейс пользователя для данного программного продукта имеет большую значимость. Должен быть реализован удобный, понятный и ненагруженный интерфейс пользователя. Пользователю должна быть понятна цепочка действий для получения необходимого результата.

Крайне желательно обеспечить возможность вывода расписания занятий в программный продукт Microsoft Excel.

3.2 Анализ требований

4 Техническое задание на разработку

4.1 Введение

4.1.1 Наименование программы

Наименование программы – «Системы электронного расписания для учебно-методического управления образовательных организаций». Далее по тексту «система».

4.1.2 Краткая характеристика области применения

Система предназначена к применению в профильных подразделениях на объектах Заказчика.

4.2 Назначения разработки

4.2.1 Функциональное назначение

Функциональным назначением системы является предоставление пользователю возможности внесения данных, с последующим, при необходимости, их модификацией и редактированием с целью получения оптимального, с точки зрения образовательной организации на которой используется система, расписанием занятий, а также необходимых отчётов для учебно-методической части.

4.2.2 Эксплуатационное назначение

Конечными пользователями программы должны являться сотрудники профильных подразделения объектов Заказчика.

4.3 Требования к программе или программному изделию

4.3.1 Требования к функциональным характеристикам

4.3.1.1 Требования к составу выполняемых функций

Система должна обеспечить следующие функции:

* ввод необходимых для составления расписания данных, их редактирование и модификацию;
* вывод на печать оптимального расписания занятий, а также необходимых для работы учебно-методической части отчётов.

4.3.1.2 Требования к организации входных данных

Входные данные должны быть организованы в виде ручного ввода в предоставленные системой поля и таблицы.

Система должна обеспечить удобный ввод, редактирование и модифицирование входных данных. А именно:

* наименование специальностей;
* названия групп;
* наименование кабинетов;
* названия дисциплин;
* ФИО преподавателя;
* время работы преподавателя;
* сводная выписка из учебного плана;

4.3.2.3 Требования к организации выходных данных

Система должна обеспечить вывод данных по средствам отчётов в формате \*.pdf и \*.elxs. А именно:

* расписание занятий;
* и ещё отчёты;

4.3.1.4 Требования к временным характеристика

Требования к временным характеристикам системы не предъявляются.

4.3.2 Требования к надёжности

4.3.2.1 Требования к обеспечению надежного (устойчивого) функционирования программы

Надежное (устойчивое) функционирование системы должно быть обеспеченно выполнением совокупности организационно-технически х мероприятий, перечень которых приведен ниже:

* организацией бесперебойного питания технических средств;
* использованием лицензионного программного обеспечения;
* регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. «Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
* регулярным выполнением требования ГОСТ 51188-98 «Защита информации. Испытания программных средств на наличие компьютерных вирусов».

4.3.2.2 Время восстановления после отказа

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 5 минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

4.3.2.3 Отказы из-за некорректных действий оператора

Отказы системы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему административных привилегий.

4.3.3 Условия эксплуатации

4.3.3.1 Климатические условия эксплуатации

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

4.3.3.2 Требования к видам обслуживания

Программа не требует проведения каких-либо видом обслуживания.

4.3.3.3 Требования к численности и квалификации персонала

Минимальное количество персонала, требуемого для работы программы, должно составлять не менее 2 штатных единиц: системный администратор и конечный пользователь программы – оператор.

Системный администратор должен иметь высшее профильное образование и сертификаты компании-производителя операционной системы. В перечень задач, выполняемых системным администратором, должны входить:

* задача поддержания работоспособности технических средств;
* задачи установки (инсталляции) и поддержания работоспособности системных программных средств – операционной системы;
* задача установки (инсталляции) программы.

Конечный пользователь программы (оператор) должен обладать практическими навыками работы с графическим пользовательским интерфейсом операционной системы.

Персонал должен быть аттестован на II квалификационную группу по электробезопасности (для работы с конторским оборудованием).

4.3.4 Требования к составу и параметрам технических средств

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий в себя:

* процессор Intel Pentium или аналогичный с тактовой частотой не менее 1,8 ГГц;
* оперативную память объёмом не менее 1 ГиБ;
* жёсткий диск или аналогичное запоминающее устройство с объёмом свободной памяти не менее 1 ГиБ.

4.3.5 Требования к информационной и программной совместимости

4.3.5.1 Требования к информационным структурам и методам решения

Требования к информационным структурам (файлов) на входе и выходе, а также к методам решения не предъявляются.

4.3.5.2 Требования к исходным кодам и языкам программирования

Требования к исходным кодам и языкам программирования не предъявляются.

4.3.5.3 Требования к программным средствам, используемым программой

Системные программные средства, используемые программой, должны быть предоставлены лицензионной локализованной версией операционной системы семейства Microsoft Windows, не ниже XP SP3, а также набор компонентов Microsoft Access Database Engine 2016 Redistributable.

Данный набор компонентов облегчает передачу данных между существующими файлами Microsoft Office, такими как Microsoft Office Access (\*.mdb и \*.accdb файл) файлами и Microsoft Office программы Excel (\*.xls,\*.xlsx и \*.xlsb) файлами и другими источниками данных, таких как Microsoft SQL Server. Также в состав набора входит поддержка подключение к существующим текстовым файлам. ODBC и OLEDB драйверы.

4.3.5.4 Требования к защите информации и программ

Требования к защите информации и программ не предъявляются.

4.3.6 Требования к маркировке и упаковке

Программа поставляется в виде программного изделия на дистрибутивном (внешнем оптическом) носители (компакт-диске).

4.3.6.1 Требования к маркировке

Программное изделие должно иметь маркировку с обозначением наименования разработчика, наименования продукта, номера версии, даты изготовления.

Маркировка должна быть нанесена на программное изделие в виде наклейки, выполненной полиграфическим или любым другим способом.

4.3.6.2 требования к упаковке

Упаковка программного изделия должна осуществляется в упаковочную тару предприятия-изготовителя.

4.3.6.3 Условия упаковывания

Упаковка программного изделия должна проводиться в закрытых вентилируемых помещениях при температуре от 15 до 50 °C и относительной влажности воздуха не более 80 % при отсутствии агрессивных примесей в окружающей среде.

4.4 Требования к программной документации

4.4.1 Предварительный состав программной документации

Состав программной документации должен включать в себя:

* техническое задание;
* руководство по установке;
* руководство системного администратора;
* руководство оператора;

4.5 Технико-экономические показатели

Ориентировочная экономическая эффективность не рассчитывается.

Предполагаемое число использования программы в год – 182 сеанса работы на одном рабочем месте.

4.5.1 Экономические преимущества разработки

Экономические преимущества разработки не рассчитываются.

4.6 Порядок контроля и приемки

4.6.1 Виды испытаний

Контроль и приёмка разработки осуществляются на основе испытаний контрольно-отладочных примеров. При этом проверяется выполнение всех функций программы.

5 Модель предметной области

Центральными сущностями предметной области являются расписание занятий, аудиторный фонд, дисциплины, преподаватели, специальности, а также рабочая нагрузка. Главные сущности представлены на рисунке 2 в виде диаграммы классов. Данная диаграмма служит для представления статической структуры модели система, различными взаимосвязями между отдельными сущностями предметно области, такими, как объекты и подсистемы, а также описания их структуры, типов и отношений.

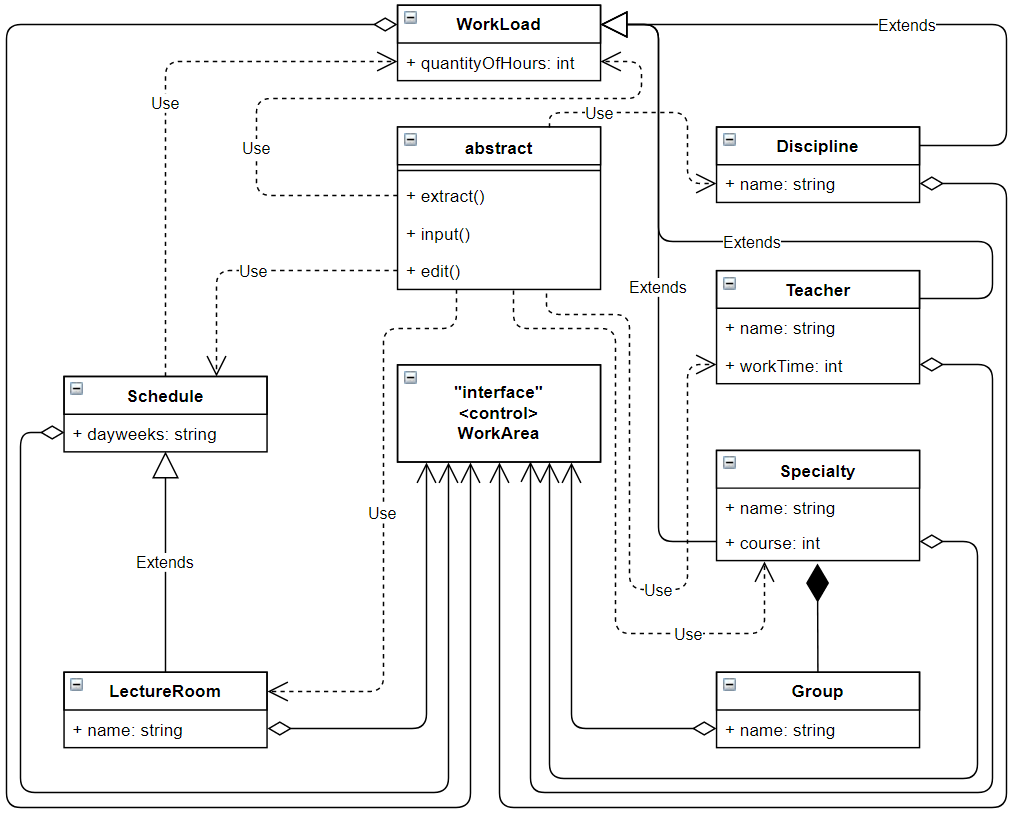


Рисунок 2 – Диаграмма классов основных сущностей

Предполагается в системе существование таких классов как: «Discipline» (Дисциплины), «Teacher» (Преподаватели), «Specialty» (Специальности) в которой, в свою очередь агрегатной ассоциацией относится класс «Group» (группы), «WorkLoad» (Рабочая нагрузка), «Shedule» (Расписание занятий), «LectureRoom» (аудиторный фонд), «abstract» и WorkArea (используемый интерфейс). Класс «abstract» является абстрактным: он имеет несколько методов необходимых для оптимальной работы и связан с классами «Discipline», «Teacher», «Specialty», «Group», «LectureRoom», «Shedule», «WorkLoad» отношением наследования. Класс «WorkArea» указывает на то, что данный класс отвечает за координацию действий, а также за интерфейс, предоставленный пользователю.

Классы «Teacher», «Specialty», «Group» должны предоставить данные для класса «WorkLoad», в свою очередь класс «WorkLoad» связан отношением наследования с классом «Schedule», которое на основе всех входных данных генерирует расписание, предоставляя его через интерфейс «WorkArea».

Динамические аспекты поведения системы отображаются на диаграммах кооперации и последовательности. На диаграмме кооперации поведение системы описывается на уровне отдельных объектов, которые обмениваются между собою сообщениями, чтобы достичь необходимой цели или реализовать некий вариант использования. Диаграмма кооперации представлена на рисунке 3.

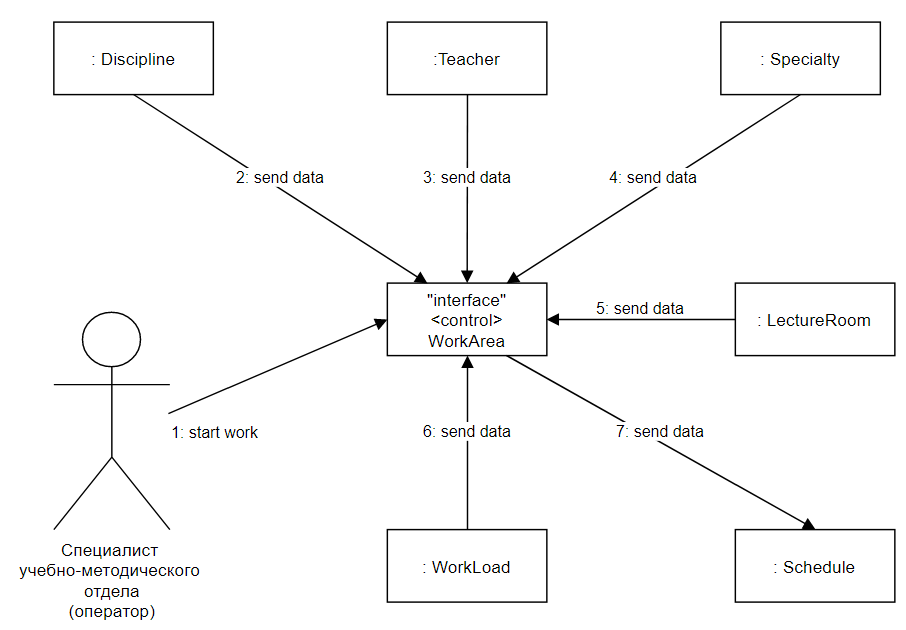


Рисунок 3 – Диаграмма кооперации

С помощью диаграмм последовательности можно описать полный процесс взаимодействий как своеобразный временной график «жизни» всей совокупности объектов, взаимодействующих между собою для реализации варианта использования системы, достижения цели или выполнения какой-либо задачи. Диаграмма последовательности представлена на рисунке 4.

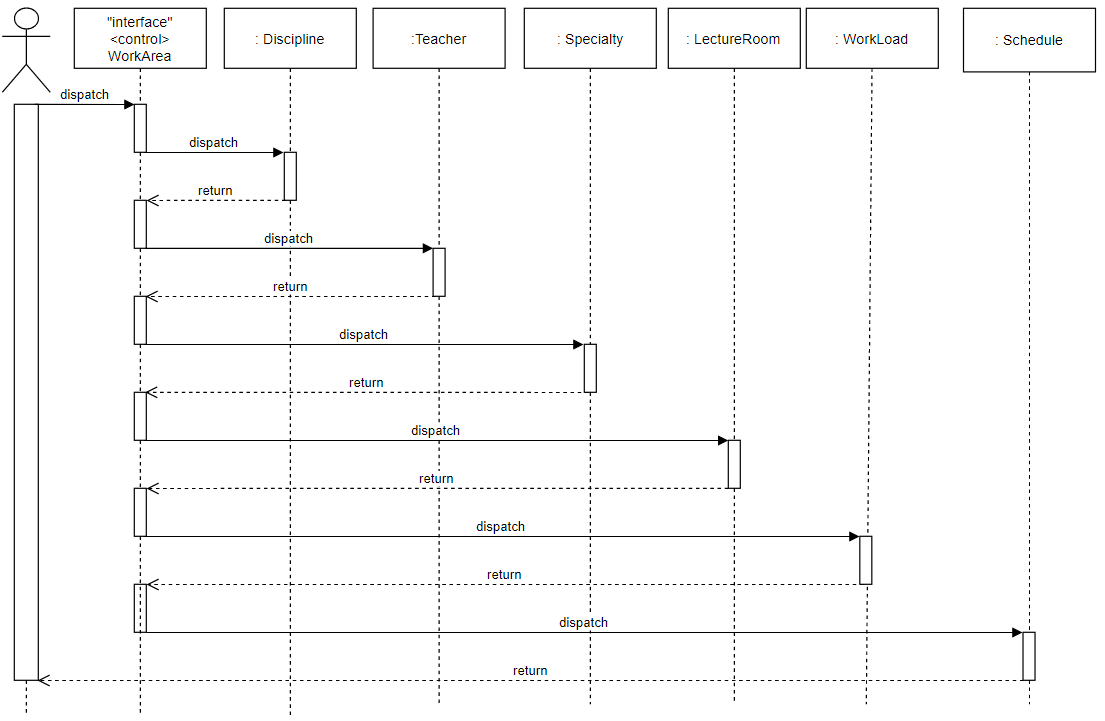


Рисунок 4 – Диаграмма последовательности

На диаграмме видно, что инициатором взаимодействия является Специалист учебно-методического отдела (оператор), который обладает фокусом управления на протяжении всей работы системы, постоянно контролируя её деятельность. С объектов «Discipline», «Teacher», «Specialty», «LectureRoom», «WorkLoad» собирается необходимая информация путем перехода фокуса управления от управляющего окна к соответствующему объекту и обратно. Когда необходима информация собрана, создаётся объект «Schedule» получающий временный фокус управления.

Далее требуется описать процесс изменения состояния системы или её подсистемы при реализации всех вариантов использования. При этом изменения состояний отдельных элементов системы может быть вызвано внешними воздействиями со стороны других элементов или извне системы. Именно для описания реакции системы на подобные внешние воздействия потребуется использовать диаграмму состояний. Данная диаграмма состояний представлена на рисунке 5.

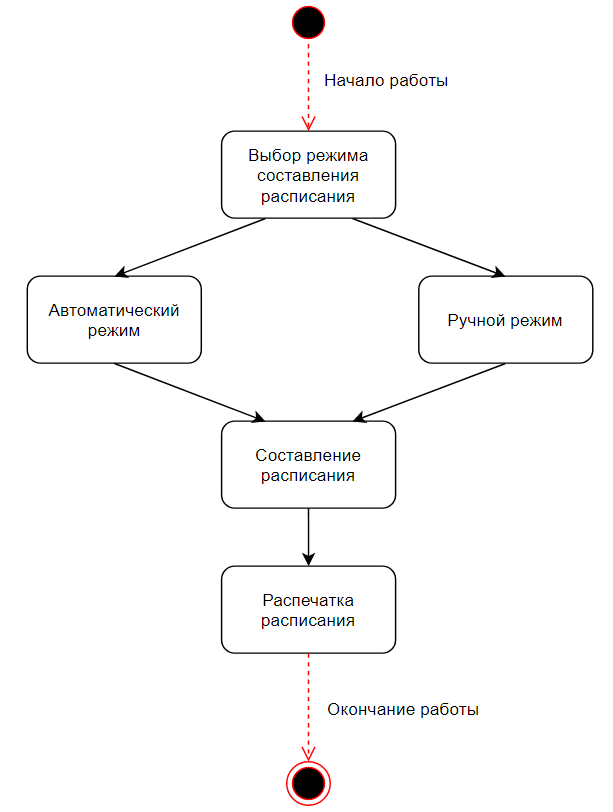


Рисунок 5 – Диаграмма состояний.

Для моделирования процесса выполнения операций используется так называемая диаграмма деятельности. Каждое состояние на этой диаграмме соответствует выполнению некоторой элементарной операции, а переход в следующее состояние выполняется только после завершения операций в предыдущем состоянии. Графически диаграмма деятельности представляется в форме графа деятельности, вершинами которого являются состояния действия, а указывающими стрелками – переходы от одного состояния в другому. Таким образом, диаграмму деятельности можно считать частным случаем диаграмм состояний. Данная диаграмма деятельности представлена на рисунке 6.

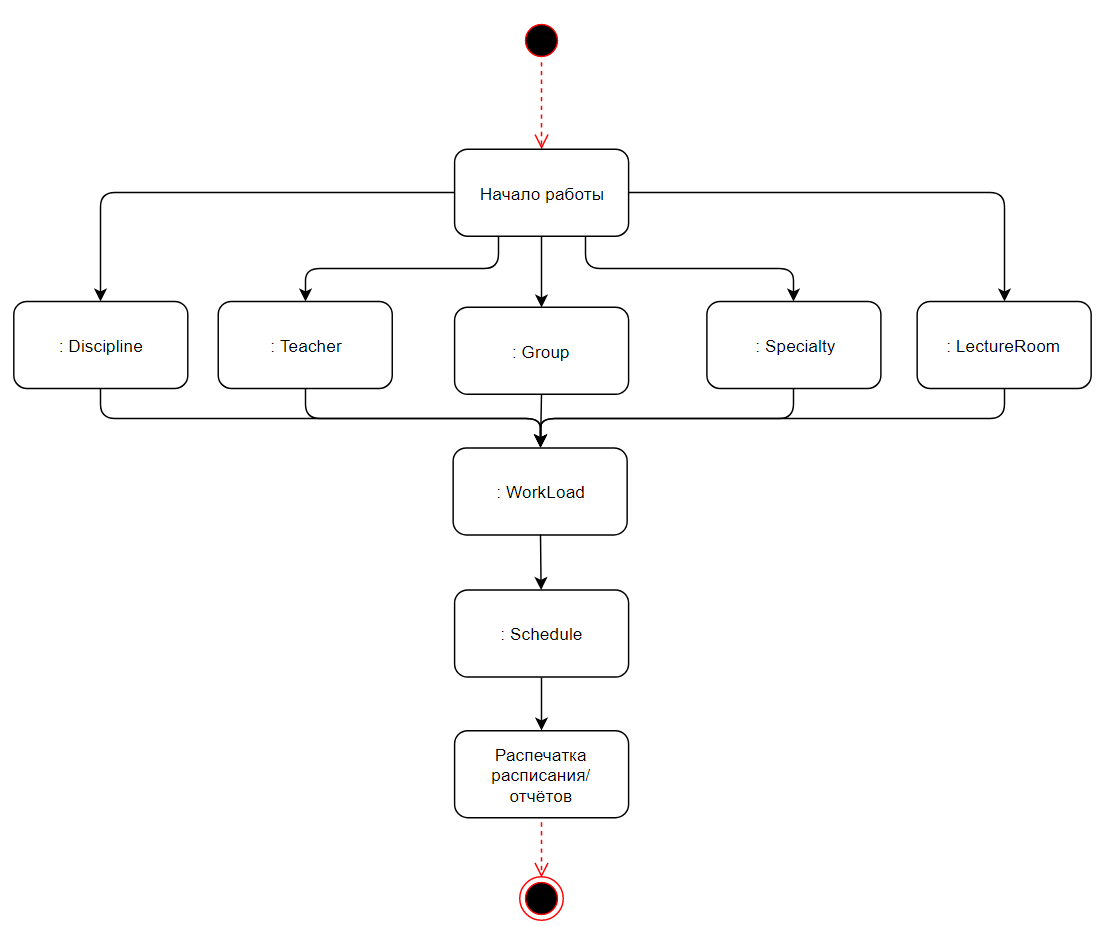


Рисунок 6 – Диаграмма деятельности.

Для создания конкретной физической системы необходимо некоторым образом реализовать все элементы логического представления в конкретные материальные сущности. Для описания таких реальных сущностей предназначен другой аспект модельного представления, а именно – физического представление модели. Для физического представления моделей систем будет использована так называемая диаграмма компонентов, которая позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами. Данная диаграммы, как правило, строится непосредственно перед компьютерной реализацией систем. Диаграмма компонентов представлена на рисунке 7.

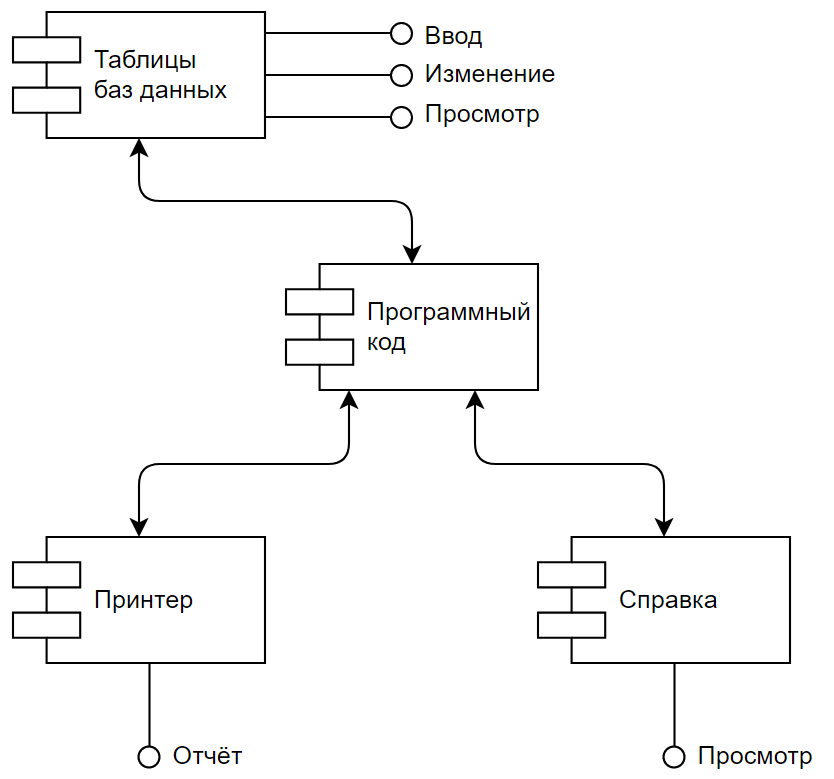


Рисунок 7 - Диаграмма компонентов.

6 Выбор метода и инструментов разработки

6.1 Выбор среды программирования

Microsoft Visual Studio — это набор инструментов для создания программного обеспечения: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов и сбора данных телеметрии по использованию. Эти инструменты предназначены для максимально эффективной совместной работы; все они доступны в интегрированной среде разработки (IDE) Visual Studio.

Visual Studio можно использовать для создания различных типов приложений, от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных, а именно можно создавать:

* приложения и игры, которые выполняются не только на платформе Windows, но и на Android и iOS;
* веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;
* приложения для самых разных платформ и устройств, включая, но не ограничиваясь: Office, Sharepoint, Hololens, Kinect и "Интернета вещей";
* игры и графические приложения для разных устройств Windows, включая Xbox, с поддержкой DirectX.

По умолчанию Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и Visual Basic. Visual Studio хорошо работает и интегрируется со сторонними приложениями, например Unity и Apache Cordova, с помощью расширений «Набор средств Visual Studio для Unity» и инструментов Visual Studio для Apache Cordova соответственно. Вы также можете самостоятельно расширить Visual Studio, создав собственные инструменты для выполнения специализированных задач.

Microsoft Office Access или просто Microsoft Access — реляционная система управления базами данных (СУБД) корпорации Microsoft. Входит в состав пакета Microsoft Office. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных. Благодаря встроенному языку VBA, в самом Access можно писать приложения, работающие с базами данных.

Основные компоненты MS Access:

* построитель таблиц;
* построитель экранных форм;
* построитель SQL-запросов (язык SQL в MS Access не соответствует стандарту ANSI);
* построитель отчётов, выводимых на печать.

Они могут вызывать скрипты на языке VBA, поэтому MS Access позволяет разрабатывать приложения и БД практически «с нуля» или написать оболочку для внешней БД.

Microsoft Jet Database Engine которая используется в качестве движка базы данных MS Access, является файл-серверной СУБД и потому применима лишь к приложениям, работающим с небольшими объёмами данных и при небольшом числе пользователей, одновременно работающих с этим данными. Непосредственно в Access отсутствует ряд механизмов, необходимых в многопользовательских базах данных, таких, например, как триггеры.

Microsoft Access Database Engine 2016 Redistributable - набор компонентов облегчает передачу данных между существующими файлами Microsoft Office, такими как Microsoft Office Access (\*.mdb и \*.accdb файл) файлами и Microsoft Office программы Excel (\*.xls,\*.xlsx и \*.xlsb) файлами и другими источниками данных, таких как Microsoft SQL Server. Также в состав набора входит поддержка подключение к существующим текстовым файлам. ODBC и OLEDB драйверы.

Встроенные средства взаимодействия MS Access со внешними СУБД с использованием интерфейса ODBC снимают ограничения, присущие Microsoft Jet Database Engine. Инструменты MS Access, которые позволяют реализовать такое взаимодействие, называются «связанные таблицы» (связь с таблицей СУБД) и «запросы к серверу» (запрос на диалекте SQL, который «понимает» СУБД).

На данный момент актуальной версией Microsoft Access является версия 2015 года: Microsoft Access 2016 (из офисного пакета приложений Microsoft Office 2016).

База данных – которая представляется в объективной форме называется совокупностью самостоятельных материалов, упорядоченная таким образом, чтобы эти хранящиеся материалы можно было найти и обработать с помощью компьютера или другого вычислительного устройства. Большое количество разработчиков указывают на серьезную часто встречающуюся ошибку, которая состоит в ошибочном применении названия «база данных» вместо термина «система управления базами данных», что указывают на большую необходимость разграничения данных понятий.

На сегодняшний день основные направления развития современной информационной технологии составляют базы данных (БД) и системы управления базами данных (СУБД), роль которых как совместного средства хранения, обработки и доступа к огромным объемам упорядоченных данных постоянно увеличивается. В данном случае существенным фактом является постоянное увеличение объемов хранимой информации, хранимой в базе данных, что в свою очередь влечет за собой требование увеличения производительных мощностей таких систем.

Стремительно увеличивается таким же образом в различных применениях спрос на удаленный доступ к хранимой информации. Данный факт особенно проявляется при организации процесса логического манипулирования информации в системах баз данных, основой которых являются созданные современные экспертные системы. Стремительное развитие необходимости использования баз данных выдвигает ряд требований к системе управления базами данных: поддержка огромного числа типов представляемых данных и манипуляций над ними; естественные и эффективные размещения в БД различных отношений между объектами рассматриваемых предметных областей; осуществление поддержки верности данных и создание дедуктивных БД; обеспечение целостности информации хранимой в БД в большом диапазоне различных предметных областей и операционных систем; осуществление удобного управления распределенными БД, включение неоднородных баз данных; заметное повышение надежности функционирования БД.

Традиционные базы данных являются пассивными. Информация часто помещается в базу данных пользователем или программным продуктом. Выборка объектов осуществляется опять-таки под воздействием внешних источников. Таким же образом под воздействием какого-либо внешнего источника информация меняет место своего нахождения в базе данных (например, переносится из исходной таблицы в другую). Бизнес-правила, используемые в содержимом базы данных (например, правила обновления определенного элемента данных, следующие за этим действия над предлагаемыми элементами как результат такого обновления), также управляются другим внешним источником. По-другому говоря, базы данных в традиционном представлении не являются активным элементов в информационных системах и вместо этого они выполняют организационную роль, которая направлена на обеспечение хранения данных. В последние годы эта роль все увеличивается, и важность концепции активных баз данных растет с каждым годом

6.2 Выбор языка программирования

Последнее время С и С++ являются наиболее используемыми языками для разработки коммерческих и бизнес приложений. Эти языки устраивают многих разработчиков, но в действительности не обеспечивают должной продуктивности разработки. К примеру, процесс написания приложения на С++ зачастую занимает значительно больше времени, чем разработка эквивалентного приложения, скажем, на Visual Basic. Сейчас существуют языки, увеличивающие продуктивность разработки за счет потери в гибкости, которая так привычна и необходима программистам на С/С++. Подобные решения являются весьма неудобными для разработчиков и зачастую предлагают значительно меньшие возможности. Эти языки также не ориентированы на взаимодействие с появляющимися сегодня системами и очень часто они не соответствуют существующей практике программирования для Web. Многие разработчики хотели бы использовать современный язык, который позволял бы писать, читать и сопровождать программы с простотой Visual Basic и в то же время давал мощь и гибкость C++, обеспечивал доступ ко всем функциональным возможностям системы, взаимодействовал бы с существующими программами и легко работал с возникающими Web стандартами.

Учитывая все подобные пожелания, Microsoft разработала новый язык - C#. В него входит много полезных особенностей - простота, объектная ориентированность, типовая защищенность, "сборка мусора", поддержка совместимости версий и многое другое. Данные возможности позволяют быстро и легко разрабатывать приложения, особенно COM+ приложения и Web сервисы. При создании C#, его авторы учитывали достижения многих других языков программирования: C++, C, Java, SmallTalk, Delphi, Visual Basic и т.д. Надо заметить что по причине того, что C# разрабатывался с чистого листа, у его авторов была возможность (которой они явно воспользовались), оставить в прошлом все неудобные и неприятные особенности (существующие, как правило, для обратной совместимости), любого из предшествующих ему языков. В результате получился действительно простой, удобный и современный язык, по мощности не уступающий С++, но существенно повышающий продуктивность разработок.

Очень часто можно проследить такую связь - чем более язык защищен и устойчив к ошибкам, тем меньше производительность программ, написанных на нем. К примеру рассмотрим две крайности - очевидно это Assembler и Java. В первом случае вы можете добиться фантастической быстроты своей программы, но вам придется очень долго заставлять ее работать правильно не на вашем компьютере. В случае же с Java - вы получаете защищенность, независимость от платформы, но, к сожалению, скорость вашей программы вряд ли совместима со сложившимся представлением о скорости, например, какого-либо отдельного клиентского приложения (конечно существуют оговорки - JIT компиляция и прочее).

В C#, как в несомненно современном языке, также существуют характерные особенности для обхода возможных ошибок. Например, помимо упомянутой выше "сборки мусора", там все переменные автоматически инициализируются средой и обладают типовой защищенностью, что позволяет избежать неопределенных ситуаций в случае, если программист забудет инициализировать переменную в объекте или попытается произвести недопустимое преобразование типов. Также в C# были предприняты меры для исключения ошибок при обновлении программного обеспечения. Изменение кода, в такой ситуации, может непредсказуемо изменить суть самой программы. Чтобы помочь разработчикам бороться с этой проблемой C# включает в себя поддержку совместимости версий (vesioning). В частности, в отличии от C++ и Java, если метод класса был изменен, это должно быть специально оговорено. Это позволяет обойти ошибки в коде и обеспечить гибкую совместимость версий. Также новой особенностью является native поддержка интерфейсов и наследования интерфейсов. Данные возможности позволяют разрабатывать сложные системы и развивать их со временем.

В C# была унифицирована система типов, теперь вы можете рассматривать каждый тип как объект. Несмотря на то, используете вы класс, структуру, массив или встроенный тип, вы можете обращаться к нему как к объекту. Объекты собраны в пространства имен (namespaces), которые позволяют программно обращаться к чему-либо. Это значит что вместо списка включаемых файлов заголовков в своей программе вы должны написать какие пространства имен, для доступа к объектам и классам внутри них, вы хотите использовать. В C# выражение using позволяет вам не писать каждый раз название пространства имен, когда вы используете класс из него. Например, пространство имен System содержит несколько классов, в том числе и Console. И вы можете писать либо название пространства имен перед каждым обращением к классу, либо использовать using как это было показано в примере выше.

Важной и отличительной от С++ особенностью C# является его простота. К примеру, всегда ли вы помните, когда пишите на С++, где нужно использовать "->", где "::", а где "."? Даже если нет, то компилятор всегда поправляет вас в случае ошибки. Это говорит лишь о том, что в действительности можно обойтись только одним оператором, а компилятор сам будет распознавать его значение. Так в C#, оператор"->" используется очень ограничено (в unsafe блоках, о которых речь пойдет ниже), оператор "::" вообще не существует. Практически всегда вы используете только оператор "." и вам больше не нужно стоять перед выбором.

Еще один пример. При написании программ на C/С++ вам приходилось думать не только о типах данных, но и о их размере в конкретной реализации. В C# все упрощено - теперь символ Unicode называется просто char (а не wchar\_t, как в С++) и 64-битное целое теперь - long (а не \_\_int64). Также в C# нет знаковых и беззнаковых символьных типов.

В C#, также как и в Visual Basic после каждого выражения case в блоке switch подразумевается break. И более не будет происходить странных вещей если вы забыли поставить этот break. Однако если вы действительно хотите чтобы после одного выражения case программа перешла к следующему вы можете переписать свою программу с использованием, например, оператора goto.

Многим программистам (на тот момент, наверное, будущим программистам) было не так легко во время изучения C++ полностью освоиться с механизмом ссылок и указателей. В C# (кто-то сейчас вспомнит о Java) нет указателей. В действительности нетривиальность указателей соответствовала их полезности. Например, порой, трудно себе представить программирование без указателей на функции. В соответствии с этим в C# присутствуют Delegates - как прямой аналог указателя на функцию, но их отличает типовая защищенность, безопасность и полное соответствие концепциям объектно-ориентированного программирования.

Хотелось бы подчеркнуть современное удобство C#. Когда вы начнете работу с C#, а, надеюсь, это произойдет как можно скорее, вы увидите, что довольно большое значение в нем имеют пространства имен. Уже сейчас, на основе первого примера, вы можете судить об этом - ведь все файлы заголовков заменены именно пространством имен. Так в C#, помимо просто выражения using, предоставляется еще одна очень удобная возможность - использование дополнительного имени (alias) пространства имен или класса.

Современность C# проявляется и в новых шагах к облегчению процесса отладки программы. Традиционным средством для отладки программ на стадии разработки в C++ является маркировка обширных частей кода директивами #ifdef и т.д. В C#, используя атрибуты, ориентированные на условные слова, вы можете куда быстрее писать отлаживаемый код.

В наше время, когда усиливается связь между миром коммерции и миром разработки программного обеспечения, и корпорации тратят много усилий на планирование бизнеса, ощущается необходимость в соответствии абстрактных бизнес процессов их программным реализациям. К сожалению, большинство языков реально не имеют прямого пути для связи бизнес логики и кода. Например, сегодня многие программисты комментируют свои программы для объяснения того, какие классы реализуют какой-либо абстрактный бизнес объект. C# позволяет использовать типизированные, расширяемые метаданные, которые могут быть прикреплены к объекту. Архитектурой проекта могут определяться локальные атрибуты, которые будут связанны с любыми элементами языка - классами, интерфейсами и т.д. Разработчик может программно проверить атрибуты какого-либо элемента. Это существенно упрощает работу, к примеру, вместо того чтобы писать автоматизированный инструмент, который будет проверять каждый класс или интерфейс, на то, является ли он действительно частью абстрактного бизнес объекта, можно просто воспользоваться сообщениями основанными на определенных в объекте локальных атрибутах.

7 Реализация разработки

7.1 Построение моделей данных

На этапе проектирования системы, проанализировав предметную область становиться возможным построение даталогической модели. Даталогическая модель базы данных является моделью логического уровня и представляет собою отображение логических связей между элементами данных. Суть этапа даталогического проектирования заключается в отображении инфологической модели предметной области базы данных на структурные и процедурные возможности. Они предоставляют: тип структуры данных, например, реляционная модель, выбранный тип СУБД, которая эту модель данных поддерживает, технологию и средства прикладного программировании.

Для каждого локального представления указывается:

* определение типов сущностей;
* определение типов связей;
* определение атрибутов и связывание их с типами сущностей и типами связей;
* определение доменов атрибутов;
* определение атрибутов, являющихся первичными, альтернативными или внешними ключами;
* определение ограничений целостности.

Соответствующие спецификации должны быть документированы. Анализ документации позволяет сформировать перечень атрибутов.

На основе представления о предметной области, для выявления информационных объектов необходимо осуществить анализ ее входной информации.

Входную информацию принято делить на условно-постоянную и оперативно-учетная информация.

К условно-постоянной информации относятся:

* план учебного расписания;
* списки отделений и факультетов;
* списки преподавателей;
* перечень дисциплин, аудиторий;
* время пары.

К оперативно-учетной информации относятся:

* списки курсов и групп;
* дата;
* количество студентов в группах.

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование сущности | Атрибуты |
| 1. | План расписания | Специальность  Группа  Тип недели  День недели  ФИО преподавателя  Дисциплина  Время  Аудитория |
| 2. | Списки преподавателей | № преподавателя  ФИО преподавателя |
| 3. | Перечень дисциплин | № дисциплины  Наименование дисциплины |
| 4. | Список специальностей | № специальности  Наименование специальности |
| 5. | Списки групп | № группы  Наименование группы |
| 6. | Списки аудиторий | № аудитории  Наименование аудитории |

Перечень первичных сущностей и их атрибутов представим в виде таблицы 1.

Таблица1 - Перечень первичных сущностей с атрибутами

Необходимо проанализировать каждый атрибут на наличие связи с другими атрибутами. Атрибут приобретает смысл тогда, когда он связан с другими, обладающими смысловым единством.

По ходу выполнения разработки, предстоит детализировать некоторые атрибуты, для упрощения и предупреждения неприятных последствий, а также создать некоторые вспомогательные информационные объекты для более интуитивного и детализированного хранения данных.

Среди выявленных атрибутов, описывающих сущность можно выделить один или несколько атрибутов, которые определяют экземпляр сущности. Такой атрибут будет называться ключом. Между ключом и другими атрибутами существует функциональная зависимость. Результат занесем в таблицу 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Информационные объекты | Ключ | Атрибуты |
| 1 | Аудиторный фонд | \* | ID кабинета  Наименование кабинета |
| 2 | Список дисциплин | \* | ID дисциплины  Наименование дисциплины |
| 3 | Кабинеты дисциплин | \*  \* | ID дисциплины  ID кабинета  Рейтинг кабинета  Курс |
| 4 | Список преподавателей | \* | ID преподавателя  ФИО преподавателя |
| 5 | Рабочее время преподавателя | \* | ID преподавателя  Номер занятия  День недели (1..6) |
| 6 | Дисциплины преподавателя | \*  \* | ID преподавателя  ID дисциплины |
| 7 | Список специальностей | \* | ID специальности  Наименование специальности |
| 8 | Список групп | \* | ID специальности  Наименование группы |
| 9 | Специальность – курс – преподаватель дисциплины | \*  \*  \*  \* | ID специальности  ID дисциплины  ID преподавателя  Курс |
| 10 | Учебная (рабочая) нагрузка | \*  \*  \*  \* | ID специальности  Курс  ID дисциплины  Семестр  Количество часов |
| 11 | Календарь | \*  \* | День недели  Тип недели |
| 12 | Семестр | \* | ID семестра  Количество недель |
| 13 | Расписание занятий |  | Тип недели  День недели  Номер занятия  Специальность  Группа  Аудитория  ФИО преподавателя |

Таблица 2 – Информационные объекты и их атрибуты.

Проанализировав предметную область, выделив все атрибуты, обозначив ключи, мы пришли к тому, что получили 13 информационных объектов:

* аудиторный фонд;
* список дисциплин;
* кабинеты дисциплин;
* список преподавателей;
* рабочее время преподавателя;
* дисциплины преподавателя;
* список специальностей;
* список групп;
* специальность – курс – преподаватель дисциплины;
* учебная (рабочая) нагрузка;
* календарь;
* семестр;
* расписание занятий.

Следующим этапом разработки является определение связей между сущностями.

Аудиторный фонд  Дисциплина кабинета

Для связи аудиторного фонда со списком дисциплин, потребуется дополнительный информационный объект, название которого «Дисциплина кабинета», так как возможна ситуация, когда кабинет может быть привязан к определённой дисциплине, также в данном объекте был внесён атрибут «Рейтинг кабинета» для определения приоритета аудитории для определённой дисциплины при составлении расписания. В данном случае тип связи между информационным объектом «Аудиторный фонд» и «Дисциплины кабинетов» будет один-ко-многим. Связь между этими объектами будет обеспечиваться с помощью атрибута ID аудитории.

Список дисциплина  Дисциплина кабинета

Тип связи между информационными объектами «Дисциплина» и «Дисциплина кабинета», будет осуществляться по такому же свойству, как и в случае со связью «Аудиторный фонд» и «Дисциплина кабинета», то есть один-ко-многим. Связь между этими объектами будет обеспечиваться с помощью атрибута ID дисциплины.

Список преподавателей  Рабочее время преподавателя

Для связи информационных объектов «Список преподавателей» и «Рабочее время преподавателя» будет использован тип один-ко-многим, так как у преподавателя могут быть разные рабочие дни, а также разные промежутки времени в эти дни, выходные, и т.д. Связь между этими объектами будет обеспечиваться с помощью атрибута ID преподавателя.

Список дисциплина  Дисциплина преподавателя

Для связи списка дисциплин со списком преподавателей, потребуется дополнительный информационный объект «Дисциплины преподавателя», так как возможна ситуация что один преподаватель может преподавать несколько дисциплин, или же у одной дисциплины может быть несколько преподавателей. В данном случае тип связи между информационным объектом «Дисциплина» и «Дисциплина преподавателя» будет один-ко-многим. Связь между этими объектами будет обеспечиваться с помощью атрибута ID дисциплины.

Список преподавателей  Дисциплина преподавателя

Тип связи между информационными объектами «Список преподавателей» и «Дисциплина преподавателя», будет осуществляться по такому же свойству, как и в случае со связью «Список дисциплин» и «Дисциплина преподавателя», то есть один-ко-многим. Связь между этими объектами будет обеспечиваться с помощью атрибута ID преподавателя.

Список специальностей  Список групп

Тип связи один-ко-многим, так как каждая специальность может иметь несколько групп. Связь между этими объектами будет обеспечиваться с помощью атрибута ID специальности.

Список специальностей

Список преподавателей Специальность – курс – преподаватель дисциплины

Список дисциплин

Стоит отметить и ситуацию, когда преподаватель может вести занятия только на определённом курсе определённой специальности, для этого был создан информационный объект «Специальность – курс – преподаватель дисциплины», где можно хранить данные с необходимым нам условием. Связь между объектами «Список специальностей», «Список преподавателей» и «Список дисциплин» обеспечивается с помощью соответствующих атрибутов ID специальности, ID преподавателя и ID дисциплины.

Для хранения данных о рабочей (учебной) нагрузки был создан объект «Учебная (рабочая) нагрузка». В рабочую нагрузку должны входить сведения о наименование специальности, курсе, дисциплине, семестре, и соответственно о количестве часов для данной дисциплины, для хранения этих данных были созданы соответствующие атрибуты и определена связь между подходящими сущностями, а именно ID дисциплины объекта «Список дисциплин» и ID специальности объекта «Список специальностей. Тип связи между объектами была определена как один-ко многим от объекта «Список дисциплин» и «Список специальностей» к объекту «Учебная (рабочая) нагрузка».

Тип связи один-к-одному, так как к сущности «Расписание» относятся индивидуальные атрибуты «Специальность» «Группа» «Аудитория» «ФИО преподавателя». Связь с сущностями осуществляется с помощью индивидуальных атрибутов.

Обозначив основные связи можно построить информационно-логическую модель.

Информационно логическая модель - модель данных отображающих предметную область в виде совокупности информационных объектов и структурных связей между ними. Представление данной модели представлено на рисунке 8.