

|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ  Федеральное государственное бюджетное образовательное учреждение высшего образования  **«Московский технологический университет»**  **МИРЭА** | | |
|  | | |
| Институт информационных технологий  Кафедра №234 УЭВМ | | |
|  | **РАБОТА ДОПУЩЕНА К ЗАЩИТЕ** | |
|  | Заведующий кафедрой | |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
|  | *Ф.И.О.* | |
|  | «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г. | |
|  |  |  |
| **ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА** | | |
| по направлению подготовки бакалавров \_\_\_\_\_ \_\_\_\_\_\_\_\_ | | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| На тему: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
|  | | |
| Обучающийся | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гаев И.А. |
|  | *подпись* |  |
| Шифр | 13И0264 |  |
| Группа | ИВБО-5-13 |  |
| Руководитель работы | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Глухов А.В. |
|  | *подпись* |  |
| Консультант по специальной части | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Баранов И.А. |
|  | *подпись* |  |
| Консультант по экономической части | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Белоусова И.В. |
|  | *подпись* |  |
|  | Москва 2017 |  |

**ТЗ стр 1**

**ТЗ стр 2**

**Аннотация**

В данной работе рассматривается и проводится разработка прикладного программного обеспечения (Приложения) для системы автоматизированного проектирования (САПР) Beremiz.

Приложение построено по архитектуре «клиент-сервер», что подразумевает его использование как сетевого приложения (Веб-приложения) с вычислительной частью на стороне сервера и интерфейсом пользователя на стороне клиента. Разработка осуществляется с помощью современных средств программирования: высокоуровневым языком Python и языком для написания сценариев JavaScript. Совместное использование данных средств разработки позволит создать современное сетевое приложение на основе большого количества библиотек и технологий с выбором подходящей архитектуры веб-приложения.

Приложение будет использоваться в сфере автоматизации производства для оптимизации процесса проектирования. В его задачи входит решение следующих проблем: влияния человеческого фактора при тестировании, увеличения времени разработки из-за многократного рутинного ввода и изменения сигналов, проверки работы ПЛК и исполняемой на нём программы в опасных производственных сценариях. Веб-приложение предоставляет возможность эмулировать значения каналов модулей ввода/вывода, используя окно браузера.

**Оглавление**

[Определения, обозначения и сокращения](#_17dp8vu) 5

[Введение](#_3dy6vkm) 7

[1. Литературный обзор](#_3pmlgjub0zau) 9

[1.1. Программируемые логические контроллеры](#_tjnt9ulb417) 9

[1.2. Стандарт МЭК 61131-3](#_3j2qqm3) 10

[1.3. Система проектирования](#_1y810tw) 14

[1.2.1. IsaGRAF](#_2xcytpi) 14

[1.2.2. CODESYS](#_2p2csry) 15

[1.2.3. Beremiz](#_3fwokq0) 17

[2. Основная часть](#_206ipza) 20

[2.1. Язык программирования](#_4k668n3) 20

[2.1.1. Java](#_2dlolyb) 21

[2.1.2. Ruby](#_3q5sasy) 22

[2.1.3. Python](#_2iq8gzs) 23

[2.1.4. Javascript](#_haapch) 25

[2.2. Веб-приложение](#_3ep43zb) 26

[2.2.1. Сервер](#_1tuee74) 28

[2.2.2. Клиент](#_1yyy98l) 31

[2.3. Работа с данными](#_4anzqyu) 35

[3. Экономическая часть](#_2pta16n) 37

[Заключение](#_14ykbeg) 41

[Список использованных источников](#_3oy7u29) 42

[Приложение 1. Код программы](#_qm5cz94v50hv) 45

# 

# 

# Определения, обозначения и сокращения

В настоящей работе используются следующие термины с соответствующими определениями.

МЭК (Международная Электротехническая Комиссия) - основанная в 1906 году организация, которая занимается разработкой стандартов электрических и электронных технологий;

ОС (Операционная Система) - системное программное обеспечение для управления ресурсами ЭВМ;

ПК (Программируемый Контроллер), или ПЛК (Программируемый Логический Контроллер) - цифровая электронная система, предназначенная для применения в промышленных условиях. ПК использует программируемое запоминающее устройство для внутреннего хранения ориентированных на пользователя инструкций, для выполнения специальных функций, таких как логические, упорядочения, отсчета времени, математические действия, управление через цифровые или аналоговые входы и выходы различными типами механизмов или процессов. ПК и связанные с ним периферийные устройства разрабатывают так, чтобы они могли быть легко интегрированы в промышленную систему управления (ГОСТ Р 51840-2001);

ПО (Программное Обеспечение) - совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ (ГОСТ 19781-90);

САПР (Система Автоматизированного Проектирования) - комплекс средств автоматизации проектирования, взаимосвязанных с необходимыми подразделениями проектной организации или коллективом специалистов (пользователем системы), выполняющий автоматизированное проектирование (ГОСТ 22487-77);

СМ ЭВМ (Система Малых Электронно-Вычислительных Машин) семейство управляющих ЭВМ, создаваемых с 1974 года [1];

СУБД (Система Управления Базами Данных) - программное обеспечение для создания и работы с базами данных;

ТП (Технологический процесс) - совокупность операций (стадий) по производству, упаковке и контролю продукции (ГОСТ Р 56639-2015);

УВК (Управляющий Вычислительный Комплекс) - иерархическая многоуровневая автоматизированная система управления производством в реальном времени с распределенной обработкой данных на нижнем уровне и с много-терминальным доступом пользователей на средних и верхних уровнях [1];

УСО (Устройство Связи с Объектом) - устройства, реализующие работу с дискретными и аналоговыми сигналами датчиков и исполнительных механизмов [1];

AJAX (Asynchronous Javascript and XML) - технология обновления части веб-страницы без её полного обновления;

DOM (Document Object Model) - инструмент для предоставления доступа к отдельным частям документа - объектам, определяемым тегами;

PEP (Python Enhancement Proposal) - руководство о том, как писать код для языка python;

SQL (Structured Query Language) - язык программирования для реализации запросов к БД;

WSGI (Web Server Gateway Interface) - стандарт взаимодействия приложения и сервера.

# 

# Введение

Основой успешного производства всегда являлось его непрерывное развитие. Предприятия и предприниматели в этой сфере должны были всегда повышать эффективность своей деятельности, иначе бы уступали своим конкурентам и теряли востребованность. Обязательным элементом любой деятельности раньше, как и сейчас, являлся человек. Возможности предприятия определялись количеством и возможностями каждого человека, а также применяемыми технологиями. С их развитием до промышленного масштаба появилась возможность развивать эффективность производства без увеличения численности сотрудников, через внедрение данных технологий - человек же становится рядом с процессом производства в качестве его контролера и регулировщика. Но машины стали заменять не только физический труд, позже появились устройства и приборы для счёта, регистрации происходящих технологических процессов (ТП). Современная вычислительная техника является как раз следствием развития таких машин. В 60-х появились приборы способные автономно выполнять задачи регулирования и контроля - промышленные контроллеры. Механизация производства сменялась его автоматизацией. Таким образом, огромный интеллектуальный труд человека был значительно заменен вычислительными мощностями машины. Первые контроллеры использовались в автомобильной промышленности для автоматизации сборочных линий и обладали жёсткой логикой, то есть настраивались аппаратно. Они были предшественниками современных программируемых логических контроллеров (ПЛК), которые настраиваются программно, с возможностью записи, при необходимости, иных инструкций.

На данные устройства загружаются прикладные программы - алгоритмы мониторинга и управления, а для создания таких программ используются системы автоматизированного проектирования (САПР). Во время разработки программ для ПЛК, прорабатываются возможные сценарии развития событий в технологическом процессе, а современному технологу необходимо неоднократно проверять разработанный алгоритм. Вручную создаются и подаются сигналы, отрабатываются различные производственные и технологические сценарии. В данный процесс вмешивается человеческий фактор, который повышает вероятность ошибки или просчета - увеличивается длительность разработки. Сюда же относится тестирование работы прикладной программы во время нештатных ситуаций. Это означает, что отработка некоторых случаев, например, аварии на атомной станции, сильно затруднена.

Такой объём работы также требует автоматизации, как в своё время этого требовал ручной труд человека или вопросы производственного контроля.

# 

# 

# 1. Литературный обзор

## 1.1. Программируемые логические контроллеры

Непосредственно программируемыми логическими контроллерами называются устройства, автоматизирующие работу производственных комплексов. ПЛК функционирует в постоянном цикле и его работа зависит от сведений, поступающих с датчиков, собирающих информацию о процессе. Полученные данные определяют работу прикладной программы, записанной на контроллер, и обратных команд (рисунок 1).



Рисунок 1.1. Цикл работы ПЛК

Программируемые контроллеры серьезно повысили эффективность управления ТП и обеспечили необходимую надежность и срок службы. ПЛК широко применяются в системах управления, решая локальные задачи участка технологического процесса и обеспечивая трансляцию данных о процессе в системы диспетчерского контроля и управления [2].

Если стандартный компьютер является достаточно универсальным устройством с широким спектром задач, то ПЛК бывают крайне разнообразными и серьезно различаются в зависимости от функциональных задач производства. Контроллеры могут представлять из себя моноблочный элемент с интерфейсными разъёмами, распределенную структуру или шкаф с модулями и возможностью горячей замены элементов: модулей ввода/вывода, модулей процессора или питания и других модулей (рисунок 2).

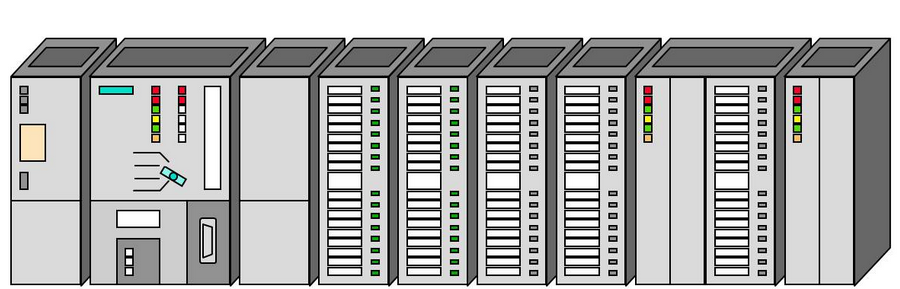


Рисунок 1.2. Заменяемые модули ПЛК

Производством ПЛК занимаются такие компании как Siemens (Simatic S7), Mithsubishi Electric (Melsec FX), Schneider Electric (TAC Xenta), ИНЭУМ (СМ1820М), Beckhoff (Embedded PC), Prosoft Systems (REGUL RX00). Каждые обладают следующими отличительными особенностями: количество вводов/выводов, конструктив, расположение модулей ввода/вывода, поддерживаемые интерфейсы, область применения и прочее.

Современные контроллеры программируются согласно стандарту МЭК 61131-3.

## 1.2. Стандарт МЭК 61131-3

Языки стандарта, как и сам стандарт, появились с несколькими целями. Во-первых, планировалось сделать доступным написание программы управления для инженера-технолога, который незнаком с высокоуровневыми языками программирования, ведь именно он лучше всего понимает принцип функционирования. Во-вторых, общий стандарт формирует единые требования к подготовке специалистов и новые сотрудники, пришедшие из других фирм, работающих с контроллерами, без проблем смогут начать свою работу. В-третьих, стандарт включает 5 достаточно разных языков, которые можно использовать отдельно или вместе, что позволит перевести старые контроллеры без проблем на новую программную базу, в соответствии с идеологией открытых систем.

В состав этого стандарта входят следующие языки:

1) Диаграммы SFC (Sequential Function Chart) - графическая диаграмма SFC состоит из шагов и переходов между ними; разрешение перехода определяется условием; с шагом связаны определенные действия; описания действий выполняются на любом языке МЭК (рисунок 3).



Рисунок 1.3. Язык SFC

2) IL (Instruction List) - это список инструкций, текстовый язык, похожий на ассемблер (машинно-ориентированный язык программирования, расширенный конструкциями языков программирования высокого уровня), позволяет работать с любыми типами данных, вызывать функции и функциональные блоки, реализованные на любом языке (рисунок 4).

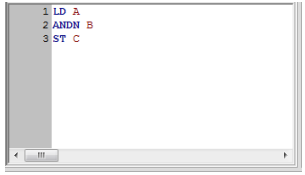


Рисунок 1.4. Язык IL

3) ST (Structured Text) - текстовый язык программирования, синтаксически представляет собой несколько адаптированный язык Паскаль (высокоуровневый процедурный язык программирования); вместо процедур Паскаля в ST используются компоненты программ стандарта МЭК (рисунок 5).



Рисунок 1.5. Язык ST

4) Диаграммы FBD (Function Block Diagram) - графический язык из функциональных блоков со входами и выходами, напоминает принципиальную схему электронного устройства на микросхемах (рисунок 6).

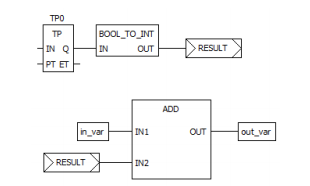


Рисунок 1.6. Язык FBD

5) Диаграммы LD (Ladder Diagram) - графический язык, реализующий структуры электрических цепей; LD-диаграмма представлена в виде двух вертикальных шин питания, а между ними расположены цепи, образованные соединением контактов (рисунок 7).

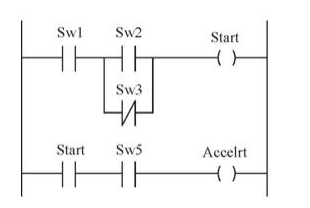


Рисунок 1.7. Язык LD

Программа, написанная на языках стандарта МЭК, разбивается на множество функциональных элементов - Program Organization Units (POU), каждый из которых может состоять из функций, функциональных блоков и программ. Имеются средства для исполнения разных фрагментов программы в разное время, с разной скоростью, а также параллельно.

Разработку на языках МЭК 61131-3 поддерживают большинство производителей контроллеров, оснащая свои САПР соответствующими возможностями.

## 1.3. Система проектирования

Для решения задач автоматизации работ на стадии проектирования и подготовки используют системы автоматизированного проектирования (САПР, в английском CAD - Computer-Aided Design). Использование САПР значительно повышает эффективность труда инженеров и контроллеры практически всех компаний сопровождаются ими. Крупные САПР для ПЛК: IsaGRAF, CoDeSys, Beremiz. Каждое из этих решений отличается количеством составных элементов (библиотек, приложений, программ), языками программирования, списком поддерживаемых устройств и архитектур, наличием качественной документации или учебных программ.

### 1.2.1. IsaGRAF

IsaGRAF - САПР компании Rockwell Automation, состоит из трёх частей: среды разработки приложений IsaGRAF Workbench (рисунок 8), исполнительной системы IsaGRAF Target и средства разработки драйверов под IsaGRAF. САПР может одновременно работать с аппаратными платформами под управлением разных ОС.



Рисунок 1.8. IsaGRAF

В данной САПР реализована возможность разработки как на языке C и языках МЭК 61131-3, так и с помощью функциональных блоков МЭК 61499. В МЭК 61499 функциональные блоки отличаются от аналогичных в МЭК 61131-3 наличием диаграммы управления выполнением (ECC - Execution Control Chart). ECC отвечает за реакцию функционального блока на внешние воздействия. Таким образом, общая циклическая модель исполнения программы в МЭК 61131-3 заменяется функциональным блоком, генерирующим периодические события, воздействующие на функциональные блоки. Помимо этого, к пяти языкам МЭК 61131-3 добавлен язык FC (Flow Chart), который является графическим языком. В FC диаграмма состоит из уникальных действий и тестов (рисунок), соединённых связями.



Рисунок 1.9. Действие и тест

Действия программируются на языках ST, LD или IL, а тест представляет собой булевские условия с развилкой связей [5].

### 1.2.2. CODESYS

CODESYS - крупная САПР, которая представляет из себя целую платформу CODESYS Development System из множества решений, включая: редактор языков МЭК 61131-3, прикладные библиотеки, отладчик и прочее (рисунок 10). Реализована возможность объектно-ориентированного и функционального программирования, присутствуют графические редакторы языков МЭК LD и FBD. В программе есть большое количество уже настроенных модулей контроллеров, готовых к использованию сразу после установки.



Рисунок 1.10. Платформа CODESYS

CODESYS поддерживает возможность соединения по протоколам Profibus, Ethercat, Canopen, Modbus и ряд других, при используется одно и тоже средство настройки (рисунок 11).

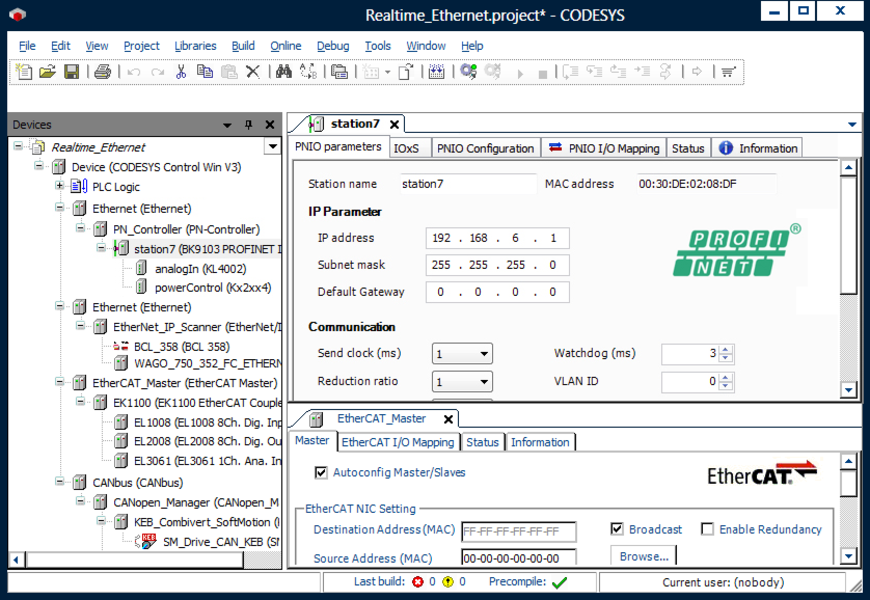


Рисунок 1.11. Настройка соединения

Помимо непосредственно средств проектирования, производитель предлагает большое количество тренировочных программ для улучшенной интеграции своего ПО (рисунок 12) и список устройств на отдельном сайте, работающих с CODESYS, от разных производителей [6].

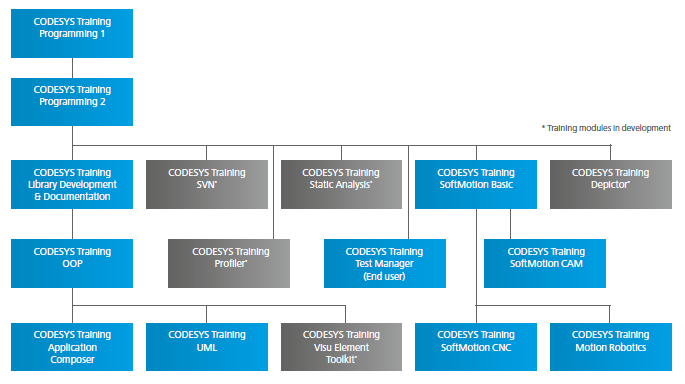


Рисунок 1.12. Тренировочные программы CODESYS

### 1.2.3. Beremiz

Beremiz является свободно-распространяемой САПР с открытым исходным кодом, написанной на языке Python (рисунок 13). Используется как среда разработки прикладных программ для отечественных контроллеров СМ1820М, а также любых устройств под управлением ОС Windows, Linux или ОС Эльбрус [7].

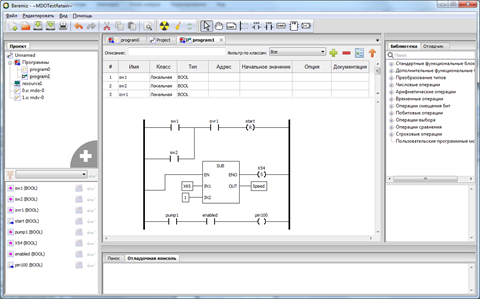


Рисунок 1.13. Beremiz

Проект в Beremiz состоит из нескольких модулей POU (unit), которые написаны на языках МЭК 61131-3, хранится в отдельной папке и представлен в формате XML. Файловая структура проекта представлена следующими компонентами:

* файл с названием проекта и промежуточным кодом на языке ST;
* файлами с алгоритмами и логикой на языке C;
* файл с кодом отладки;
* файл, который содержит код для целевой архитектуры;
* файлы с кодом драйверов для модулей УСО;
* исполняемый файл, компилируемых из остальных файлов проекта.

Средствами Beremiz исполняемый бинарный файл размещается на целевом устройстве [4].

Beremiz позволяет работать в двух режимах: проектирования алгоритмов управления и отладки прикладной программы. В первом случае создается программа, алгоритмы и логика, связываются с внешними модулями УСО (устройство связи с объектом). Во втором прикладная программа передаётся на целевое устройство и может быть запущена с режимом отладки и без отладки.

САПР позволяет проверить работу программы удаленно, получая сигналы с объекта управления.

Даже в небольшом промышленном производстве единовременно присутствует десятки и сотни аналоговых и дискретных сигналов. Проверка работы прикладной программы с ручным изменением этих сигналов очень рутинный и долгий процесс, требующий повышения эффективности и оптимизации. С учетом возможности Beremiz проверить работу прикладной программы в реальном времени, замена набора реальных сигналов с объекта управления на аналогичный набор искусственных сигналов, редактируемых в человеко-машинном интерфейсе, повысит эффективность тестирования прикладной программы и позволит проводить его ещё до выпуска контроллера, для которого программа разрабатывается.

В связи с этим целесообразно и необходимо разработать приложение-эмулятор сигналов модулей ввода/вывода ПЛК, в виде расширения для САПР Beremiz.

# 2. Основная часть

Разрабатываемое ПО будет использоваться программистом в паре с САПР Beremiz (рисунок 1), позволяя изменять значения аналоговых и цифровых сигналов и мгновенно проверять работу прикладной программы.



Рисунок 2.1. Рабочий процесс

Выбор языка программирования зависит от задач разрабатываемого ПО и возможностей используемого языка.

## 2.1. Язык программирования

Высокоуровневые языки программирования ориентированы по своему виду и структуре на восприятие человеком, а не машиной, которая его выполняет. Необходимо рассмотреть современные языки программирования на предмет возможности реализации эмулятора.

Согласно данным крупнейшей платформы для разработки ПО GitHub за 2016 год, в тройке языков программирования по количеству запросов на включение этого языка в новые проекты, находятся: Javascript, Java, Python, Ruby [8].

### 2.1.1. Java

Программы, написанные на данном языке транслируются в специальное промежуточное представление - набор инструкций для виртуальной машины Java (рисунок 2). Благодаря этому, разработка Java-программ может вестись на любой ОС и компьютерной архитектуре, главное чтобы на них была среда исполнения JRE (Java Runtime Environment). Это и стало причиной популярности для разработки мобильных приложений.



Рисунок 2.2. Тестовая программа на языке Java

Тем не менее, использование промежуточного представления кода делает его исполнение дольше, чем у аналогичной программы, написанной на предшественнике языка Java Си.

Концептуально язык является объектно-ориентированным, то есть элементы программы являются объектами, у которых есть поля и методы. Также язык характерен сильной системой типов - принудительным приведением типа объекта к необходимому, и явным объявлением переменных.

Java активно поддерживается - последняя версия 8 вышла в марте 2014. Поддержку осуществляет крупный производитель ПО - компания Oracle. По индексу TIOBE, который оценивает популярность языка на основе запросов из множества поисковых сервисов, начиная с 2015 года по текущий момент, Java является самым популярным языком программирования [9].

### 2.1.2. Ruby

Высокоуровневый язык Ruby изначально создавался как объединение особенностей и возможностей многих других языков для широкого функционала и просто синтаксиса.

В языке Ruby тип переменной определяется в момент присваивания значения (динамическая типизация), что позволяет менять тип переменной по ходу выполнения программы. Как и Java, Ruby является объектно-ориентированным: все функции являются методами, а все классы получены из класса Object. Для формирования блоков не обязательно использовать фигурные скобки или конструкции begin..end, что делает язык визуально чистым (рисунок 3).



Рисунок 2.3. Пример кода на языке Ruby

Язык очень похож на Python и с точки зрения мультипарадигмальности (использование более чем одной парадигмы программирования), и с точки зрения внешнего вида, а также свободной лицензии. Работает на ОС Linux, Windows, Mac OS, Dos и ряде других. Для данного языка существует популярный инструмент разработки интернет-приложений - Ruby on Rails [10].

### 2.1.3. Python

Python - высокоуровневый объектно-ориентированный язык программирования с простым и чистым синтаксисом. В Python предусмотрена возможность работы в интерактивном режиме, с построчным выполнением (рисунок 4).

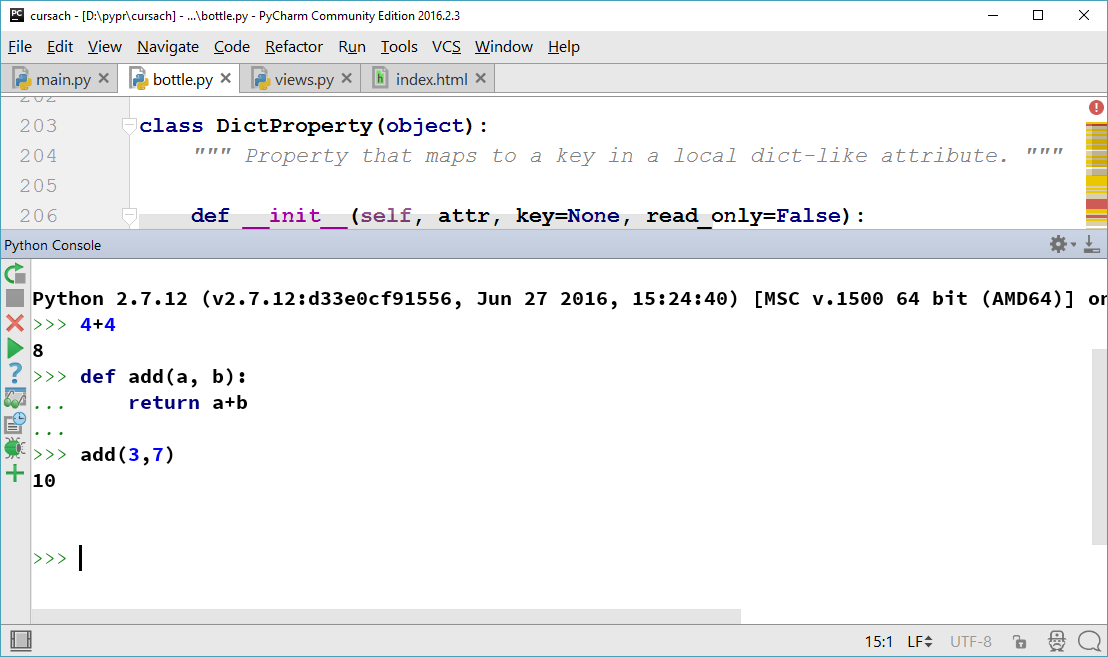


Рисунок 2.4. Интерактивный режим

Он работает практически на всех платформах, включая мобильные ОС. Существует версия под названием Jython, призванная запускать код Python средствами виртуальной машины Java, Cython для трансляции в код Си и множество других сторонних инструментов и реализаций языка Python.

Список операторов стандартный для почти всех высокоуровневых языков: условный оператор if, другие условие elif, оператор альтернативы else, операторы цикла while и for, прерывания и перехода break и continue, оператор определения класса class, функции def, генератора yield, возврата return, цепочки обработки исключений try-except-finally и оператор для создания пустых блоков pass.

Основной особенностью языка является его легкая читаемость. Это достигается во многом благодаря выделению блоков кода с помощью отступов (рисунок 5), взамен выделения скобками или операторами begin..end. Для Python существует специальная документация PEP 8, которая стандартизирует правила написания кода для сохранения “правильного” вида.

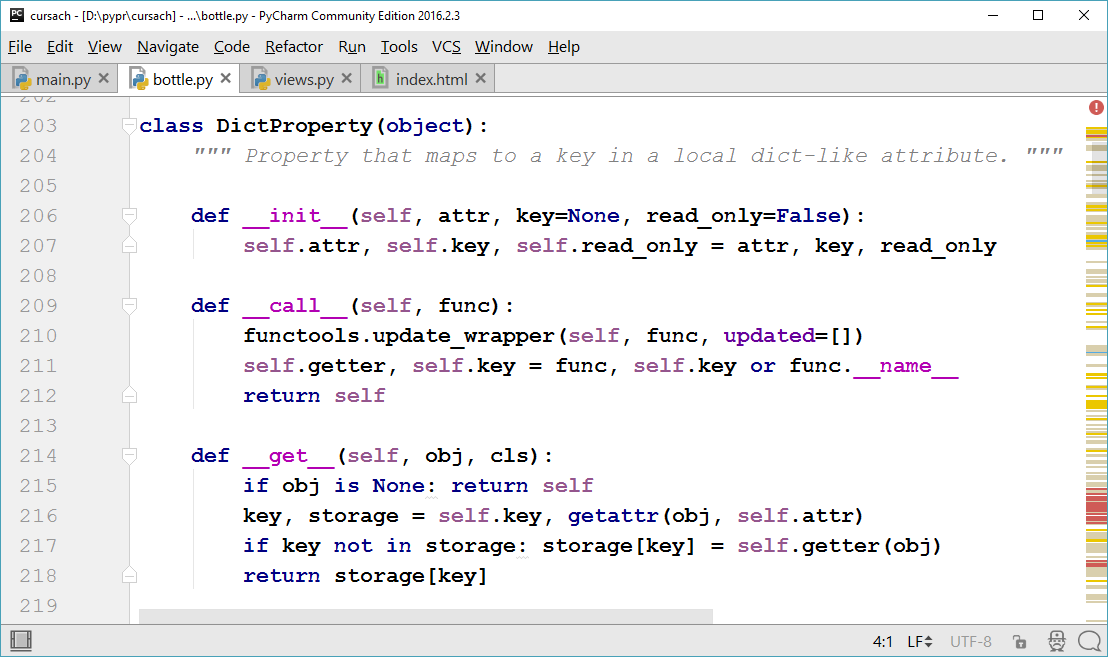


Рисунок 2.5. Табуляция перед функциями и их содержимым

Реализованы возможности использования декораторов, итераторов, обработка исключений, лямбда-функций, что позволяет писать в парадигме функционального программирования, регулярные выражения и много другое.

Язык активно поддерживается сообществом программистов, поэтому существует большое количество модулей расширения и программных интерфейсов, среди которых:

- пакеты доступа к различным СУБД: Oracle, MySQL, SQLite;

- программный интерфейс для создания сервера (приложения, посылающего ответы на запросы другого приложения, называющегося «клиент») WSGI (Web Server Gateway Interface);

- программный интерфейс C API для написания модулей на языке Си;

- библиотеки для создания графических интерфейсов tkinter и wxPython;

- веб-фреймворки для разработки интернет-приложений: Django, Flask, Pylons, Bottle [11].

Благодаря использованию веб-фреймворков возможно реализовать сервер, но клиентская логика реализуется другими языками, например, Javascript.

САПР Beremiz реализована как раз на высокоуровневом языке Python. Разрабатывать приложение-эмулятор для существующей системы в целях долгосрочного использования и возможности развития следует на том языке программирования, который использовался для создания соответствующего ПО.

### 2.1.4. Javascript

Главное отличие Javascript от вышеописанных языков - это сфера его применения: он используется для создания интерактивных HTML-страниц (документ со специальной разметкой, который преобразуется интернет-браузером в понятный человеку вид). Спецификация языка называется ECMAScript, пятая версия которой поддерживается всеми браузерами, а шестая постепенно интегрируется.. JS может применятся и не в браузере, но главное его преимущество - это полная интеграция с HTML (рисунок 6) и CSS (язык для описания внешнего вида разметки HTML и не только), что и определяет его популярность и лидерство в своей сфере применения.



Рисунок 2.6. Код JavaScript в HTML-документе

Интерпретаторы JS встроены во все основные браузеры. Современные интерпретаторы не выполняют скрипты JS сразу, а преобразуют его в машинный или подобный ему код, оптимизируют и уже затем выполняют.

Из-за изначальной ориентации на выполнения скриптов HTML-страниц, язык не имеет средств низкоуровневых языков программирования, таких как работа с памятью и процессоров. Но с точки зрения работы как скриптового языка он предоставляет широкий ассортимент возможностей для работы с HTML и CSS, средствами ввода, данными (в границах браузера, JS не имеет прямого доступа к ОС) и запросами сервера. Это во многом определяет использование JS для создания веб-приложений [12].

## 2.2. Веб-приложение

Сетевые, или веб-приложения - ПО, основаны на вычислительной архитектуре «клиент-сервер», где клиентом является браузер, а запросы выполняются по HTTP-протоколу. Как правило, сервер и клиент находятся на разных ЭВМ, что позволяет клиентским пользователям использовать вычислительные мощности сервера.

HTTP (HyperText Transfer Protocol) - протокол прикладного уровня по модели OSI (Open Systems Interconnection), регулирующего передачу данных при помощи запросов и ответов, которые формируются приложениями-клиентами и приложениями-серверами. Протокол работает в рамках архитектуры «клиент-сервер» [13].

Логика приложения распределяется между клиентом и сервером, причем структура этого распределения зависит от решаемой задачи. Для работы простых одностраничных веб-сайтов не требуется большая вычислительная мощность и большинство действий осуществляется скриптами JS, загружаемыми на клиент и там же выполняющимися. Для работы больших сайтов, включающих операции с множеством данных, состоящих из нескольких страниц, требующих постоянных запросов на сервер, необходима большая вычислительная мощность. Обмен данными между клиентом и сервером, расположенными на разных ЭВМ, осуществляется интернет-соединением (рисунок 7).



Рисунок 2.7. Обмен данными между клиентом и сервером

В таких приложениях клиенты не зависят от ОС пользователя, тем самым веб-приложения автоматически становятся мощными межплатформенными сервисами, а также снижаются требования к вычислительной мощности клиента, в связи с чем его роль могут исполнять ЭВМ практически любой мощности (рисунок 8).



Рисунок 2.8. Веб-приложение на разных ЭВМ

К минусам данного вида ПО можно отнести зависимость работы приложения от наличия и качества интернет-соединения.

С повышением скорости интернет-соединений, многие веб-приложения стали вытеснять локальное ПО, ведь при наличии хорошей сети нет необходимости, к примеру, скачивать карту местности – она подгружается сервером в режиме реального времени и отображается клиенту. Помимо этого, веб-приложение может быть запущено на любой ЭВМ, где есть браузер, а он присутствует в базовом комплекте ПО любой современной ОС.

В связи с этим, можно сделать вывод о преимуществе разработки именно веб-приложения, взамен локального. Такое приложение позволит работать с сигналами вне зависимости от платформы и не потребует создания нескольких версий приложения для работы с Beremiz на разных ОС.

### 2.2.1. Сервер

Специально для создания веб-сервера для Python разработано более десятка специальных библиотек - веб-фреймворков, и даже принят стандарт WSGI (Web Server Gateway Interface) в рамках документации PEP, который определяет работу большинства python-серверов.

Django - веб-фреймворк, который помогает в разработке как малых веб-приложений, так и крупных. Среди полезных возможностей, которые есть в Django, но нет в других фреймворках, можно отметить следующие:

- обращение к базам данных с помощью своего ORM;

- встроенная панель администратора (рисунок 9);

- огромное количество встроенных модулей [14].

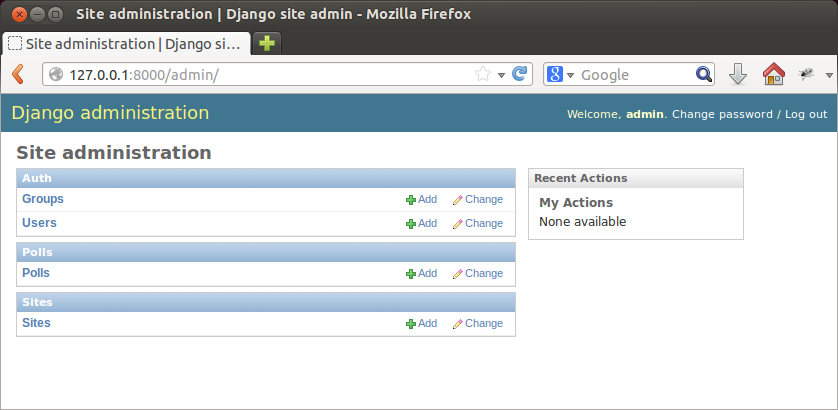


Рисунок 2.9. Встроенная панель администратора Django

Flask представляет другую сторону развития функциональности своего фреймворка, через большое количество расширений, причем команда разработчиков утверждает, что одобренные расширения обязательно будут работать и в будущем. В качестве стандартного способа взаимодействия с базами данных используется SQLAlchemy, но есть возможность использовать и другие БД. Для удобной работы с отображением страницы используется система шаблонов Jinja2, которая, фактически, представляет из себя целую библиотеку с возможностями по работе с шаблонами: наследование, макросы, настраиваемый синтаксис. Собственно преимущество Flask перед Django в том, что разработчик веб-приложения использует в своём проекте ровно столько возможностей, сколько ему необходимо, а в случае необходимости - использует расширение, в то время как Django является массивной целостной сборкой [15].

Существует около десятка веб-фреймворков, которые отличаются количеством базовых возможностей, особенностей, способов работы с БД, но все они значительно объемнее фреймворка Bottle. Он объединяет приемлемый набор базовых возможностей по созданию веб-сервера в один только файл - Bottle.py. Для развертывания сервера необходимо сделать минимальное количество действий (рисунок 10), предварительно подключив файл библиотеки в проект [16].

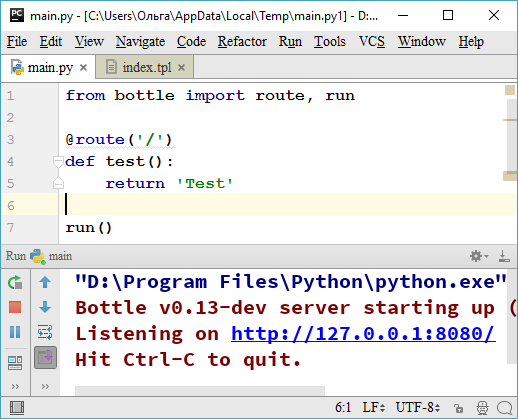


Рисунок 2.10. Запуск элементарного веб-сервера

Таким образом, bottle является самым компактным из существующих веб-фреймворков и требует меньше всего действий для реализации сервера, что отлично подходит для одностраничных веб-сайтов и решения небольших задач средствами веб-приложений, аналогичных веб-эмулятору по масштабу.

Для хранения данных на стороне сервера используются базы данных. В языке Python есть встроенная библиотека sqlite3, написанная на языке C, которая обеспечивает работу с БД на языке SQL (рисунок 2.11).

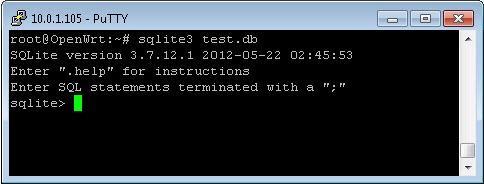


Рисунок 2.11. SQLite

Данный инструмент относится к реляционным (табличным) системам управления базами данных. Большинство баз данных SQL работает по схеме клиент/сервер.

Таким образом, за итоговую реализацию веб-эмулятора со стороны сервера будут отвечать следующие инструменты: python, веб-фреймворка bottle и СУБД SQLite. Данного набора должно быть достаточно для выполнения запросов клиента.

### 2.2.2. Клиент

Основной компонент клиента - это используемый пользователем браузер. Во все современные браузеры встроен интерпретатор языка Javascript, что позволяет использовать его как универсальное средство разработки клиентской части веб-приложения.

Клиентская часть отвечает за отображение веб-страницы, которая состоит из структуры HTML-тегов, таблицы стилей (CSS) и скриптов на языке Javascript, которые отвечают за поведение страницы (рисунок 12).

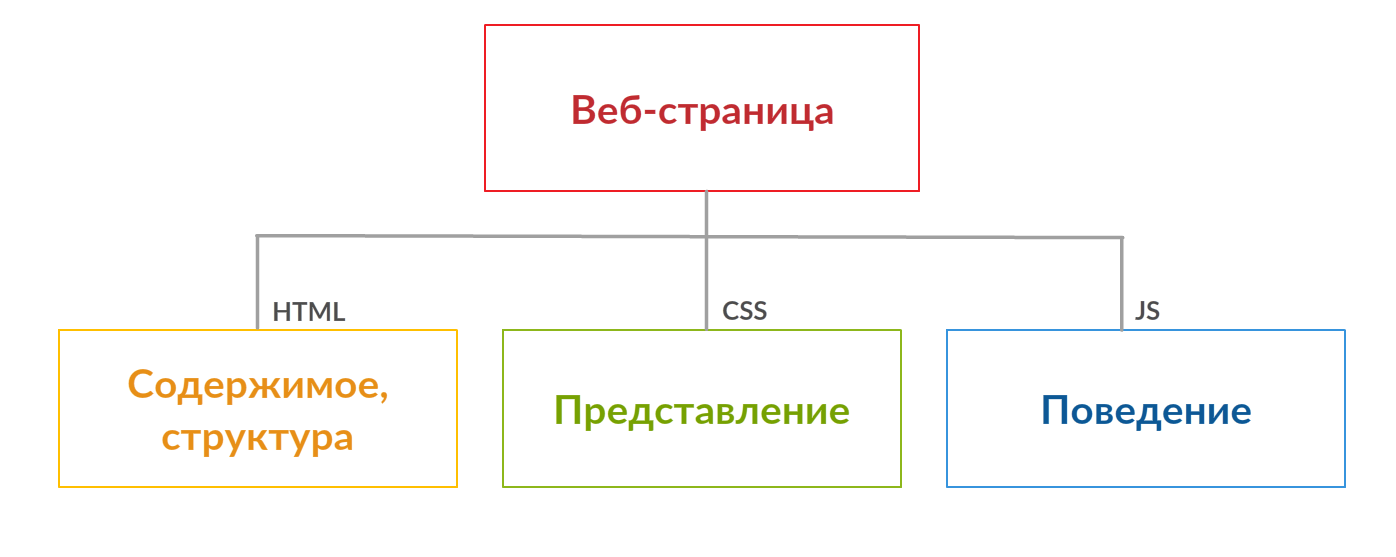


Рисунок 2.12. Составные элементы реализации веб-страницы

CSS-представление можно прописывать вручную, но как и в любых языках, существует большое количество готовых библиотек/фреймворков, предоставляющих широкие возможности программного решения любых вопросов. Bootstrap является фреймворком для удобной верстки и гарантии одинакового отображения страницы во всех браузерах. Благодаря этому фреймворку реализуются адаптивные сайты для любых устройств, а широкий и подробный инструментарий помогает ускорить разработку веб-страниц. В частности есть возможность отрисовки адаптивной сетки, применения готовых стилизованных элементов при помощи назначения классов (рисунок 13), разнообразных компонентов (включая js-реализацию их поведения) и скачивания готовых шаблонов веб-страниц.

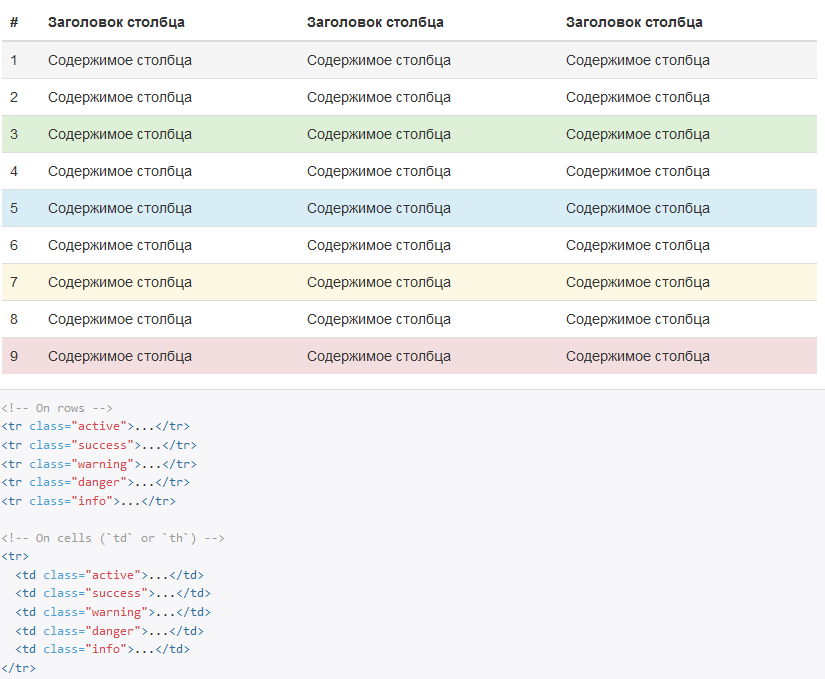


Рисунок 2.13. Базовые стили применяются через указание класса

Для реализации модели поведения может использоваться библиотека JQuery, которая осуществляет взаимодействие JavaScript и HTML. Ее использование предоставляет простой доступ к любому элементу DOM (Document Object Model - программный интерфейс, не зависящий от платформы и языка, позволяющий программам и скриптам получить доступ к содержимому HTML-документов, а также изменять содержимое, структуру и оформление таких документов.), обращаться к их атрибутам и содержимому (рисунок 14), манипулировать ими.

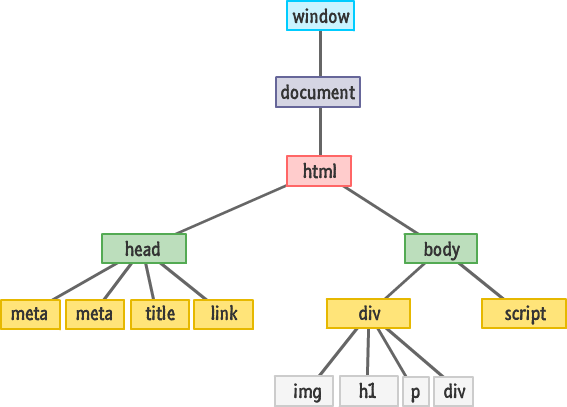


Рисунок 2.14. Архитектура DOM

Также библиотека JQuery предоставляет набор готовых компонентов для работы с AJAX (Asynchronous Javascript And Xml), методом обмена данными браузера с сервером в «фоновом» режиме (рисунок 15). В результате, при обновлении данных HTML-страница не перезагружается полностью, и веб-приложения становятся быстрее и удобнее.



Рисунок 2.15. AJAX

Итоговая структура веб-эмулятора со стороны клиента будет выглядеть как представлено на рисунке (рисунок 16).



Рисунок 2.16. Клиентская часть приложения

## 2.3. Работа с данными

Приложение является расширением Beremiz и его устройство зависит от среды проектирования. Если в нормальном режиме данные берутся из драйвер, то САПР в режиме эмуляции формирует на основе используемых модулей конфигурационный файл и считывает значения сигналов из бинарного файла, персонального для каждого модуля ввода.

Во время параллельной работы прикладной программы в Beremiz и эмулятора используются общие данные. Для согласования работы с ними и организации межпроцессорного взаимодействия используется бинарный файл. Возможны и альтернативные способы организации использования общих данных, такие как: именованные канал, семафор или использование разделяемой памяти [17].

Веб-приложение оперирует со значениями сигналов модулей ввода/вывода. Прежде чем начать с ними работу, необходимо их считать, преобразовать в используемый приложением вид. Получается, что входными данными самого приложения является конфигурационный файл модулей ввода/вывода. После работы с данными, они сохраняются в бинарные файлы - по одному на каждый модуль, в соответствии с конфигурационным файлом. Все сигналы дискретного модуля хранятся в одном 4-байтном значении, а каждый аналоговый сигнал - в 2-байтном. Результат работы приложения - бинарные файлы (рисунок 17).



Рисунок 2.17. Данные на входе и выходе

Изменение данных в приложении выполняется человеком в окне браузера. Там, разработчик видит уже сами сигналы, их тип, методы работы с ними и выборки сигналов (рисунок 18).

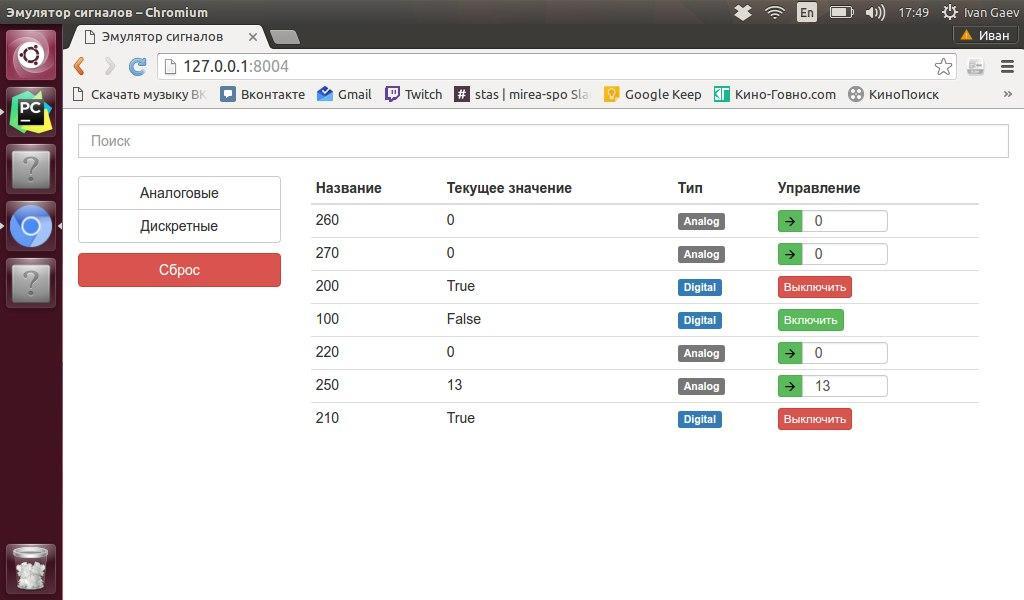


Рисунок 2.18. Окно браузера

В итоге перед разработчиком удобный интерфейс работы с сигналами, который можно использовать на любой ОС. Использование вместе с Beremiz позволит существенно ускорить процесс тестирования прикладных приложений для ПЛК и даже начать проектировать прикладные приложения для тех ПЛК, которые еще в процессе производства.

# 3. Экономическая часть

Данная работа посвящена разработке веб-приложения для редактирования аналоговых и цифровых сигналов

1) Разработка основных разделов бизнес-плана.

Приложение предназначено для программистов, занимающихся проектированием в среде Beremiz в сфере промышленной автоматизации. Промышленная автоматизация необходима для повышения эффективности работы предприятия любого размера. Программное обеспечение является прикладным для САПР Beremiz, которая, в свою очередь, свободно-распространяемая. В связи с этим, и разрабатываемое приложение также будет свободно-распространяемым и узкоспециализированным. Приложение разрабатывается как часть другого программного обеспечения (или как его модификация), поэтому его распространению способствует распространение основного ПО, а также рассылка тем клиентам, которые уже используют основное ПО.

В составе проекта задействовано 3 человека (рисунок 1):



Рисунок 3.1. Состав проекта

2) Организация и планирование работ по теме (таблица 1).

Таблица 1 - Организация работ по теме

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Название этапа** | **Исполнитель, должность** | **Трудоемкость, чел/дни** | **Продолжительность, дни** |
|  | Утверждение технического задания | Руководитель | 1 | 1 |
|  | Анализ библиотек и фреймворков языков Python и JS | Консультант | 5 | 10 |
| Инженер | 5 |
|  | Определение архитектуры приложения и организации работы отдельных его модулей | Инженер | 11 | 11 |
|  | Разработка программного кода | Инженер | 11 | 11 |
|  | Тестирование веб-приложения и отладка | Консультант | 3 | 11 |
| Инженер | 8 |
|  | Сдача готового продукта | Инженер | 1 | 1 |
|  | **Итого:** | | **45** | **45** |

Календарный график исполнения работы представлен на рисунке.



Рисунок 3.2. Календарный график исполнения работ

3) Определение договорной цены (таблица 2)

Таблица 2 - Определение договорной цены

|  |  |  |
| --- | --- | --- |
| **№** | **Номенклатура статей расходов** | **Затраты, руб.** |
| 1 | Материалы, покупные изделия и полуфабрикаты | 240 |
| 2 | Специальное оборудование | - |
| 3 | Основная заработная плата | 37736 |
| 4 | Дополнительная заработная плата | 11321 |
| 5 | Страховые отчисления | 14717 |
| 6 | Командировочные расходы | - |
| 7 | Контрагентские услуги | 3120 |
| 8 | Накладные расходы | 75472 |
| 9 | Прочие расходы | 2880 |
| 10 | Полная себестоимость | 145486 |
| 11 | Плановая прибыль | 29097 |
| 12 | Цена предприятия | 174583 |
| 13 | Налог на добавленную стоимость (НДС) | 31425 |
| 14 | Договорная цена | 206008 |

4) Оценка экономической целесообразности проведения работ по теме.

Экономическая целесообразность разработки веб-приложения для редактирования аналоговых и цифровых сигналов заключается в следующем:

- Данный вид программного обеспечения не является самостоятельным и должен восприниматься не как самодостаточный продукт, а как надстройка к уже существующей системе;

- Данное приложение помогает тестировать и отлаживать спроектированные проекты по автоматизации промышленности, что, безусловно является актуальной задачей во всех направлениях производства;

- Благодаря данному проекту, технологу или программисту нет необходимости ездить на объекты, расположенные далеко от места разработки ПО, что сокращает расходы на транспортировку;

- Возможность тестировать опасные производственные случаи и отлаживать аварийные ситуации снижает риски производства, которые могут привести к серьезным непредвиденным затратам;

- Свободное распространение продукта позволяет использовать его любым предприятием, а также помогает сформировать пользовательское сообщество, которое может не только модернизировать продукт и делиться своими разработками в сети, но и помогать в решении проблем другим пользователям;

- Мультиплатформенность сетевых приложений отбрасывает необходимость покупать или разрабатывать дополнительное ПО, в случае несоответствия платформы данному ПО.

Разработка является экономически целесообразной.

# Заключение

В процессе разработки веб-эмулятора аналоговых и цифровых сигналов был проведен обзор сферы промышленной автоматизации, средств и возможностей для ее реализации, проведен полноценный анализ крупных систем автоматизированного проектирования и установлены их преимущества. Установлен ряд проблем, которые присутствуют в системах проектирования, среди которых: долгий процесс тестирования прикладных приложений, ручное изменение сигналов модулей аналогового и дискретного ввода/вывода в процессе тестирования.

В процессе данного обзора установлена возможность и необходимость для разработки дополнительного прикладного программного обеспечения для САПР Beremiz, с целью повышения эффективности процесса проектирования.

Изучение Beremiz и анализ существующих средств разработки программ привели к необходимости реализации веб-приложения средствами языка Python и Javascript. Разработка осуществлялась с использованием современных технологий, библиотек и фреймворков для данных языков, таких как: CSS-фреймворк Bootstrap, python-фреймворк Bottle, СУБД Sqlite, JS-библиотека JQuery, технология Ajax. Создание приложения и использование всех возможностей данных языков программирования проводилось в соответствии Российскими и международными стандартами, а также рекомендации разработчиков данных языков программирования.

Установлена экономическая целесообразность разработки и её результатом стало веб-приложение для САПР Beremiz, с возможностью эмуляции аналоговых и цифровых сигналов в реальном времени.

# Список использованных источников

1. Управляющие вычислительные комплексы для промышленной автоматизации: учебное пособие / Н.Л.Прохорова, В.В.Сюзева; Издательство МГТУ им.Н.Э.Баумана, 2012 - 372 с.

2. Языки стандарта IEC-61131 для вычислительных комплексов на базе отечественных микропроцессоров с архитектурой SPARC / И.А. Баранов, А.В. Глухов; ОАО «ИНЕУМ им. И.С. Брука», 2012.

3. Программируемые контроллеры. Стандартные языки и приёмы прикладного программирования / И.В.Петров, В.П.Дьяконов; СОЛОН-Пресс, 2004 - 256 с.

4. Программные средства поддержки языков программирования высокого уровня стандарта МЭК 61131-3 для ПЛК семейства СМ1820М: Руководство программиста / ПАО «ИНЕУМ им. И.С. Брука»; Литера, 2012 - 213 с.

5. ISAGRAF version 3.5: Руководство пользователя / ICS Triplex IsaGRAF Inc.; 2003 - 452 с.

6. Why CODESYS [Электронный ресурс]. - Режим доступа: <https://www.codesys.com/the-system/why-codesys.html>

7. Система программирования Beremiz [Электронный ресурс]. - Режим доступа: <http://www.sm1820.ru/produktsiya/programmnoe-obespechenie/sistemy-programmirovaniya/item/3-beremiz.html>

8. The state of Octoverse 2016 [Электронный ресурс]. - Режим доступа: <https://octoverse.github.com/>

9. TIOBE Index [Электронный ресурс]. - Режим доступа: <https://www.tiobe.com/tiobe-index//>

10. О Ruby [Электронный ресурс]. - Режим доступа: <https://www.ruby-lang.org/ru/about/>

11. Изучаем Python, 4-е издание / М. Лутц; Символ-Плюс, 2010 - 1280 с.

12. Язык JavaScript [Электронный ресурс]. - Режим доступа: <https://learn.javascript.ru/js>

13. RFC 1945 Hypertext Transfer Protocol HTTP/1.0 [Электронный ресурс]. - Режим доступа: [http://www.rfc-base.org/rfc-1945.html#](http://www.rfc-base.org/rfc-1945.html)

14. Django. The web framework for perfectionists with deadlines. [Электронный ресурс]. - Режим доступа: <https://www.djangoproject.com/>

15. Добро пожаловать во Flask [Электронный ресурс]. - Режим доступа: <http://flask-russian-docs.readthedocs.io/ru/latest/>

16. Bottle: Python Web Framework [Электронный ресурс]. - Режим доступа: <http://bottlepy.org/docs/dev/>

17. Знакомство с межпроцессным взаимодействием на Linux [Электронный ресурс]. - Режим доступа: <https://habrahabr.ru/post/122108/#part_namedpipe>

ГОСТ Р 56639-2015 Технологическое проектирование промышленных предприятий. Общие требования;

ГОСТ 29099-91 Сети вычислительные локальные. Термины и определения;

ГОСТ Р 53623-2009 Информационные технологии. Информационно-вычислительные системы. Комплекты вычислительной техники (компьютерные классы) для общеобразовательных учреждений. Характеристики качества. Технические требования;

ГОСТ Р 51840-2001 Программируемые контроллеры. Общие положения и функциональные характеристики;

ГОСТ 19781-90 Обеспечение систем обработки информации программное. Термины и определения;

ГОСТ 28397-89 Языки программирования. Термины и определения;

ГОСТ 22670-77 Сеть связи цифровая интегральная;

ГОСТ 22487-77 Проектирование автоматизированное. Термины и определения;

ГОСТ Р ИСО/МЭК 7498-1-99 Взаимосвязь открытых систем.

# 

# 

# Приложение 1. Код программы