**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ПП.09.02.07-1.24.202.15

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

ПК.04 Обеспечивать защиту программного обеспечения компьютерных систем программными средствами

|  |  |  |
| --- | --- | --- |
| Руководитель от предприятия:  М.П. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Л.Г. Стрежнёва) |
| Руководитель от техникума: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (С.Н. Касьяненко) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Д.А. Отчесов) |

Иркутск 2024

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc163790557)

[1. Виды СУБД – описать критерии выбора. 4](#_Toc163790558)

[2. Характеристики программного обеспечения. 8](#_Toc163790559)

[3. Описание компонентов. 11](#_Toc163790560)

[4. Защита программного обеспечения. 13](#_Toc163790561)

[ЗАКЛЮЧЕНИЕ 16](#_Toc163790562)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 17](#_Toc163790563)

# ВВЕДЕНИЕ

Целью производственной практики по направлению информационные системы и программирование является применение на практике теоретические знаний, полученных в процессе обучения, приблизиться к понимаю своей будущей работы и обязанностей.

Цели практики:

– расширение и систематизация знаний на основе изучений

деятельности;

– приобретение практических навыков в сфере информационных технологий.

Задачи практики:

– Овладение практическими навыками;

– Развитие профессионального мышления;

– Проверка профессиональной готовности к самостоятельной трудовой

деятельности.

# Виды СУБД – описать критерии выбора.

Существует много типов данных мы рассмотрим следующие типы баз данных:

* Реляционные,
* Ключ-значение,
* Документо-ориентированные,
* Базы данных временных рядов.
* Графовые базы данных,
* Поисковые базы данных (Search Engines),
* Объектно-ориентированные базы данных.

Реляционные базы данных

Наиболее известными реляционными базами данных являются Open Source проекты PostgreSQL, MySQL и SQLite, а также проприетарные решения Oracle, Microsoft SQL Server и IBM Db2Relational.

Суть реляционных баз в хранении данных в связанных таблицах.

Стоит заметить, что реляционные базы бывают с хранением данных по строкам (PostgreSQL) и по столбцам/колонкам (ClickHouse, Vertica).

Колоночные/столбцовые базы лучше подходят для аналитики, в то время как ориентация на строки лучше подходит для транзакционных нагрузок. Реляционные СУБД – самый распространенный тип баз данных. В подкате таблица из более чем 150 вариантов.

Key-value (ключ значение) базы данных

Тип баз данных Key-value предназначен для осуществления быстрых, почти мгновенных запросов для таких задач как кэш, отображение баланса и т.д. Высокая скорость осуществляется за счет хранения данных по принципу ключ-значение, и в большинстве случаев благодаря работе в оперативной памяти.

Словари содержат коллекцию объектов или записей, а объекты содержат множество различных полей, каждое из которых содержит данные. Записи хранятся и извлекаются с использованием ключа, который однозначно идентифицирует запись и используется для быстрого поиска данных.

Основным применением является ускорение отображения данных для конечных пользователей и снижение нагрузок, в том числе I/O на инфраструктуру организаций. Наиболее известными и широко используемымиKey-Value решениями являются Redis и Memcached.

Документо-ориентированные базы данных

Документо-ориентированные базы данных созданы для хранения иерархических структур данных (документов). Основой документо-ориентированных СУБД являются документные хранилища, имеющие структуру дерева или леса. Деревья начинаются с корневого узла и может содержать несколько внутренних и листовых узлов. Листовые узлы содержат данные, которые при добавлении документа заносятся в индексы, это дает возможность даже при достаточно сложной структуре находить путь к искомых данных. В отличие от хранилищ типа ключ-значение, выборка по запросу к документному хранилищу может содержать части большого количества документов без полной загрузки этих документов в оперативную память.

Базы данных временных рядов

Если у вас есть упорядоченные по времени данные с временными метками, такие как метрики от инфраструктуры или данные датчиков, может быть полезно использовать одну из баз данных временных рядов.

Общие характеристики баз данных временных рядов:

* Данные временных рядов всегда собираются на протяжении определенного периода времени.
* Данные из рабочих нагрузок являются новыми и записываются как вставки. Уже существующие данные не обновляются путем замены.
* Когда данные записываются, они автоматически назначаются последнему интервалу времени.

Базы данных временных рядов часто используются для осуществления мониторинга различных метрик (будь то загрузка CPU, или показатели работы какого-либо датчика).

Графовые базы данных

А если вам нужно анализировать отношения данных, их связи или просто упростить запросы с километровыми Join, имеет смысл использовать графовые базы данных.

Данные и их связи представляются как вершины и ребра графа соответственно. Таким способом можно легко представить денежные переводы (для определения различных мошеннических схем), связи в социальной сети и граф общения между операторами сотовой сети.

Поисковые базы данных (Search Engines)

Eсли вам необходимо осуществлять поиск большим объемам данных, особенно неструктурированным, как пример поиск по нескольким терабайтами логов, то вам может пригодиться использовать базу данных, совмещающую с функционалом хранения информации еще и функционал поиска по текстам.

Представим, что у вас есть N петабайт логов (или других текстовых данных). Обычный поиск по словам уже не подойдет, чтобы осуществить поиск и аналитику в разумное время.

На помощь приходит индексирование. Если очень утрировано его рассмотреть, можно его представить следующим способом. Похожая система используется для построения систем поиска плагиата, правда там чаще применяют не слова, а индексы с наслоением, а искать по индексу существенно быстрее, чем по совпадению по словам в документах.

Строго говоря, для поиска по документам можно использовать и эмбеддинги нейронных сетей, в которых закодирован "смысл" высказываний. Но для данной задачи лучше подойдут векторные базы данных, которые замыкают наш список. Современные поисковые СУБД предлагают значительно более широкий функционал. Наиболее популярны такие решения как Elasticsearch (и его версия - OpenSearch), проприетарный Splunk, о котором я писал одну из прошлых статей и Sphinx.

Объектно-ориентированные базы данных

Объектно-ориентированные базы данных представляют собой базы данных, в где информация представлена в виде объектов, как в объектно-ориентированных языках программирования. Такие базы данных появились как способ нативной коммуникации кода, написанного с использованием объектно-ориентированных языков с базой данных.

Объектно-ориентированные базы данных обладают следующими преимуществами:

– Нет проблемы несоответствия модели данных в бд и приложении, так как в БД они сохраняются в том же виде.

– Не нужно отдельно поддерживать модель данных на стороне базы данных.

– Все объекты на уровне источника строго типизированы.

# Характеристики программного обеспечения.

Программное обеспечение (ПО) – это совокупность программ, позволяющая организовать решение задач на ЭВМ. ПО и архитектура ЭВМ (аппаратное обеспечение) образуют комплекс взаимосвязанных и разнообразных функциональных средств ЭВМ, определяющих способность решения того или иного класса задач.

Программные средства можно классифицировать по разным признакам. Наиболее общей является классификация, в которой основополагающим признаком служит область использования программных продуктов:

* аппаратная часть компьютеров и сетей ЭВМ;
* технология разработки программ;
* функциональные задачи различных предметных отраслей.
* Исходя из этого выделяют три класса программных продуктов (рис. 4.3):
* Системное программное обеспечение;
* инструментарий технологии программирования;
* пакеты прикладных программ.

Под системным ПО понимается совокупность программ и программных комплексов для обеспечения работы компьютеров и сетей ЭВМ. Системное ПО управляет всеми ресурсами ЭВМ и осуществляет общую организацию процесса обработки информации и интерфейсы между ЭВМ, пользователем, аппаратными и программными средствами. Системное ПО тесно связано с типом компьютера, является его неотъемлемой частью независимо от специфики предметной области и решаемых задач. Главную часть системного ПО составляет Операционная система (ОС). К системному ПО также относятся программы для диагностики и контроля работы компьютера, архиваторы, антивирусы, программы для обслуживания дисков, программные оболочки, драйверы внешних устройств, сетевое ПО и телекоммуникационные программы. Примеры важнейших системных программ: MS-DOS, NortonCommander, NortonUtilities, Windows, Linux.

Инструментарий технологии программирования предназначен для эффективной разработки программных средств различного назначения. Примеры важнейших систем программирования: Turbo Pascal, QBasic, Borland C++, Visual Basic. Пакеты прикладных программ предназначены для решения задач из различных областей деятельности человека.

Системное программное обеспечение.

Системное программное обеспечение по характеру использования отдельных комплексов программ подразделяется на следующие классы:

* базовые системы ввода-вывода;
* операционные системы;
* операционные оболочки.

Базовая система ввода-вывода (BIOS – Basic Input Output System) представляет собой набор программ, обеспечивающих взаимодействие операционной системы с различными устройствами компьютера:

* поддержку функций ввода-вывода;
* тестирование оборудования при включении компьютера;
* загрузку операционной системы не только с жесткого или гибкого дисков, но и с приводов CD-ROM.

Программы BIOS находятся в специальном постоянном запоминающем устройстве, расположенном на системной плате компьютера. Поэтому BIOS может быть отнесена к особой категории компьютерных компонентов, занимая промежуточное положение между аппаратурой и ПО.

Операционная система (ОС) является основой системного ПО. ОС управляет работой всех устройств компьютера и процессом выполнения программ пользователей от момента их поступления в систему до выдачи результатов.

В состав ОС входит комплекс программ, выполняющих следующие функции:

* управление работой всех устройств компьютера;
* контроль работоспособности оборудования;
* первоначальная загрузка системы;
* управление файловой системой;
* распределение ресурсов компьютера, таких, как оперативная память, процессорное время, периферийные устройства, между программами пользователей;
* управление загрузкой и выполнением прикладных программ.

По широте охвата одновременно выполняемых задач ОС можно разбить на три группы: однозадачные (однопользовательские), многозадачные (многопользовательские) и сетевые.

* Однозадачные ОС предназначены для работы одного пользователя в каждый конкретный момент с одной конкретной задачей (например, операционные системы типа MSDOS).
* Многозадачные ОС обеспечивают коллективное использование компьютера в мультипрограммном режиме разделения времени.

Сетевые ОС связаны с появлением локальных и глобальных сетей. Они предназначены для обеспечения доступа пользователя ко всем ресурсам вычислительной сети.

Однозадачные дисковые операционные системы различных фирм MSDOS, PCDOS и Novell DOS были просты и экономичны, но морально устарели и уступили место операционным системам нового поколения.

Современные ОС, такие, как ОС семейств UNIX и WINDOWS, являются многозадачными, предоставляют пользователю развитый графический интерфейс, совместимы с приложениями, разработанными дляMSDOS. Они независимы от аппаратуры, поддерживают все виды периферийных устройств. Они способны использовать все возможности современных микропроцессоров, устойчивы в работе, так как имеют средства защиты от сбоев и ошибок.

# Описание компонентов.

Компоненты бывают: Мелкими и крупными; Визуальными и сервисными; Компонентами данных. К мелким компонентам относят, например, кнопки, поля ввода, выпадающие списки. Они решают простые задачи. Крупные компоненты объединяют в себе мелкие, реализуя при этом более сложный функционал.

Визуальные компоненты (visual components) — это видимые элементы пользовательского интерфейса: кнопки, метки, блоки списков и др. Они выглядят одинаково и на стадии проектирования, и во время работы приложения. Компоненты данных используются для доступа и манипулирования данными из различных источников. Существует множество готовых библиотек UI компонентов, упрощающих создание интерфейса.

Самые известные из них:

* Material UI;
* Bootstrap;
* Ant Design.

Структура и состав компонента Любой компонент состоит из обязательных и дополнительных элементов. К обязательным относятся:

* Логика (код);
* Представление (шаблон).

Логика реализует функционал компонента, представление определяет способ отображения данных. Дополнительно компонент может включать: Стили; Тесты; Документацию. Они позволяют улучшить внешний вид, качество и удобство использования компонента. В процессе эксплуатации компонент проходит определенные этапы жизненного цикла: создание, инициализацию, обновление состояния, уничтожение. При проектировании компонентной архитектуры применяются различные паттерны, например MVC, MVVM, Flux и другие. Разработка компонентов существует несколько основных принципов, которыми следует руководствоваться при создании компонентов: Принцип единой ответственности. Компонент должен решать лишь одну задачу. Принцип слабой связности. Зависимости между компонентами следует свести к минимуму. Принцип переносимости. Компонент должен быть максимально независим от окружения.

Разработчики компонентов должны следовать определенным соглашениям, касающимся как структуры файлов внутри компонента, так и правил именования классов, функций, свойств. Это обеспечивает понятность кода для всей команды. Инкапсуляция и API компонента. Важно грамотно выстроить архитектуру компонента таким образом, чтобы скрыть (инкапсулировать) внутреннюю реализацию и предоставить доступ к функционалу только через публичный API. Это повышает безопасность и независимость компонента. Юнит-тестирование компонентов Неотъемлемой частью разработки компонентов является их тестирование, особенно на уровне юнит-тестов. Юнит-тесты проверяют работу отдельных функций и классов компонента, что позволяет выявить ошибки на ранней стадии. Компоненты в популярных фреймворках.

Рассмотрим реализацию компонентной модели в таких популярных фреймворках как React, Angular и Vue.js. В React компоненты могут быть как функциональными, так и классовыми. Первые проще в написании и тестировании. Для расширения функциональности используются специальные функции - hooks.

Ценность программного обеспечения заключается в двух аспектах: поведении и структуре. Поведение – это соответствие требованиям, определенным набором тестов разного уровня. Структура, или архитектура, обеспечивает простоту внесения изменений в поведении компьютеров.

# Защита программного обеспечения.

Программные продукты и компьютерные базы данных являются предметом интеллектуального труда специалистов высокой квалификации. Процесс проектирования и реализации программных продуктов характеризуется значительными материальными и трудовыми затратами, основан на использовании наукоёмких технологий и инструментария, требует применения и соответствующего уровня дорогостоящей вычислительной техники. Это обусловливает необходимость принятия мер по защите интересов разработчика программ и создателей компьютерных баз данных от несанкционированного их использования.

Программное обеспечение является объектом защиты также и в связи со сложностью и трудоёмкостью восстановления его работоспособности, значимостью программного обеспечения для работы информационной системы. Защита программного обеспечения преследует цели:

* ограничение несанкционированного доступа к программам или их преднамеренное разрушение и хищение;
* исключение несанкционированного копирования (тиражирования) программ.
* Программный продукт и базы данных должны быть защищены по нескольким направлениям от воздействия:
* человека – хищение машинных носителей и документации программного обеспечения; нарушение работоспособности программного продукта и др.;
* аппаратуры – подключение к компьютеру аппаратных средств для считывания программ и данных или их физического разрушения;
* специализированных программ – приведение программного продукта или базы данных в неработоспособное состояние (например, вирусное заражение), несанкционированное копирование программ и базы данных и т. д.

Самый простой и доступный способ защиты программных продуктов и базы данных – ограничение доступа. Контроль доступа к программному продукту и базе данных строится путем:

* парольной защиты программ при их запуске;
* использования ключевой дискеты для запуска программ;
* ограничения программ или данных, функций обработки, доступных пользователям, и др.

Могут также использоваться и криптографические методы защиты информации базы данных или головных программных модулей.

Программные системы защиты от несанкционированного копирования предотвращают нелицензированное использование программных продуктов и баз данных. Программа выполняется только при опознании некоторого уникального не копируемого ключевого элемента.

Таким ключевым элементом могут быть:

* дискета, на которой записан не подлежащий копированию ключ;
* определённые характеристики аппаратуры компьютера;
* специальное устройство (электронный ключ), подключаемое к компьютеру и предназначенное для выдачи опознавательного кода.
* Программные системы защиты от копирования программных продуктов:
* идентифицируют среду, из которой будет запускаться программа;
* устанавливают соответствие среды, из которой запущена программа, той, для которой разрешен санкционированный запуск;
* вырабатывают реакцию на запуск из несанкционированной среды;
* регистрируют санкционированное копирование;
* противодействуют изучению алгоритмов и программ работы системы.

Для идентификации запускающих дискет применяются следующие методы:

* нанесение повреждений на поверхность дискеты («лазерная дыра»), которая с трудом может быть воспроизведена в несанкционированной копии дискеты;
* нестандартное форматирование запускающей дискеты.

Идентификация среды компьютера обеспечивается за счёт:

* закрепления месторасположения программ на жёстком магнитном диске (так называемые неперемещаемые программы);
* привязки к номеру BIOS (расчёт и запоминание с последующей проверкой при запуске контрольной суммы системы);
* привязки к аппаратному (электронному) ключу, вставляемому в порт ввода-вывода, и др.

В последнее время большую значимость приобретают методы правовой защиты программных продуктов и баз данных.

Правовые методы защиты программ включают:

* патентную защиту;
* закон о производственных секретах;
* лицензионные соглашения и контракты;
* закон об авторском праве.

Различают две категории прав:

* экономические права, дающие их обладателям право на получение экономических выгод от продажи или использования программных продуктов и баз данных;
* моральные права, обеспечивающие защиту личности автора в его произведении.

# ЗАКЛЮЧЕНИЕ

В результате прохождения производственной практики были закреплены теоретические знания, получены новые профессиональные навыки и умения, также были выполнены поставленные задачи и получен положительный опыт.

Производственная практика помогла мне научиться планировать и грамотно распределять поставленные задачи. Я освоил и закрепил новые умения и навыки, на практике применил полученные в ходе освоения теоретические знания.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. https://habr.com/ru/articles/775574/
2. https://znanio.ru/media/harakteristiki-programmnogo-obespecheniya-2535455
3. https://studref.com/424231/informatika/sredstva\_zaschity\_programmnogo\_obespecheniya
4. https://rtmtech.ru/articles/soft-license-protection/