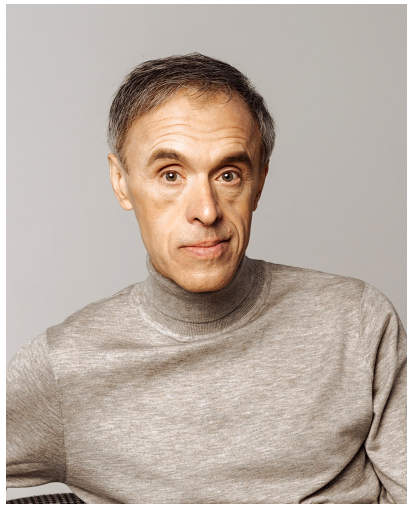
КОНСОМ ГРУПП

Участник

РЕТРОСПЕКТИВА РИСКОВ РАЗРАБОТКИ ПО при СОЗДАНИИ ПРОМЫШЛЕННЫХ СИСТЕМ АСУ

Спикер: Волщуков Ю.Н. – к.т.н., CIO

О ВЫСТУПАЮЩЕМ



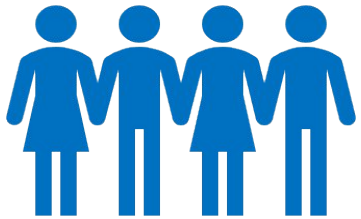
38 лет профессиональной деятельности

- Вузовское образование – автоматизация технологических процессов в горно-металлургической отрасли
- Более 30 лет принимает участие в проектах по разработке систем автоматизации и управления производственно-экономической деятельности предприятий различной формы собственности и направлений деятельности
- 25 лет преподавательской деятельности в направлениях автоматизации и программирования промышленных систем
- Сфера научных интересов – моделирование физико-химических процессов на металлургическом производстве
- Свыше 50 публикаций, патентов, авторских свидетельств по профессиональной деятельности

КОНСОМ ГРУПП

О КОМПАНИИ

ВЕДУЩИЙ РОССИЙСКИЙ СИСТЕМНЫЙ ИНТЕГРАТОР



С 1995 по настоящее время

ВЫПОЛНЕНО БОЛЕЕ 2500 ПРОЕКТОВ

БОЛЕЕ 60 ПРОЕКТОВ СТОИМОСТЬЮ СВЫШЕ 1 МЛН ЕВРО

ВЕДУЩИЙ РОССИЙСКИЙ СИСТЕМНЫЙ ИНТЕГРАТОР

БОЛЕЕ 350 СОТРУДНИКОВ

- Монтажники-строители
- Инженеры -сметчики
- Инженеры связи и сетевой инфраструктуры
- Инженеры АСУТП
- Программисты КФС
- Системные архитекторы
- Системные аналитики
- Аналитики по работе с нейро-сетевыми алгоритмами
- Бизнес аналитики
- Программисты
- Тестировщики
- Инженеры пуско-наладки

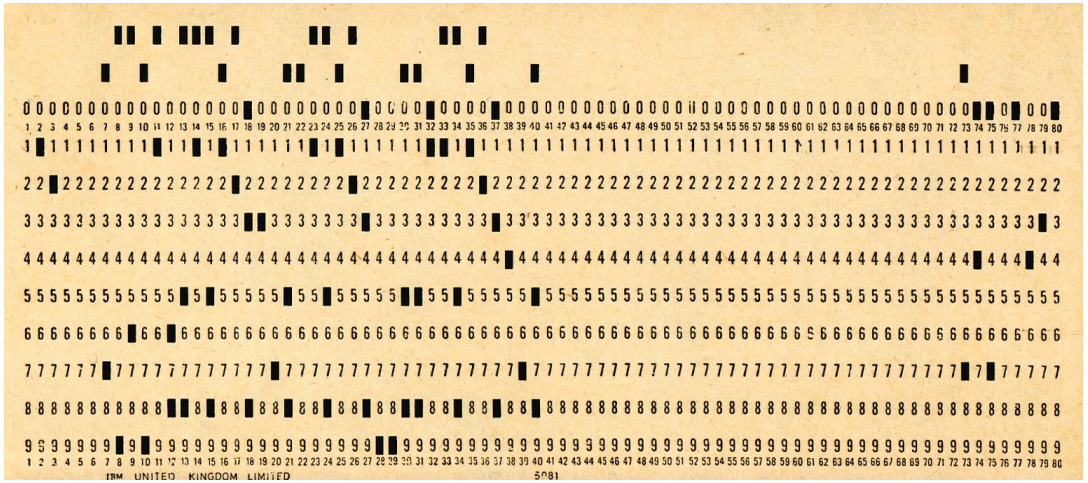
ОТРАСЛЕВЫЕ КОМПЕТЕНЦИИ

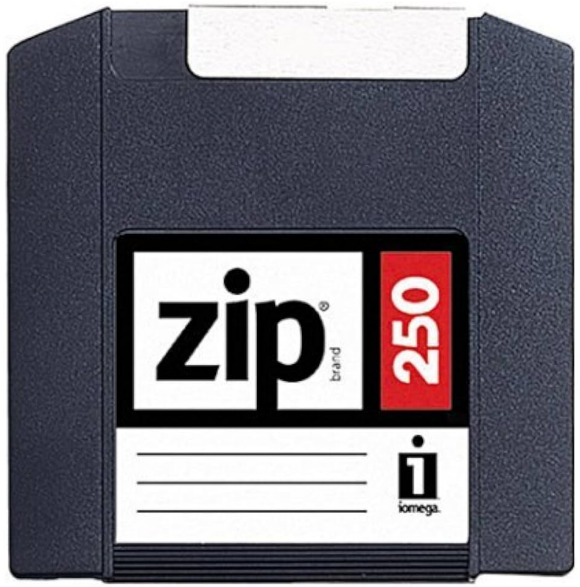
- Автоматизация предприятий Черной металлургии
- Автоматизация предприятий Цветной металлургии
- Автоматизация предприятий Горнорудной промышленности
- Автоматизация предприятий Химической промышленности
- Автоматизация предприятий Деревообрабатывающей промышленности
- Автоматизация предприятий Энергетической промышленности
- Автоматизация предприятий Газо-нефтеперерабатывающей промышленности
- Металлургическое машиностроение – конструкторское бюро

Предмет разговора

1. Программное обеспечение. Что покупаем ?
2. Ключевые отличия программного обеспечения от других продуктов, товаров. Пробуем на «зуб».
3. Про мышление бизнеса и программиста.
4. Краткое ретро трансформации программного обеспечения.
5. Влияние трансформации ПО на риски бизнеса.
6. Что делать ?

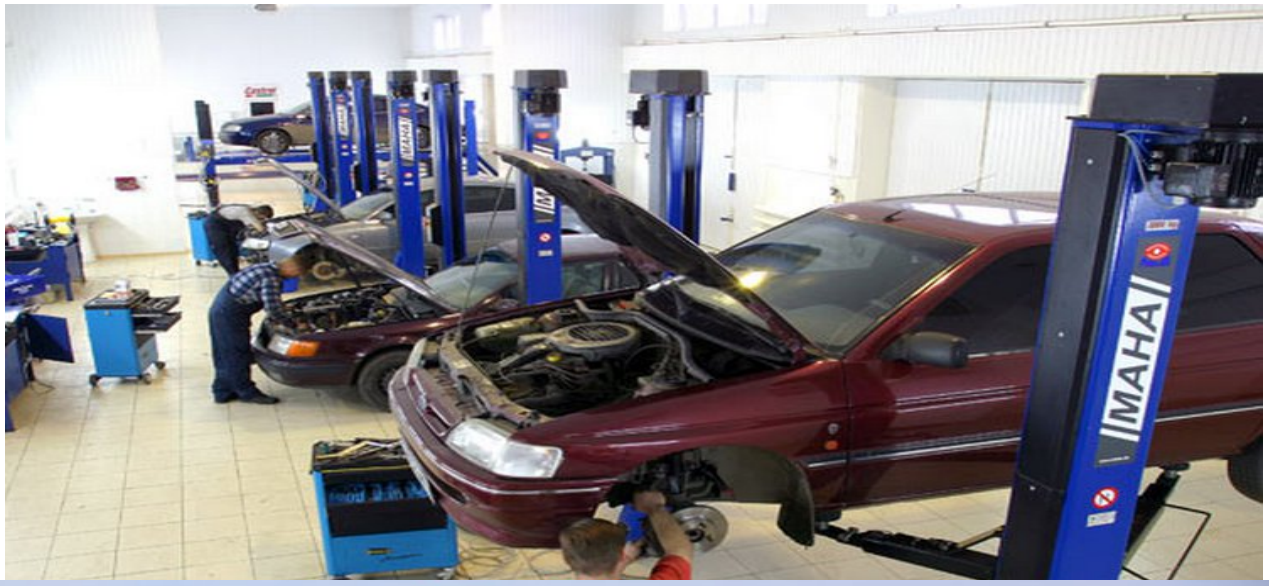
Хранилище кода

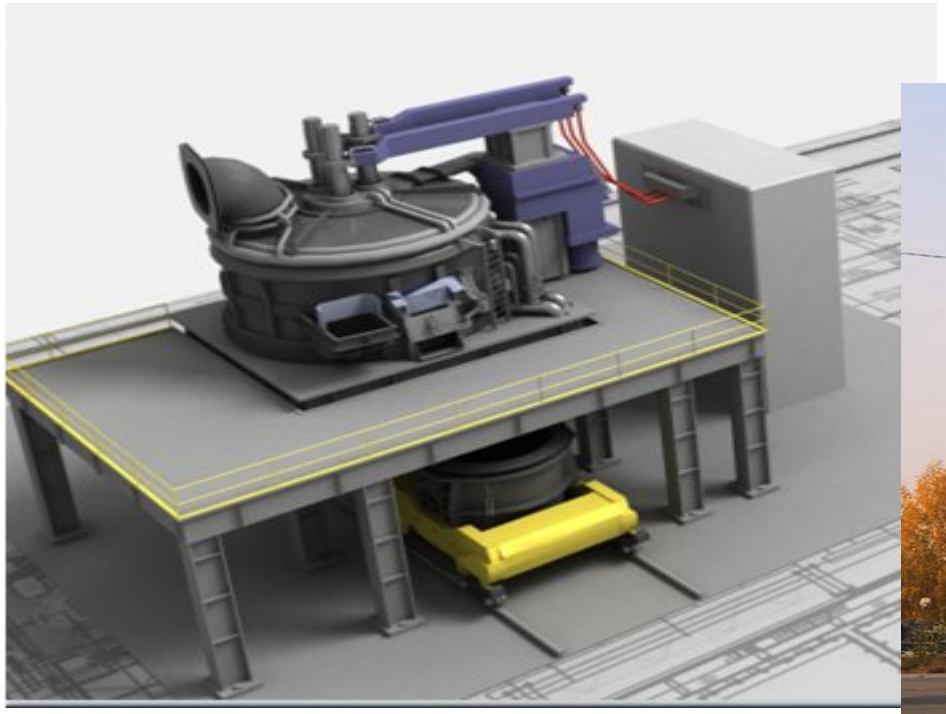





```cpp
// classes.cpp: определяет точку входа для консольного приложения.

#include "stdafx.h"
#include <iostream>
using namespace std;

class CppStudio // имя класса
{
private: // спецификатор доступа private
    int day, // день
        month, // месяц
        year; // год
public: // спецификатор доступа public
    void message() // функция (метод класса) выводящая сообщение на экран
    {
        cout << "\nwebsite: cppstudio.comntheme: Classes and Objects in C + +\n";
    }
    void setDate(int date_day, int date_month, int date_year) // установка даты в формате дд.мм.гг
    {
        day   = date_day; // инициализация день
        month = date_month; // инициализация месяц
        year  = date_year; // инициализация год
    }
    void getDate() // отобразить текущую дату
    {
        cout << "Date: " << day << "." << month << "." << year << endl;
    }
}; // конец объявления класса CppStudio

int main(int argc, char* argv[])
{
    setlocale(LC_ALL, "rus"); // установка локали
    int day, month, year;
    cout << "Введите текущий день месяц и год!\n";
    cout << "день: ";     cin >> day;
    cout << "месяц: ";    cin >> month;
    cout << "год: ";  cin >> year;
    CppStudio objCppstudio; // объявление объекта
    objCppstudio.message(); // вызов функции класса message
    objCppstudio.setDate(day, month, year); // инициализация даты
    objCppstudio.getDate(); // отобразить дату
    system("pause");
    return 0;
}
```

Реальная жизнь, реальные объекты





1. **Что** может быть произведено? (определение доступных ресурсов).

2. **Как** производить? (определение как делать продукт).

3. **Когда** и в **Какой** последовательности производить? (определение расписания).

4. **Когда** и **Что** было произведено? (определение производительности).

Мышление со стороны искусственных систем

Информационная система

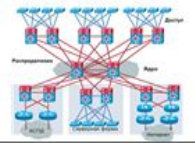
1. Техническое обеспечение

2. Программное обеспечение

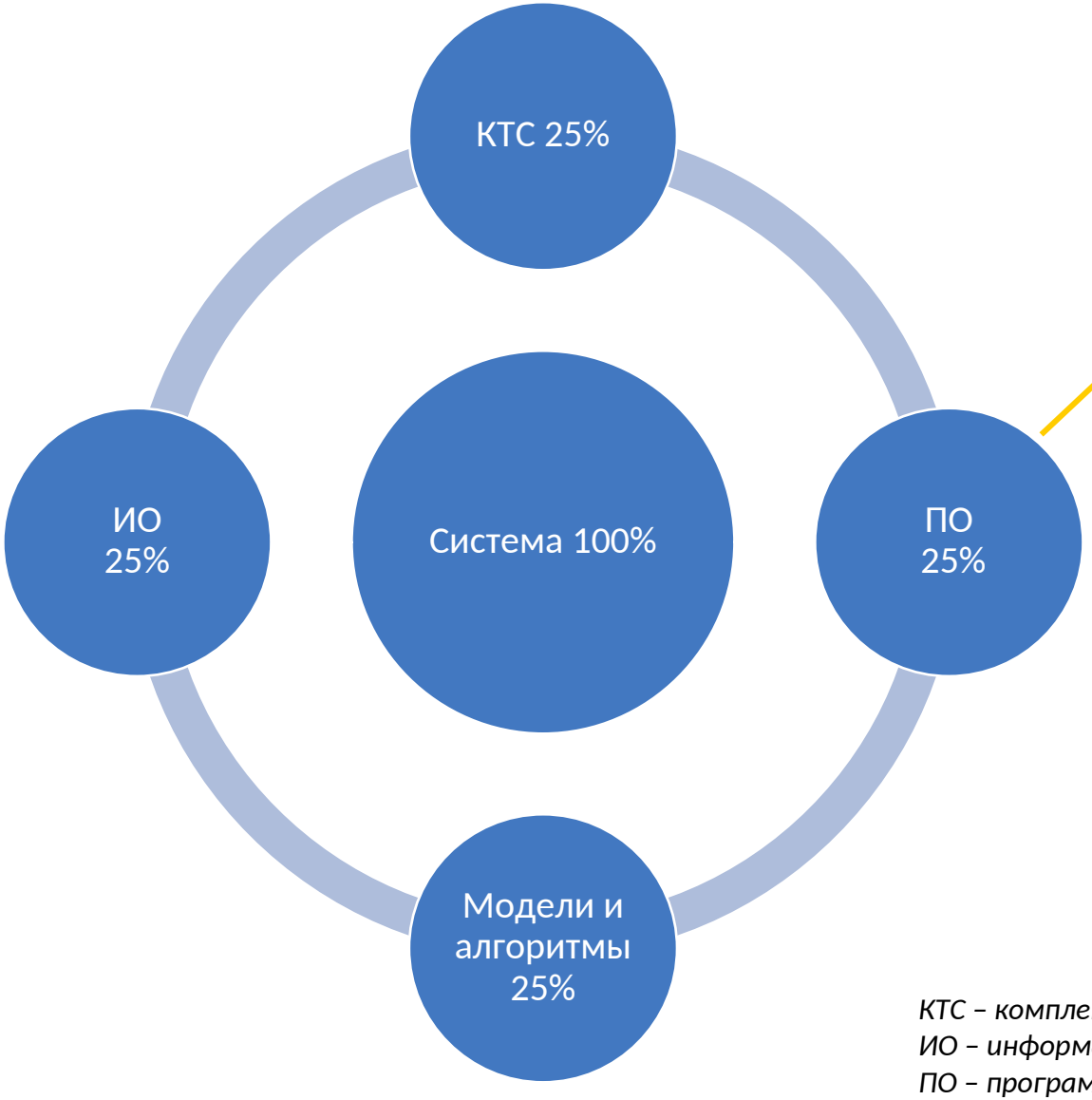
3. Математическое обеспечение

$$(\sqrt{u})' = \frac{u'}{2 * \sqrt{u}}$$

4. Сетевая инфраструктура



Разработка под требования бизнеса



30% - Базовый код
70% - Пользовательский код

1. Форма
2. Бизнес-Правила
3. Бизнес-Логика
4. Роли

Элементы модели

КТС – комплекс технических средств
ИО – информационное обеспечение
ПО – программное обеспечение

Ретро: от 80-ых прошлого до 20-ых текущего 100летия

1. *Языки программирования:* В этот период широко использовались языки программирования, такие как C, Pascal и Fortran. На производстве было кодирование.
2. *Ограниченные ресурсы:* Компьютеры в то время были ограничены по мощности и памяти, поэтому разработчики много времени тратят на оптимизацию кода, о моделях только заикались.
3. *Отсутствие графического интерфейса*: Большинство приложений работали в **текстовом режиме** без графического интерфейса пользователя.
4. Локальное программирование: **Разработка** программного обеспечения происходила **отдельно каждым**, а совместная работа и обмен кодом были ограничены.

1. *Развитие интернета*: В этот период Интернет стал все более распространенным, что привело к возможности разработки веб-приложений и **клиент-серверных систем**
2. *Объектно-ориентированное программирование*: Методология объектно-ориентированного программирования стала широко применяться, и появились языки, такие как Java и C++, которые поддерживали этот подход.
3. *Повышение производительности*: Компьютеры стали более мощными и доступными, что позволило разработчикам создавать **более сложные и производительные программы**
4. *Среды разработки:* В этот период стали появляться интегрированные **среды разработки** (IDE), которые предоставляют разработчикам инструменты для разработки, отладки и тестирования кода.

1. *Веб-разработка:* **Веб-приложения** стали все более популярными, и появились различные фреймворки и технологии для разработки веб-сайтов и веб-приложений, такие как PHP, ASP.NET, Ruby on Rails и другие.
2. *Мобильное программирование:* С появлением смартфонов и планшетов разработка **мобильных приложений** стала важной областью. Языки и фреймворки, такие как Objective-C, Java для Android и Swift, стали популярными в этой области
3. *Облачные вычисления:* **Облачные платформы и сервисы** предоставили новые возможности для разработки, развертывания и масштабирования приложений

1. *ИИ и машинное обучение:* Искусственный интеллект и машинное обучение предоставляют инструменты для разработки и обучения моделей различного рода
2. *Интернет вещей (IoT):* Разработка ПО для устройств IoT появляется. Разработчики создают приложения, которые взаимодействуют с различными устройствами, такими как датчики, умные дома и промышленные системы.
3. *Контейнеризация и микросервисы:* Контейнеризация и микросервисная архитектура становятся популярными подходами к разработке и развертыванию программного обеспечения. Технологии, такие как Docker и Kubernetes, облегчают управление и масштабирование приложений

Резюме

Мы приспосабливаемся и живем в условиях нового техногенного ландшафта
И при этом живём в стереотипах мышления по отношению к ПО:

А именно:

1. Чем больше размер у программы, тем дороже можно её оценивать.
2. Чем больше документации, а особенно если она по ГОСТу выполнена, то это залог качественного проектирования на создаваемое программное обеспечение.
3. Чем больше данных мы соберём в «озера данных», тем эффективнее будет деятельность предприятий.
4. Нам аналитика не предлагайте, дайте программиста. Мы ему объясним, что надо делать.

Специализация в области ИТ

1985

Оператор ЭВМ
Инженер программист
Программист баз данных

2020

Количество специальностей в ИТ насчитывает более 50.

Это:
Программисты различного стека
Дизайнеры графические
UX – разработчики
Системные администраторы
ИТ-безопасность
Бизнес-аналитики
Системные аналитики
Аналитики больших данных
DevOps специалисты
Специалисты машинного обучения

ПРИЗМА ИЗМЕНЕНИЙ

Ключевые отличия программного кода (как продукта/товара) от других видов:

Виртуальная природа. Программный код создает виртуальную систему, которая не имеет физической формы. Это отличает его от проектирования, например, зданий или машин.

Итеративность. Проектирование ПО - это итеративный процесс с постоянным тестированием и улучшением. Это позволяет быстро вносить изменения по сравнению с физическими объектами.

Абстракции. Программный код использует высокоуровневые абстракции, такие как классы, функции и объекты. Это позволяет проектировать сложные системы более структурированно.

Человеческий фактор. Пользовательский интерфейс и удобство использования играют большую роль при проектировании ПО. Этот аспект менее значим при проектировании, например, мостов.

Документирование. Проектирование ПО требует подробной документации архитектуры, кода и дизайна. Это отличает его от некоторых других видов проектирования.

Быстрое внедрение. Программный код можно быстро внедрить и запустить после проектирования. Этап реализации короче для ПО по сравнению, например, со строительством.

Что влияет на качество и количество программного обеспечения

1. **Недостаток опыта.** Если команда разработчиков не имеет достаточного опыта в создании программного продукта, то это может привести к низкому качеству кода, ошибкам в проектировании и реализации, несоблюдению стандартов и лучших практик.
2. **Переоценка возможностей.** Неверный выбор состава и размера команды, что приводит к переоценке своих возможностей по скорости разработки и как следствие может привести к провалу или значительным задержкам.
3. **Трудности в планировании.** Без достаточного опыта может быть сложно правильно оценить объем работ, сроки и затраты на проект. Это часто приводит к перерасходу бюджета и сроков.
4. **Недостаток тестирования.** Маленькая команда не может уделять достаточного внимания тестированию, что может привести к выходу продукта с багами.
5. **Отсутствие управления проектом.** Неопытная команда может столкнуться со сложностями в организации, планировании и управлении проектом.
6. **Зависимость от ключевых сотрудников.** Уход одного опытного сотрудника может нанести серьезный урон проекту. И часто просто катастрофический.
7. **Ограниченные ресурсы.** Маленькая команда имеет меньше финансовых и человеческих ресурсов для разработки, что повышает риски.

Классификация бизнес рисков и их трансформация на современном этапе

1. **Бизнес риски –** риски связанные с операционной деятельностью предприятия, Они могут включать в себя технические сбои, проблемы с поставками, брак продукции, увольнения персонала, изменения в законодательстве и регулировании, а также другие факторы, которые негативно влияют на процессы производства и поставок
2. **Финансовые** – риски, связанные с финансовыми потерями в результате осуществления деятельности предприятия
3. **Риски рыночной среды -** эти риски связаны с изменениями на рынке, такими как изменения в потребительском спросе, конкуренция, технологические изменения, изменения в политике или макроэкономические сдвиги
4. **Репутационные риски** - эти риски связаны с ущербом репутации компании, вызванным негативными событиями, скандалами, недобросовестным поведением сотрудников
5. **Регуляторные риски** - эти риски связаны с невыполнением компанией законодательных требований, нормативов или стандартов, что может привести к штрафам, судебным искам, утрате лицензий или другим негативным последствиям

КОНСОМ ГРУПП

ВОПРОСЫ

КОНСОМ ГРУПП

Участник

СПАСИБО ЗА ВНИМАНИЕ

INFO@KONSOM.RU

KONSOM.RU

8 800 268 05 48