Содержание:

ВВЕДЕНИЕ

Принципы рыночной экономики требуют от ее участников новых технологий менеджмента: приоритетным является стремление к экономической эффективности, высокая степень адаптации к динамизму рынка, который в наши дни отличается чрезмерной нестабильностью.

Достижения техники и инновации человеческой мысли приводят к тому, что любая, даже микро – организация становится все более сложной системой.

Это все создает условия для вовлечения в современные условия хозяйствования новые методы менеджмента, отражающие все сложности факторов среды функционирования компаний.

Одним из проявлений данных новшеств является применение системы управления компьютерными системами в процессе реализации политики управления, обеспечивающего принятие эффективных кратко- и долгосрочных управленческих решений.

Исследование проводилось в Университете «Университет».

Целью написания курсовой работы было практическое закрепление и углубление полученных теоретических знаний по вопросам вычислительной техники, информационных технологий и информационных систем, применяемых на предприятиях и в организациях, изучение программного, аппаратного и информационного обеспечения управляющих систем различного уровня и назначения.

В процессе выполнения курсовой работы были выполнены следующие задачи:

- рассмотрены основные компоненты, технологии и этапы проектирования информационных систем;

- выполнено ознакомление с основными направлениями деятельности учреждения и организационной структурой;

- рассмотрена структура отдела информационных технологий и автоматизации, а также изучены технологии проектирования;

- выполнено ознакомление с аппаратным и программным обеспечением, информационными технологиями предприятия и перспективами их внедрения;

- разработана информационная модель учреждения;

- разработана функциональная модель отдела по информационным технологиям.

ГЛАВА 1. ЯЗЫКИ ПРОГРАММИРОВАНИЯ: СУЩНОСТЬ, КЛАССИФИКАЦИЯ, СРАВНИТЕЛЬНАЯ ХАРАКТЕРИСТИКА, ОСНОВНЫЕ ЭТАПЫ РАЗРАБОТКИ

1.1 Сущность, классификация особенности современных языков программирования

Сущность программирования заключается в умении сотворения различных программ посредством специальных алгоритмов программирования. Алгоритм программирования связан с системой языков программирования, то есть определенной последовательности символов, необходимых для создания программных продуктов, при этом символы ясны исключительно автору изобретения.

В рамках языка программирования описывается система канонов, базисов, принципов, описывающих сам программный продукт, а также порядок выполнения действий, которые необходимо выполнять оператору при применении данного продукта.

На данный момент, начиная с зари компьютерной эры, сотворено не меньше 2 500 различных языков программирования, при этом данный список постоянно обновляется, расширяется, так как развитие технологий не стоит на месте. Можно отметить, что некоторые языки известны только единицам пользователей, некоторые становятся достоянием человеческой цивилизации. Одним из общих

правил создания программных продуктов является применение нескольких языков.

Сейчас насчитывается несколько основных классификаций языков программирования.

Самым известным критерием, по которому распределяются все существующие языки программирования, является набор определений и терминов, описывающих алгоритм, в соответствии с ним языки бывают низкого и высокого уровня.

В том случае когда язык применяет в основном алгоритм, созданный человеком, то такой язык считается высоким, а если все пишется технически, то язык низкого уровня.

К последней группе можно отнести машинные языки и языки кодирования символами, например, Автокод, Ассемблер. В данном случае составителями алгоритмов считаются команды, выполняемые роботом, они преобразованы в шифры мнемонического характера, операндами здесь выступают имена, состоящие из символов. Данные языки направлены на конкретный вид машин, то есть зависимы от некоторого типа компьютеров.

Направленные на выполнение компьютерами языки программирования – языки, порядок алгоритмов, модели, коды, шифры, реализация которых определяются возможностями и техническими характеристиками компьютеров, например, таких как состояния памяти, особенностей внутреннего алгоритма.

К наиболее развитым языкам, то есть высшим языкам назначают Фортран (дешифратор кодов создан в 50-е годы в компании IBM, он применяется для написания продуктов, реализующих определенные расчеты математического и ручного труда), Алгол, Кобол (сфера применения в основном в экономике и коммерческом деле), Паскаль, Бейсик (создан программистами в Дармуте), Си (создан в 1972 году), Пролог (он основан на применении математики и логических моделей).

Данные языки не являются зависимыми от определенного класса компьютеров, то есть они опираются не на порядок выполнения программ отдельным видом компьютеров, а на отдельную модель символов, отражающих некие алгоритмы. Недостатком языков высокого уровня является потребность в гораздо большем объеме памяти, а также более низкая скорость их реализации, чем для программ, созданных технически.

Так как программный продукт, созданный при помощи языка более высокого уровня, не может быть понят компьютером, машина принимает исключительно алгоритм машинного характера, то для успешной реализации применяют трансляторы, которые позволяют преобразовать коды.

Используются следующие типы трансляторов:

- интерпретатор, его действия заключаются в преобразовании информации после работы с исполнителем, а также создание начального шифра;

- компилятор, порядок работы которого состоит в следующем: меняется весь программный продукт, он принимает форму некоего модуля на техническом языке, вносится в память, а затем реализуется;

- ассемблер, функции состоят в переносе продукта на одноименном языке в продукт машинного характера.

Все имеющиеся языки программирования могут классифицироваться по ряду поколений:

- к первому относят направленные на определенный тип компьютеров, при этом их реализация ведется вручную, исключительно на устаревших компьютерах;

- на языках второго поколения применяется система автокодов, то есть все алгоритмы имеют мнемонический характер;

- в качестве языков третьей волны называют универсальные, они применяются для написания любых приложений, в качестве примера можно привести Бейсик, Кобол, Си, Паскаль;

- под языками четвертой волны понимают улучшенные, их функция состоит в работе с базами данных, они носят более специализированный характер;

- самыми новыми, молодыми, являются языки пятого поколения, они делятся на декаларативные, визуальные и направленные на отдельные объекты, в качестве образцов называют Пролог, ЛИСП (предназначены для обслуживания приложений с искусственным интеллектом), Си++, Delphi, Visual Basic.

Кроме того, все языки разделяют на процедурные и непроцедурные.

Среди языков первой группы механизм таков: продукт задает порядок реализации процедур, сам итог формируется некой моделью, которую можно создать только

после конкретного действия.

Все языки программирования процедурного характера можно разделить на структурные и операционные, исполнителем при этом в структурных отображаются полные циклы, ветки, а в операционных для этого применяют ряд операций. Наиболее известными структурными языками являются Паскаль, Си, Ада, ПЛ/1, соответственно среди операционных выделяют Фортран, Бейсик, Фокал.

Истоки декларативных языков программирования были заложены в 70-е годы, все языки этого типа бывают функциональными и логическими.

Особенностью функциональных языков является расчет конкретной функции, она производна от группы простых, а те распадаются на еще более мелкие. Базовой составляющей данного типа языков является рекурсия. Отличием функциональных языков является отсутствие оператора присвоения и циклов.

Программы, построенные на основе применения логических языков, никаких процедур и действий не характеризуют, они только назначают данные и взаимосвязи. После определения параметров, модели назначают группу вопросов, компьютер анализирует данные и отвечает, при этом алгоритм не характеризуется в программе, но подразумевается его наличие. Основным, отражающим все особенности данного типа программ, можно назвать язык Пролог. Основными элементами Пролога являются предикаты – базы данных, содержащих в себе порядок решения проблем и набор правил, характеризующих условия их выполнения.

Обособленно стоят языки программирования, направленные на отдельный объект. Эти языки не содержат в себе порядок реализации для выполнения отдельных задач, но в то же время в их составе имеются элементы процедурных языков. Данный тип языков оставляют за автором право выбора решения поставленных задач наиболее приемлемым для него способом.

Самым первым языком данного класса является Simula, разработанный в 60-е годы. При помощи языков данного типа был создан основной браузер Интернета, на языке Ява, которая является одной из защищенных и стабильных языков и по сей день.

Перспективным направлением развития языков программирования в наши дни является формирование публикаций посредством ссылок HTML, что дает возможность конструировать сайты, порталы, применяя при этом широкий набор

мультимедийных возможностей.

1.2 Сравнительная характеристика языков программирования высокого уровня

Для упрощения сравнительного анализа составим таблицу 1, в которой поместим основные критерии и наиболее известные языки программирования высокого уровня.

Таблица 1. – Сравнительная характеристика языков программирования высокого уровня

Язык	Перенос кодировки	Легкость доступа	Сетевые способности	Возможность параллельной работы	Расчеты с символами	Ст за
Java	Кодировка в байтах, виртуальный шифратор	Windows 95/NT, Solaris SPARC /Intel, HP-UX, OS/2, Macintosh, Linux	APIs (библиотеки классов), Remote Method Invocation, сетевая сериализация	Threads синхронизованные с помощью мониторов	Нет	Ин
TeleScript	Преобразование скрипта	Solaris SPARC, HP-UX, OS IRIX	TCP/IP, UDP, сетевая сериализация	множественные процессы работающие в вытесняющей многозадачности	Нет	во яз би кл ср

Tcl/Tk	Преобразование скрипта	Macintosh, Windows 3.1, Windows 95/NT, Solaris			Нет
Oz		Solaris SPARC, Sun OS, SGI IRIX, HP-UX, DEC Ultrix, IBM RS6000, Linux			Нет
Obliq	Преобразователь		гетерогенные распределенные сетевые вычисления	множественные threads внутри одного адресного пространства, множественные адресные пространства в пределах машины, распределенные вычисления в гетерогенных сетях	Нет
April			UDP		
AKL					

Scheme 48			
Penguin			
Python	Преобразователь	Windows, DOS, Macintosh, UNIX	интерфейс к TCP/IP
Facile			
AgentSpeak	Преобразователь		аналог потоков - намерения

Итак, таблица 1 нам показывает, что нет одного, самого лучшего языка программирования. Каждый из них в некоторых аспектах лучше, подходит для решения отдельной проблемы.

Следовательно, при выборе алгоритма языка программирования нужно руководствоваться задачами и целями его применения.

1.3 Основные этапы и технологии разработки программ на языке программирования высокого уровня

Составление программы зачастую является непростой задачей. Перед ее созданием у вас нет ни полного набора правил, ни алгоритма, указывающего вам, как именно писать программу. Создание программ — это творческий процесс. Но все же существует универсальный план написания программы.

Весь процесс написания программы можно разделить на две фазы: фаза решения задачи и фаза реализации.

- результатом фазы решения задачи является алгоритм, записанный в виде последовательных инструкций;

- затем алгоритм переводится на язык программирования, например C++. Эта фаза называется фазой реализации.

Первое, что нужно сделать — это убедиться, что решаемая с помощью компьютера задача корректно поставлена. Нельзя недооценивать это обстоятельство.

Необходимо заранее решить, что будет являться входной и выходной информацией для вашей программы, и в какой форме эта информация должна быть представлена.

Многие начинающие программисты не сразу осознают важность предварительного создания алгоритма и пытаются ускорить процесс создания программы, либо полностью исключив фазу поиска решения, либо просто сократив ее до постановки задачи.

Как показывает опыт, это не экономит время — написание программы в две стадии быстрее приводит к корректно работающей программе, поскольку, создавая алгоритм, вы можете не обращать внимание на особенности конкретного языка программирования, скажем Си. Таким образом, процесс создания алгоритма становиться менее сложным и уменьшается вероятность ошибок.

Даже при написании небольшой программы лучше за полдня аккуратно выполнить всю работу, чем потратить несколько дней на поиск ошибок в программе, написанной на скорую руку.

Фаза реализации — это тоже не простой процесс. На некоторые моменты нужно обратить особое внимание. Несмотря на то, что они могут оказаться сложными, они гораздо проще, чем кажется на первый взгляд. Познакомившись с Си или другим языком программирования, вы обнаружите, что запись алгоритма на языке программирования представляет собой рутинную работу.

Результаты обеих фаз создания программы необходимо подвергнуть тщательной проверке.

Перед тем, как написать саму программу, следует проверить и, в случае необходимости, исправить алгоритм. Такая проверка осуществляется мысленным прохождением всех инструкций алгоритма.

Написанная на Си программа проверяется при компиляции, а затем на простых примерах в процессе работы.

Некоторые ошибки умеет находить компилятор, тогда он выдает сообщение об ошибке. Остальные ошибки вам придется искать самим, вводя в программу простые данные и проверяя результат работы программы.

На практике ошибки и недочеты в программе обнаруживаются на самых разных стадиях создания программы, и вы будете вынуждены вернуться назад и исправить предыдущие этапы. Например, проверка алгоритма может выявить ошибки в постановке задачи. В этом случае вам придется вернуться к началу и переформулировать постановку задачи.

Иногда недочеты в формулировке задачи выясняются уже при проверке работы программы. Тогда вам придется изменить постановку задачи или алгоритм и все последующие этапы создания программы.

Как компьютер, так и любая система программирования ориентированы на обработку данных, то есть объектами управления являются данные, представленные обычно в виде переменных и форм их представления, называемых типами данных. Набор действий по обработке данных (переменных) в языке программирования обычно называется операциями, которые, соединяясь между собой, образуют выражения. Непосредственно алгоритм составляется на основе операторов языка программирования. Алгоритм обычно разбивается на логически завершенные части, которые называются модулями (процедурами, функциями).

Все это в совокупности составляет программу:
данные - переменные, создаваемые на основе типов данных;
выражения, включающие переменные и операции по их обработке;
логика алгоритма, составленная из операторов;
модули, соответствующие логически завершенным частям алгоритма.

Если отнести последние три компоненты к алгоритмической части программы, то можно дать такое короткое определение: любой язык программирования содержит средства для представления перечисленных выше компонент.

1.4 Языки программирования высокого уровня

Независимо от того, считается ли язык высокоуровневым или низкоуровневым (или где-то посередине), речь идет об абстракции. Машинный код не имеет абстракции - он содержит отдельные инструкции, передаваемые на компьютер. И поскольку машины имеют дело только с числами, они представлены в двоичном виде (хотя они иногда записываются в десятичной или шестнадцатеричной нотации). Вот пример машинного кода: В машинном коде операции должны быть указаны точно. Например, если часть информации должна быть извлечена из памяти, машинный код должен будет сообщить компьютеру, где в памяти его найти. Писать непосредственно в машинный код возможно, но очень сложно. Низкоуровневые языки программирования добавляют немного абстракции к машинным кодам. Эта абстракция скрывает конкретные инструкции машинного кода за декларациями, которые более читабельны для человека. Языки ассемблера являются языками самого низкого уровня рядом с машинным кодом. В машинный код вы можете написать что-то вроде «10110000 01100001», но язык ассемблера может упростить это как «MOV AL, 61h». Между тем, что написано на языке ассемблера, и инструкциями, переданными машине, по-прежнему существует почти одно-однозначное соответствие.

Перейдя на более популярные языки программирования, вы придете к чему-то вроде C. Хотя этот язык не такого низкого уровня, как язык ассемблера, все еще существует сильное соответствие между тем, что написано на C и машинным кодом. Большинство операций, написанных на C, могут быть заполнены небольшим количеством инструкций машинного кода. Языки программирования высокого уровня

Как и языки более низкого уровня, более высокие уровни охватывают широкий спектр абстракций. Некоторые языки, такие как Java (многие относят его к языкам программирования среднего уровня), все же дают вам большой контроль над тем, как компьютер управляет памятью и данными.

Другие, такие как Ruby и Python, очень абстрактны. Они дают вам меньше доступа к функциям нижнего уровня, но синтаксис гораздо легче читать и писать. Вы можете группировать вещи в классах, которые наследуют характеристики, поэтому вам нужно только объявить их один раз. Переменные, объекты, подпрограммы и циклы являются важными частями языков высокого уровня. Эти и другие концепции помогут вам рассказать машине о множестве вещей с короткими, краткими заявлениями. Если язык ассемблера имеет почти единообразное

соответствие между его командами и командами машинного кода, язык более высокого уровня может отправлять десятки команд с помощью одной строки кода. Важно отметить, что «языки программирования высокого уровня» могут включать в себя все, что более абстрактно, чем язык ассемблера. Какой язык изучать: низкого или высокого уровня? Это, безусловно, общий вопрос среди новых и начинающих программистов. Какие языки программирования лучше изучать: высокого или низкого уровня? Как и в случае со многими вопросами программирования, вопрос о языках программирования высокого и низкого уровня не так прост. Оба типа языков имеют важные преимущества. Низкоуровневые языки, так как они требуют небольшой интерпретации компьютером, обычно работают очень быстро. И они дают программистам большой контроль над хранением, памятью и извлечением данных. Однако языки высокого уровня интуитивно понятны и позволяют программистам писать код намного эффективнее. Эти языки также считаются «более безопасными», так как есть больше гарантий, которые препятствуют кодеру издавать плохо написанные команды, которые могут нанести ущерб. Но они не дают программистам такого же контроля над процессами низкого уровня. Помня об этом, вот список популярных языков по шкале от низкого до высокого: C C++ Java C# Perl Lisp JavaScript Python Ruby SQL

Конечно, это отчасти субъективно. И включает только крошечную часть доступных языков. Но это должно дать вам некоторое представление о том, на каком уровне находятся интересующие вас языки.

Если вы хотите программировать операционные системы, ядра или что-то, что необходимо для работы на максимальной скорости, язык более низкого уровня может быть хорошим выбором. Большая часть Windows, OS X и Linux написана на языках C и C-производных языках, таких как C ++ и Objective-C. Многие современные приложения пишутся на языках более высокого уровня или даже на предметно-ориентированных языках. Python и Ruby особенно популярны для веб-приложений, хотя HTML5 становится все более мощным. Языки, такие как Swift, C #, JavaScript и SQL, имеют свои сильные и слабые стороны. Рассмотрите возможность обучения языкам обоих уровней Недавно читал тему на форуме по программированию и наткнулся на интересное предложение: изучите сразу оба уровня. Вы получите более глубокое понимание типов абстракций, которые делают язык более высокого уровня более эффективным. Конечно, изучение двух языков одновременно непросто, так что вы можете немного растянуть их изучение. И выбор двух языков, которые наиболее похожи, может быть полезным. Опять же, мы вернемся к тому, о чем я говорил раньше: выберите язык, основанный на том, что

вы хотите сделать. Проведите некоторое исследование, чтобы узнать, какие языки люди используют в своей области. Затем используйте эту информацию, чтобы выбрать язык высокого и низкого уровня, и начните изучать их. Вы скоро увидите параллели, и вы получите гораздо более глубокое понимание того, как работает программирование.

Сосредоточьтесь на цели, а не на средстве Существует множество критериев, которые вы можете использовать для выбора языка программирования. Одним из критериев является высокий и низкий уровень. Но почти в каждом случае критерии, которые вы должны использовать, - это то, что вы хотите запрограммировать. Вашему проекту может быть полезен язык низкого уровня. Или это может быть намного более эффективно на высоком уровне. Вы должны сами выбрать правильный инструмент для работы. Сосредоточьтесь на своей цели, и каждый раз выбирайте правильный язык.

ГЛАВА 2. ИЗУЧЕНИЕ РОЛИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННЫХ СИСТЕМ В ДЕЯТЕЛЬНОСТИ УНИВЕРСИТЕТА «УНИВЕРСИТЕТ»

2.1 Организация работы департамента информационных технологий, его цели и задачи

Департамент информационных технологий является структурным подразделением учреждения.

Миссия Департамента

Предоставлять студентам, преподавателям и сотрудникам университета удобную возможность пользоваться максимальным набором современных информационных технологий.

Перечень внедряемых новшеств

Для студентов:

- виртуальная обучающая среда, основанная на электронных образовательных ресурсах, с поддержкой кредитно-модульной системы и индивидуальных образовательных траекторий, с возможностью доступа с мобильных устройств и ноутбуков;
- электронно-библиотечная система, обеспечивающая единый доступ к фондам, в т.ч. внешним;
- сервисы обеспечения научно-исследовательской и проектной деятельности, в т.ч. интерактивная виртуальная площадка взаимодействия и удаленный доступ к научному и исследовательскому оборудованию;
- сервисы: пропускная система, заказ услуг в сфере спорта, здравоохранения, проживания;
- единый портал электронного университета, обеспечивающий взаимодействие студентов/преподавателей и работу с любыми сервисами ЭУ, в т.ч. на мобильных устройствах;
- технологические решения, позволяющие получать доступ к виртуальным учебным аудиториям с помощью видеоконференцсвязи, в т.ч. с ноутбуков и нетбуков;
- внедрение универсальной электронной карты студента/преподавателя, которая будет использоваться вместо пропуска, читательского билета, как средство платежа.

1. WiFi-сеть на территории учебных корпусов и общежитий с доступом в Интернет.
2. Мобильная связь: привлекательный тарифный план для студентов с бесплатным доступом к ресурсам.
3. Print-киоски - установленные в специально отведенных местах в учебных корпусах принтеры, копиры и сканеры, в т.ч. широкоформатные и цветные, которые могут использоваться студентами самостоятельно в рамках отведенных персональных лимитов.
4. Информационные киоски с функцией приема платежей для оплаты различных услуг, в т.ч. оплаты за обучение, оплаты сотовой связи и прочих услуг (телевидение, Интернет, охрана) и широкими информационными возможностями: доступ к порталу, городская афиша, городские новости, карта города с маршрутами и остановками городского транспорта, полезные телефоны и т.д.
5. Единая Служба техподдержки пользователей информационных технологий.

Для сотрудников и ППС:

1. Электронный документооборот и управление взаимодействием, позволяющий осуществлять работу с документами и решать целый спектр задач: поиск информации, поддержание ее в актуальном состоянии, обеспечение режима конфиденциальности и сохранности документов и т.д.
2. Виртуальная АТС, предоставляющая сервисы по обеспечению доступности сотрудников для внешних и внутренних абонентов, позволяющая осуществлять звонки между сотрудниками по внутреннему номеру в т.ч. за пределами, используя аппарат мобильной связи.
3. Мобильная связь: привлекательный тарифный план с бесплатными звонками между сотрудниками и преподавателями с бесплатным доступом к ресурсам.
4. Единая Служба техподдержки пользователей информационных технологий, позволяющая централизовано сообщать о проблемах и вопросах, быть в курсе их решений и максимально быстро получать квалифицированную помощь.
5. Print-киоски - установленные в специально отведенных местах в учебных корпусах принтеры, копиры и сканеры, в т.ч. широкоформатные и цветные, которые могут использоваться сотрудниками и преподавателями самостоятельно в рамках отведенных персональных лимитов.
6. Виртуальная обучающая среда, основанная на электронных образовательных ресурсах, с поддержкой кредитно-модульной системы и индивидуальных образовательных траекторий, с возможностью доступа с мобильных устройств и ноутбуков. Особое внимание будет уделено
7. Электронно-библиотечная система, обеспечивающая единый доступ к фондам, в т.ч. внешним.
8. Сервисы обеспечения научно-исследовательской и проектной деятельности, в т.ч. интерактивная виртуальная площадка взаимодействия и удаленный доступ к научному и исследовательскому оборудованию с доступом через Интернет.

Основные задачи Департамента:

1. Департамент определяет единые стандарты развития информационных систем в рамках всего учреждения и осуществляет контроль над их исполнением.
2. Обеспечивает надлежащий уровень функционирования всех информационных систем учреждения и инфраструктуры, предназначенной для функционирования информационных систем.

3. Осуществляет управление и выполнение комплексных проектов, реализуемых Университетом в рамках федеральных и региональных программ в части создания, внедрения и обеспечения информационными технологиями.
4. Координация и экспертиза процессов информатизации учреждения, внедрение новых информационных технологий в образовательную и управленческую сферу. Организация разработки комплексной программы информатизации учреждения.
5. Проведение экспертиз предложений, поступающих от подразделений, связанных с приобретением вычислительной техники, подключением к информационным сетям, разработкой программных средств и формированием банков данных.
6. Выработка концепции и проведение мероприятий по обеспечению информационной безопасности университета.
7. Консультационная поддержка подразделений университета по вопросам компьютеризации.
8. Организация и проведение постоянно действующих семинаров, курсов, лекций, семинаров и консультаций для студентов, сотрудников и профессорско-преподавательского состава по вопросам информатизации.
9. Разработка и внедрение документов системы менеджмента качества, закрепленных за подразделением.
10. Разработка для подразделений университета программного обеспечения, реализующего информационные технологии в преподавании учебных дисциплин и управлении.
11. Проведение прикладных научно-исследовательских работ в области использования современных информационных технологий.

2.2 Аппаратное и программное обеспечение учреждения, перспективы внедрения новых информационных систем

Университет является одним из участников международной программы MSDN Academic Alliance. Она нацелена на льготное обеспечение ВУЗов по всему миру программными продуктами компании Microsoft, а также дополнительными материалами и документацией; в том числе, и дисковой версией MSDN Library.

Предоставляемое программное обеспечение может быть установлено на любом количестве компьютеров, расположенных в стенах университета, но только в учебных и исследовательских целях. Данная лицензия запрещает его использование для поддержания сетевой инфраструктуры университета.

Кроме того, некоторые программные продукты, распространяемые в рамках данной программы, предоставляются для домашнего изучения всем сотрудникам и студентам/аспирантам ТПУ. Для получения программного обеспечения используется система ELMS (Electronic License Management System).

Таблица 2. – Данные о программном обеспечении и портах сервера лицензий

Название ПО	**Порт сервера лицензий**
ANSYS Academic Research	1055
Ansoft	27001
MathCAD	7788
Pro/ENGINEER	7788
MATLAB	27000
Procast	27002
NX, Teamcenter	28000
Technomatix	28006
AVL (www.avl.com)	27003
SolidWorks	25734
Intel Parallel Studio XE Cluster Edition for Linux	28518

Чтобы приступить к работе в системе ELMS необходимо перейти на сайт университета, нажать кнопку "Вход" и ввести Вашу учетную запись в домене и пароль, и согласиться с условиями использования получаемого программного обеспечения на следующей странице.

Зайдя в систему под своей учетной записью, можно заказать необходимые продукты. При этом, ключ активации того или иного продукта выдаётся автоматически, а дистрибутив можно скачать по ссылке "Файл" в любое время.

Лицензионное программное обеспечение, доступное в сети, может использоваться всеми подразделениями университета. Для этого в корпоративной сети функционирует сервер лицензий. Чтобы использовать эти лицензии, необходимо в программном обеспечении указать в качестве адреса сетевого сервера лицензий (hostname) и порт в соответствии с таблицей 2.

ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ МОДЕЛИ И МОДЕЛИ ОСНОВНЫХ БИЗНЕС-ПРОЦЕССОВ УНИВЕРСИТЕТА

Создадим программный продукт, позволяющий оптимизировать работу службы кадров учреждения (необходимые параметры указаны в таблице 3.1). В данном департаменте помещены в архиве все сведения касательно каждого работника университета – от руководящего звена до вспомогательного персонала. Особенностью является наличие информации о степени и звании специалистов. Помимо этого, дополнительной информацией являются данные о послужном списке работника, что существенно упрощает процесс составления договоров с кандидатами на свободные вакансии, или при переоформлении договоров на продление сотрудничества с организацией. Также в отдельной папке хранятся данные о характере исполняемых обязанностей, то есть о наличии премий либо наказаний, при этом взыскания описаны в компьютерном варианте.

Таблица 3. – Характеристика параметров для составления базы данных в отделе кадров университета

№	Поле	Тип	Размер	Описание

1	PersonID	Числовой	5	Код специалиста
2	Name	Текстовый	40	Инициалы специалиста
3	Departament	Текстовый	40	Кафедра специализации
4	Institute	Текстовый	40	Институт, в котором работает
5	Birth	Дата/время	Авто	Информация о дате рождения
6	Place	Текстовый	20	Сведения о месте рождения
7	Address	Текстовый	60	Адрес проживания
8	Phone	Текстовый	15	Телефон для контактов
9	Education	Текстовый	40	Университет, в котором получено образование
10	Year	Числовой	4	Год получения диплома
11	Speciality	Текстовый	30	Специализация и профиль специалиста
12	Picture	Объект OLE	Авто	Фото специалиста
13	DegreeYes	Логический	1	Наличие ученой степени (есть/нет)
14	Degree	Числовой	1	Характеристика ученой степени

15	Rank	Числовой	1	Характеристика ученого звания
16	Post	Текстовый	20	Должность, по которой работает
17	Comment	Поле Memo	Авто	Дополнительные сведения
18	Passport	Текстовый	20	Номер паспорта
19	PassportDate	Дата/время	Авто	Дата выдачи паспорта
20	Region	Текстовый	40	Организация, выдавшая паспорт
21	WorkBegin	Дата/время	Авто	Дата начала трудового стажа
22	WorkEnd	Дата/время	Авто	Дата завершения трудового стажа
23	Work	Текстовый	20	Ранее занимаемая должность
24	WorkPlace	Текстовый	20	Организация, в которой ранее работал специалист
25	WorkAddress	Текстовый	60	Адрес организации, в которой ранее работал специалист
26	WorkPhone	Текстовый	15	Телефон организации
27	Reason	Текстовый	30	Причина, по которой был уволен специалист
28	Penalty	Поле Memo	Авто	Данные о нарушениях в процессе выполнения обязанностей

Процесс формирования базы данных выполняется в следующей очередности:

- построение табличной модели, приведение сведений в рабочий вид

С целью дальнейшей обработки информации нужно построить базу данных, для этого используем продукт компании Microsoft Office, а именно MS Access 2007.

Под базой данных следует понимать комбинацию сведений, находящихся в определенной связи между собой, призванных для решения нужд и запросов компании.

К системе управления базами данных можно отнести программное обеспечение, посредством которых осуществляется процесс формирования, улучшения баз данных, а также выполняются операции с доступом или блокировкой пользователей.

Перед стартом работы сгруппируем все сведения в начальной табличной форме и разместим их в ряд второстепенных, при этом выделим самую важную, приоритетную, в нее поместим всю информацию о каждом специалисте организации.

Ячейку Address разделим на следующие секторы: StreetName, Sign, First и скинем их в ранее выделенную, самую важную табличную форму.

Ячейки PersonID указанной ранее таблицы выделены основными полями, то есть все изменения будут наблюдаться только по данным секторам. Для выявления базовой ячейки нужно его пометить как ключевое, тогда ячейка будет определена с соответственным символом ключа.

Следующим шагом построения базы данных является формирование связей, при этом все данные и сведения между собой коррелируются и база данных запускается в работу. В Access применяются следующие типа корреляций:

- друг к другу

- один ко всем

- все к одному

- все ко всем.

С целью формирования корреляции необходимо запустить модель данных, что осуществляется в следующем порядке: надо пометить «Создание» на ленте, затем запустить «Схема данных». Затем данные табличные формы перекинем на место модели данных, выведем связь между таблицами, в результате должно высветиться окно «Смена связей и корреляций».

С целью придания целого характера сведений поставим флажки в полях «Обеспечение целых данных» и «Обновление корректирующих ячеек каскадом».

После выполнения данного этапа работы приступаем к самой трудоемкой части работы – формированию программных приложений. Если отсутствует полностью выполненное программное приложение, то оператор не имеет возможности вести действия со сведениями, а также отсутствует завершенная модель программного продукта. Целью приложения является создание корреляции данных графического характера с информацией символьной, это даст возможность результативно работать со сведениями.

Предлагаемое программное обеспечение разработано при помощи Visual Basic, являющегося основой для значительной части программ, сокращенное его название VBA.

Перед тем как проводить операции в VBA нужно в Автокаде осуществить операции во вкладке «Сервис-Макросы-Редактор», после чего будет запущено окно VBA.

Перед стартом работы построим начальную табличную форму и выполним для нее запуск, это будет необходимо для дальнейших запусков Автокада. Произведем выбор элемента во вкладке «Insert UserForm», высветится отдельная табличная форма, в которую поместим инструменты редактирования посредством Toolbox, их изменим в соответствии с вкладкой Properties.

По завершению данной процедуры отобразим код редактирования для помеченных инструментов, с целью дальнейшего запуска начального массива информации сделаем редактор событий AcadDocument_Activate(), он будет необходим для подтверждения рабочих документов. Для выполнения данного действия в соответствующем поле Project активируем ThisDrawing, что приведет к запуску поля изменения введения информации. Здесь будет открыт доступ к спискам, при этом слева отберем AcadDocument, а справа Activate и будет сгенерирован редактор этих изменений, здесь добавим такую запись:

Private Sub AcadDocument_Activate()

StartForm.Show ' Во время запуска документа демонстрируем стартовую форму End Sub

Этим действием будет создана начальная форма данных.

Разрабатываемый продукт осуществляет операцию как в режиме оператора, так и редактора. По умолчанию путь к базе данных определяется автоматически в той же вкладке, как и рисунок. В то же время можно провести и отбор другого варианта пути, при этом нажмем кнопку доступа. Видно, что в случае режима работы редактора происходит запуск Автокада и все пароли отсутствуют.

Также можно выбрать другой путь к базе данных, нажав на кнопку открытия. При выборе режима конструктора осуществляется переход к Автокаду и никакой код не выполняется. При выборе пользовательского режима блокируются все слои, кроме слоя Блоков, дабы избежать потери графической информации. В том случае если оператор работает с базой данных, то он может выбрать один из трех рисунков в модели, каждый из рисунков снабжен справочной информацией и возможностью просмотра для детальных сведений.

Данная процедура выполняется в редакторе событий ThisDrawing.

AcadDocument_SelectionChanged() (смена выбора) в таком порядке:

If ThisDrawing.PickfirstSelectionSet.count > 0 Then 'Проверка выбора чего-либо

Set objGen = ThisDrawing.PickfirstSelectionSet.Item _

(ThisDrawing.PickfirstSelectionSet.count - 1) 'Если есть выбор, то отбираем objGen как финальный

If objGen.ObjectName = "AcDbBlockReference" Then 'Проверка, отобранный массив блок или нет

Select Case objGen.Name 'Сверка, какой объект отобран

Case 1

If MsgBox("Выбрана Библиотека" & vbCr & "Показать информацию об этом помещении?", _

vbOKCancel, "Выбрано помещение") = vbOK Then 'Выдаем сообщение и запрос на вывод информации

ShowInf = True 'Показать информацию

End If

ID = 1 'Устанавливаем номер выбранного помещения

Case 2

If MsgBox("Выбран Деканат" & vbCr & "Показать информацию об этом помещении?",

_

vbOKCancel, "Выбрано помещение") = vbOK Then

ShowInf = True

End If

ID = 2

Case 3

If MsgBox("Выбрана Кафедра" & vbCr & "Показать информацию об этом помещении?", _

vbOKCancel, "Выбрано помещение") = vbOK Then

ShowInf = True

End If

ID = 3

End Select

Осуществляется процесс связи с базой данных и отправка запроса на весь персонал, функционирующий в данном университете, при этом перечень специалистов отмечен в ListBox. Сведения о запросе отмечены таким образом:

Public record As ADODB.Recordset 'Переменная запроса к базе данных

...

```
Set record = New ADODB.Recordset 'Создаем переменную запроса к базе

...

With record

'Создаём запрос в базу

.Source = "Select tblWorker.PersonID, tblWorker.Family, tblWorker.FirstName,
tblWorker.SecondName, " & _

"tblWorkPlace.Place From tblWorker, tblWorkPlace where
tblWorker.WorkPlace=tblWorkPlace.WorkPlace and " & _

"tblWorker.WorkPlace=" & ID & " order by Family, FirstName, SecondName"

'Открываем его

.Open

CountQuery = .RecordCount 'Считаем кол-во записей в запросе

End With

FlatInf.ListBox1.Clear

FlatInf.TextBox1.Text = record!Place 'Устанавливаем место работы сотрудника

FlatInf.Label3.Caption = "Всего: " & CountQuery & " " &
Operations.intToStroka(CountQuery) 'Устанавливаем в Label3 кол-во работников,
попавших в запрос

For i = 0 To CountQuery - 1 'Перебираем все записи

FlatInf.ListBox1.AddItem (record!Family & " " & record!FirstName & " " &
record!SecondName) 'Добавляем в список Фамилию, имя и отчество работника
record.MoveNext 'Переходим к следующей записи

Next i
```

При запуске кнопки «Информация о специалисте» активируется таблица с данными о каждом специалисте.

В случае формирования этого массива информации отправляется запрос в построенную базу данных, при этом ведется сбор сведений по всем табличным формам, выполняются также сверки с корректностью сведений в табличных формах.

```
Rem Фамилия

If rec!Family <> "" Then

.TextBox1.Text = rec!Family

.Caption = .Caption & rec!Family

Else TextBox1.Text = ""

End If
```

Инструменты ComboBox пополняются сведениями из всех табличных форм, в качестве примера приведем список кафедр университета:

```
Rem Составляем список кафедр

DopRec.Open ("Select * from tblDepartament") 'Получаем все кафедры

For i = 0 To DopRec.RecordCount - 1 'Проходим по всем записям

.ComboBox1.AddItem (DopRec!Departament) 'Добавляем запись в ComboBox1

DopRec.MoveNext 'Переходим к следующей записи

Next i

Rem Выбор Кафедры

If (Not rec!DepartamentID = 0) Then 'Если в поле кафедры есть значение

.ComboBox1.ListIndex = rec!DepartamentID - 1 'Выбираем его номер в списке

Else 'Иначе

.ComboBox1.ListIndex = -1 'Ничего не выбираем

End If
```

DopRec.Close

Из данной табличной формы становится доступным вспомогательное окно, содержащее сведения о состоянии трудовой книжки специалистов. Данные сведения приобретаются при помощи запросов к базе данных.

Из табличного массива с данными о специалисте необходимо осуществлять переход к начальному массиву, активировав необходимую кнопку. Здесь также появляется возможность смотреть пометки базы данных, редактировать, вносить новые сведения о работнике и удаления пометок.

ЗАКЛЮЧЕНИЕ

Целью проведенного исследования явился обзор применяемых в практике ведения учебного процесса программных продуктов на базе Университета «Университет».

Выбранная тема представляет интерес для специалистов, ведущих свою деятельность в поле высоких технологий, поскольку в последнее время имеет место увеличение роли применения компьютерной техники во всех сферах общественной жизни, а особенно в системе высшего образования, так как данный институт общества является определяющим для развития каждой личности и введение технических новшеств обеспечит большую эффективность образовательного процесса и, следовательно, развитие государства на мировой арене.

В Российской Федерации развитие компьютерной модели общества осуществляется высокими темпами, что дает возможность интегрироваться в мировую модель высшего образования, основанного на использовании передовых технологий.

В результате аналитической работы проанализированы особенности применения компьютерных технологий, основные модели программного обеспечения в Университете «Университет», а именно рассмотрены:

- модель системы, ее компоненты, взаимодействие между технической составляющей и прикладными программными продуктами;

- сравнительная характеристика различных языков программирования с целью определения перспектив их использования в конкретном университете;

- разработана собственная база данных как объект инновационного подхода для повышения эффективности деятельности учреждения.

В заключении можно сказать, что внедрение программных продуктов и высоких технологий в обществе приобретает все большую популярность.

Всех сфер деятельности, в которых применяется комплекс компьютерного оборудования и прикладных программ, нельзя перечислить, при этом они выполняются главные функции, среди которых можно перечислить и учетную, и аналитическую, предоставляют необходимую информацию, способствуют более качественному контролю за процессами.

На современном этапе развития системы высшего образования уже не представляется возможным обеспечить высокие результаты работы без наличия высоких технологий.

СПИСОК ЛИТЕРАТУРЫ

1. Баронов В.В. Автоматизация управления предприятия. – М.: Инфо-М, 2016. – 239с.

2. Буч Г. Объектно-ориентированное проектирование с примерами применения – М.: Радио и связь, 2015. - 149с.

3. Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем. – М.: Вильямс, 2014. - 176с.

4. Титоренко Г.А. Автоматизированные информационные технологии в экономике. – М.: Финансы и статистика, 2014. – 265с.

5. Карминский А.М., Нестеров П.В. Информатизация бизнеса. – М.: Финансы и статистика, 2015. - 397с.

6. Корнеев И.К., Машурцев В.А. Информационные технологии в управлении. – М.: ИНФРА-М, 2015. – 254с.

7. Лихачева Г.Н. Информационные технологии в экономике: Учебно-практическое пособие / МЭСИ. – М.: МЭСИ, 2014. - 364с.

8. Якубайтис Э.А. Информационные сети и системы. – М.: Финансы и статистика, 2014. – 212с.

9. Юценко А.М. Моделирование информационного обеспечения управленческих решений. - М.: Вильямс, 2014. – 419с.

10. Соммервилл И. Инженерия программного обеспечения, 9-ое издание.: Пер. с англ. – М.: Вильямс, 2015. - 408с.