Оглавление

[Введение 2](#_Toc6782427)

[1. Организационная структура предприятия 3](#_Toc6782428)

[2. Характеристика и описание технологий, использованных в течение практики для разработки прошраммного обеспеченья 5](#_Toc6782429)

[3. Приложение, разработанное в результате преддипломной пракики 7](#_Toc6782430)

[Заключение 15](#_Toc6782431)

[Список использованных источников информации 16](#_Toc6782432)

# Введение

Производственная практика является составной частью учебного процесса подготовки квалифицированных специалистов. Во время практики происходит закрепление и конкретизация результатов теоретического обучения, приобретение студентами умения и навыков практической работы по присваиваемой квалификации и избранной специальности.

Со стороны предприятия должны быть проведены лекции по охране труда и технике безопасности для того, чтобы студент имел представление о возможных рисках и в случае непредвиденной ситуации не паниковал и знал, какие действия следует предпринимать. Студент также должен придерживаться правил внутреннего трудового распорядка.

Во время практики студент должен придерживаться календарного графика, составленного совместно с руководителем практики от предприятия. Руководитель же со своей стороны должен предоставить все необходимые для выполнения задания материалы либо сообщить, в каких источниках их можно найти.

Целью данной преддипломной практики является разработка приложения, позволяющего симмулировать дискретный канал передачи данных с использованием каскадыных кодеков, устраняющих ошибки для оценки их эфективности.

# Организационная структура предприятия

Общество с ограниченной ответственностью "Леверекс Интернешнл" создана в 2003 году и является частью LeverX International – международной компании с штаб-квартирой в г. Маунтин Вью, офисами в Детройте, Вальдорфе, Москве и Минске.

Данная компания является официальным сервисным партнером SAP и ориентирована исключительно на создание информационных систем, а также разработку и внедрение программного обеспечения на платформе программных продуктов и средств разработки компании SAP. Основной целью деятельности компании является повышение отдачи инвестиций заказчиков в решения SAP.

Стратегия развития компании – постоянное расширение спектра оказываемых услуг и их совершенствование с целью создания для клиентов наиболее оптимальных информационных решений на основе SAP. Ключевыми рынками для компании являются Беларусь, Россия, США и Германия. Кроме того, активно ведут деятельность в множестве стран Западной Европы, Азии и Африки.

Численность IT-специалистов компании составляет более чем 350 человек, каждый из которых является специалистом в той или иной области программного обеспечения SAP, что дает возможность LeverX International принимать участие в различных проектах, не ограничиваясь какой-то одной областью разработок с использованием SAP.

Организационная структура компании представлена на рисунке 1.1.

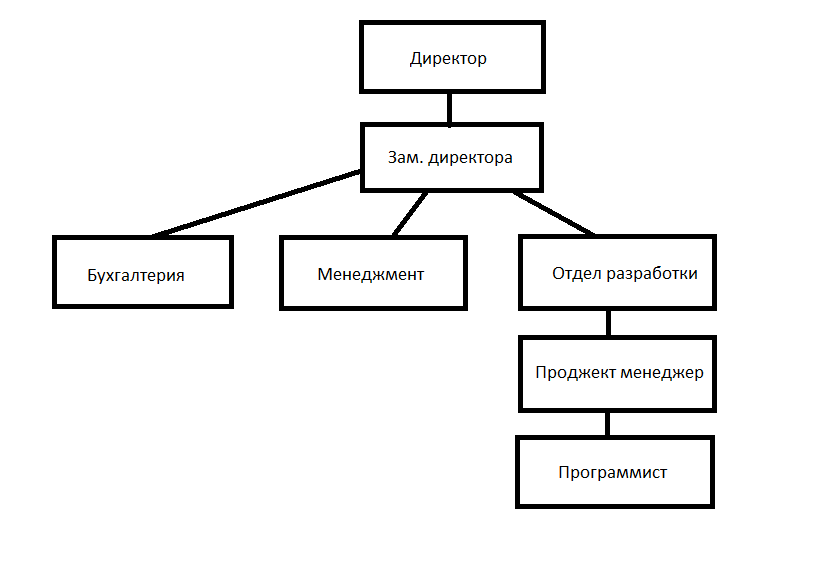


Рис. 1.1 Организационная структура компании

Организационная структура управления представляет собой совокупность органов управления и существующих между ними должностных и информационных связей.

Как видно, организационная структура предприятия по виду передаваемых полномочий является линейной - задачи и полномочия передаются от руководителя к подчиненному и далее к другим подчиненным, образуя при этом уровни управления.

Главным органом управления предприятия, и соответственно первым уровнем управления, является директор. Второй уровень представляет заместитель директора, третий уровень представляют бухгалтерия, менеджмент, отдел разработки. Четвертый уровень управления составляют проджект менеджер и программисты.

Организационная структура управления предприятием характеризуется большим диапазоном управления и небольшим количеством уровней управления. Полномочия и вся информация сосредотачивается в руках руководителя каждого уровня. Это гарантирует быстроту принятия управленческих решений и возможность легкого контроля за работой исполнителей.

# Характеристика и описание технологий, использованных в течение практики для разработки приложения

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Стандартная библиотека включает большой объём полезных функций. Версия языка для написания приложения — 3.7.1. Используемый интерпритатор Сython.

Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений, высокоуровневые структуры данных. Поддерживается создание программ их различных более мелких модулей.

Вышеописанная технология была выбрана по причине имения хороших библиотек для анализа данных и поддержкой высокоуровневых математических функций, предназначенных для работы с многомерными массивами.

Для графического интерфейса программы использовался модуль PyQt5. PyQt — набор «привязок» графического фреймворка Qt для языка программирования Python, выполненный в виде расширения Python.

PyQt разработан британской компанией Riverbank Computing. PyQt работает на всех платформах, поддерживаемых Qt: Linux и другие UNIX-подобные ОС, Mac OS X и Windows. PyQt практически полностью реализует возможности Qt. А это более 600 классов, более 6000 функций и методов, включая:

* существующий набор виджетов графического интерфейса;
* стили виджетов;
* доступ к базам данных с помощью SQL (ODBC, MySQL, PostgreSQL, Oracle);
* QScintilla, основанный на Scintilla виджет текстового редактора;
* поддержку интернационализации (i18n);
* парсер XML;
* поддержку SVG;
* интеграцию с WebKit, движком рендеринга HTML.

PyQt также включает в себя Qt Designer (Qt Creator) — дизайнер графического интерфейса пользователя. Программа pyuic генерирует Python код из файлов, созданных в Qt Designer. Это делает PyQt очень полезным инструментом для быстрого прототипирования. Кроме того, можно добавлять новые графические элементы управления, написанные на Python, в Qt Designer.

Для некоторых сложных вычислений использовался NumPy. NumPy — это open-source модуль для python, который предоставляет общие математические и числовые операции в виде пре-скомпилированных, быстрых функций. Они объединяются в высокоуровневые пакеты. Они обеспечивают функционал, который можно сравнить с функционалом MatLab. NumPy (Numeric Python) предоставляет базовые методы для манипуляции с большими массивами и матрицами. SciPy (Scientific Python) расширяет функционал numpy огромной коллекцией полезных алгоритмов, таких как минимизация, преобразование Фурье, регрессия, и другие прикладные математические техники.

Для графического представления результатов работы программы является граффик отображенный с помощью модуля pandas. Pandas — программная библиотека на языке Python для обработки и анализа данных. Работа pandas с данными строится поверх библиотеки NumPy, являющейся инструментом более низкого уровня. Предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временны́ми рядами. Название библиотеки происходит от эконометрического термина «панельные данные» (англ. panel data), используемого для описания многомерных структурированных наборов информации. Pandas распространяется под новой лицензией BSD. Для корректной работы данной библиотеки необходим модуль tkinter версии выше 2.0.

Так же в приложении для хранения данных использована СУБД PostgreSQL 11 версии. PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД). Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows. Сильными сторонами PostgreSQL считаются:

* высокопроизводительные и надёжные механизмы транзакций и репликации;
* расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl; дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh и PL/V8, а также имеется поддержка загрузки C-совместимых модулей;
* наследование;
* легкая расширяемость.

Для работы приложения с базой данных была использована библиотека SQLAlchemy. SQLAlchemy — это программное обеспечение с открытым исходным кодом для работы с базами данных при помощи языка SQL. Оно реализует технологию программирования ORM (Object-Relational Mapping), которая связывает базы данных с концепциями объектно-ориентированных языков программирования. SQLAlchemy позволяет описывать структуры баз данных и способы взаимодействия с ними прямо на языке Python. SQLAlchemy реализована в виде пакета для Python под лицензией MIT, а значит возможно ее использование в проприетарном ПО.

# 3. Аналитический обзор литературы и постановка задачи;

На текущий момент существуют тенденции построения и эксплуатации распределённых информационных вычислительных систем для решения разного класса задач. Для корректного функционирования распределенных приложений необходимы устойчивые и быстрые каналы связи. Современные каналы передачи данных достаточно быстры и защищены от помех, а дополнительное применение современных помехоустойчивых кодов позволяет свести на нет вероятность неправильной передачи информации. Помехоустойчивое кодирование решает две основные задачи:

* повышение эффективности передачи данных, за счёт устранения избыточности данных в каналах, не содержащих помехи;
* повышение вероятности правильного декодирования пакета в канале с помехами с минимальной дополнительной избыточностью.

В контексте работы рассмотрим вторую задачу применения помехоустойчивых кодов. Целью научной работы является построение имитационной модели передачи информации по каналу с шумами с применением каскадного способа построения кодеков и выявление общих закономерностей и характеристик каскадных кодеков.

В ходе работы необходимо разработать программное средство помехоустойчивого кодека на основе последовательной каскадной схемы. Для разработки будет использована язык python версии 3.7.+. Программное средство должно поддерживать и позволять применить в каскаде четыре вида кодеров (Хемминга, Циклический, Свёрточный, Фонтанный), а также задать их параметры. Разработанная система должна поддерживает использование перемежителей после каждого составного кодера. В функции программного каскадного кодека входит установка разных уровней и видов шумов в канале при передаче кодового сообщения по каналу связи. Существует возможность определить контент передаваемых данных и количество выполняемых тестов для каждого уровня шума.

Далее описаны необходимые минимальные сведения для общего понимания цели данной работы.

## Теория помехоустойчивого кодирования.

Для дальнейшего упрощения понимания контекста данной работы далее будут введены основные понятия и объекты касаемо темы помехоустойчивого кодирования.

**Канал связи –** система технических средств, предназначенная для передачи информации от источника к приёмнику.

**Теория кодирования** — одна из областей математики область действия которой распространяется на передачу данных по зашумленным каналам, а предметом является обеспечение корректности переданной информации.

**Теория кодов, контролирующих ошибки** — **кодирование**, предназначенное для защиты цифровых данных при передаче по каналам связи от появляющихся ошибок.

Теория кодирования изучает возможности модифицирования данных, чтобы

## Формы представления информации. Модель системы передачи информации.

Различают две системы передачи информации:

* дискретная;
* непрерывная.

Сигнал называется непрерывным, если его параметр в заданных пределах может принимать любые промежуточные значения. Сигнал называется дискретным, если его параметр в заданных пределах может принимать отдельные фиксированные значения.

При дискретном канале связи система связи соединяет источник информации с получателем посредством специальных устройств. В системе связи можно выделить основные функции, представленные на рисунке 1.1 ниже.

Сперва данные, которые поступают к источнику обрабатываются кодером источника. Полученные данные после этого этапа именуются кодовым словом источника. Затем данные обрабатываются кодером канала. Кодовое слово канала может быть представлено в виде битов или группы битов. Далее модулятор преобразует данные в аналоговое представление информации в соответствие с протоколом канала передачи данных. При передаче по каналу могут возникнуть шумы, из-за которых информация может подвергнуться искажению.

Демодулированная информация называется принятым словом. Декодер канала используя избыточность кодового слова источника позволяет декодировать принятое слово с некоторыми найденными ошибками.

В данной работе планируется создать имитацию дискретного канала связи с помехами.

  
  
Рисунок 3.1. Блок схема цифровой системы связи

Кодирование – преобразование сообщения в сигнал, т.е. отображение сообщений сигналами в виде определенного сочетания элементарных дискретных символов, называемых кодовыми комбинациями (кодовыми словами).

Код – правило, согласно которому каждому сообщению однозначно ставится в соответствие некоторая кодовая комбинация. Кодер – устройство, осуществляющее кодирование.

Кодер источника (КИ) – кодер, использование которого позволяет путем устранения избыточности существенно снизить среднее число символов на букву сообщения (такое кодирование называется оптимальным или эффективным). При отсутствии помех это дает выигрыш во времени передачи или в объеме ЗУ, т.е. повышает эффективность системы передачи данных.

Кодер канала (КК) – позволяет путем внесения избыточности обеспечить достоверность передачи данных при наличии помех (такое кодирование называется помехоустойчивым).

Канал – совокупность средств, предназначенных для передачи сигнала от передатчика к приемнику информации (передатчик, приемник, линия связи и т.д.). Канал связи может быть односторонний (симплексный) и двухсторонний (дуплексный).

Передатчик – служит для преобразования электрического сигнала в сигнал, пригодный для передачи по линии связи.

Модуляцией называется изменение параметров переносчика сигнала в соответствии с функцией, отображающей сообщение. Несущим сигналом может быть ток (телеграфия), гармонические низкочастотные или высокочастотные колебания (телефония и т.д.), высокочастотные импульсы (радиорелейная связь и т.д.). Модулируемые параметры называются информативными и могут быть амплитудой, частотой, фазой и т.д. Модулятор – устройство, осуществляющее модуляцию.

При передаче по каналу связи происходит ослабление и искажение передаваемого сигнала, вносимых каналом и действием помех.

Линейные искажения – определяются частотными и временными характеристиками канала. Нелинейные искажения – определяются нелинейностью звеньев канала и видом модуляции.

**Определение энтропии с помощью собственной информации**

Можно определить энтропию случайной величины, введя предварительно понятия распределения случайной величины *X*, имеющей конечное число значений: (1.1).

От основания логарифма зависит единица измерения информации и энтропии: бит, трит и другие.

 Интегральный показатель качества функционирования цифровых систем связи. Определяется как отношение количества искаженных битов данных к общему числу переданных битов.

Мера качества передачи. В общем случае выражается отрицательной степенью 10-7, например 10-7 означает 1 ошибку на 107 бит.

**Коэффициент ошибок**— отношение числа неверно принятых битов (0 вместо 1 и наоборот) к полному числу переданных битов при передаче по каналу связи. Эквивалентно понятию вероятности ошибки. В современных сетях связи характерные значения коэффициента - 1E-9 и лучше.

## Математическое выражение коэффициента битовых ошибок

Определим коэффициент битовых ошибок для реальных приёмников, которым свойственно наличие различных источников шумов. При этом будем считать, что приёмник принимает решение, какой бит (0 или 1) был передан в каждом битовом интервале путем стробирования фототока. Очевидно, что из-за наличия шумов данное решение может быть неверным, что приводит к появлению ошибочных битов. Поэтому, чтобы определить коэффициент битовых ошибок, необходимо понять, каким образом приемник принимает решение относительно переданного бита.

Обозначим через I1 и I0 фототоки, стробированные приемником в течение 1 и 0 битов, соответственно, а через s12 и s02 соответствующие шумы. Принимая, что последние имеют гауссовское распределение, проблема установления истинного значения принятого бита имеет следующую математическую формулировку. Фототок для битов 1 и 0 является выборкой гауссовской переменной со средним значением I1 и вариацией s1, а приёмник должен отслеживать этот сигнал и решать, является ли переданный бит 0 или 1. При этом существует много возможных правил принятия решения, которые могут быть реализованы в приёмнике с целью минимизации коэффициента битовых ошибок. Для значения фототока I этим оптимальным решением является наиболее вероятное значение переданного бита, которое определяется путём сравнения текущего значения фототока с пороговым значением Iп, используемым для принятия решения.

Пусть при  I ³ Iп принимается решение о том, что был передан бит 1, в противном случае – бит 0. Когда биты 1 и 0 равновероятны, что и рассматривается в дальнейшем, пороговый ток приблизительно равен:

https://studfiles.net/html/2706/752/html_5VPw5yrLaC.oMPo/img-GF4lJO.png(1.2)

Вероятность того, что I < Iп, т. е. вероятность ошибки при передаче бита 1, обозначим через Р0,1, а вероятность решения для переданного бита 1, когда I ³ Iп при переданном 0, обозначим Р1,0.

Пусть Q(х) обозначает вероятность того, что нулевая средняя вариация гауссовской переменной превышает значение х, тогда:

https://studfiles.net/html/2706/752/html_5VPw5yrLaC.oMPo/img-9RIwmH.png(1.3) https://studfiles.net/html/2706/752/html_5VPw5yrLaC.oMPo/img-aPnabl.png(1.4)https://studfiles.net/html/2706/752/html_5VPw5yrLaC.oMPo/img-x6_0us.png(1.5)

Можно показать легко увидеть, что BER определяется по формуле,

https://studfiles.net/html/2706/752/html_5VPw5yrLaC.oMPo/img-4_TeoR.png(1.6)

Очень важно отметить, что в ряде случаев эффективным является использование изменяемого в зависимости от уровня сигнала порога принятия решения, как, например, шума оптического усилителя. Многие высокоскоростные приёмники обладают такой особенностью. Однако более простые приемники имеют порог, соответствующий среднему уровню принимаемого тока, а именно (I1 + I0)/2. Такая настройка порогового   значения дает большой коэффициент битовых ошибок, определяемый выражением.

https://studfiles.net/html/2706/752/html_5VPw5yrLaC.oMPo/img-ScdEDR.png(1.7)

Выражение (1.6) можно использовать для оценки BER, когда известны как мощность полученного сигнала, соответствующего битам 0 и 1, так и статистика шумов.

Битовые ошибки являются основным источником ухудшения качества связи, проявляющегося в искажении речи в телефонных каналах, недостоверности передачи информации или снижении пропускной способности передачи данных, и характеризуются статистическими параметрами и нормами на них, которые определены соответствующей вероятностью выполнения этих норм. Последние делятся на долговременные и оперативные нормы, первые из которых определяются рекомендациями ITU-T G.821 и G.826, а вторые – М.2100, М.2110 и М.2120, при этом, согласно М.2100, качество цифрового тракта по критерию ошибок делят на три категории:

· нормальное – BER <10-6;

· пониженное – 10-6≤ BER <10-3 (предаварийное состояние);

· неприемлемое – BER ≥ 10-3 (аварийное состояние).

Так как появление ошибок является следствием совокупности всех текущих условий передачи цифровых сигналов, имеющих случайный характер, то при отсутствии данных о законе распределения ошибок его отдельные элементы могут быть определены с определенной степенью достоверности только по результатам продолжительных измерений. В то же время на практике необходимо, чтобы значения параметров ошибок для ввода в эксплуатацию и технического обслуживания систем передачи основывались на достаточно коротких интервалах времени измерения.

Для измерения коэффициента ошибок разработан ряд специальных BER анализаторов – измерителей коэффициента ошибок, включающих генераторы псевдослучайных и детерминированных последовательностей передаваемых кодированных символов, а также приемное оборудование, осуществляющее собственно измерение коэффициента ошибок. В случае посимвольного сравнения кодов измерение может быть выполнено с использованием шлейфа, т.е. путем измерения ошибок с одной оконечной станции при установке на противоположном конце шлейфа. Другой метод основан на выделении ошибок благодаря избыточности используемых кодов и используется для измерений от передающей до приемной сторон тракта или участка линии, т.е. когда выделение и фиксация ошибок производятся на ее приемном конце. Очевидно, что в первом случае требуется использование одного комплекта, а во втором – двух комплектов приборов. При этом измеренное значение коэффициента ошибок отражает качество передачи при прохождении сигнала в обоих направлениях и в каждом направлении соответственно.

**Помехоустойчивое** **кодирование** — **кодирование**, предназначенное для передачи данных по каналам с помехами, обеспечивающее исправление возможных ошибок передачи вследствие помех.

 Для обнаружения ошибок используют **коды обнаружения ошибок**, для исправления — **помехоустойчивые коды**.

Классификация помехоустойчивых кодов

В непрерывных кодах передаваемая информационная последовательность не разделяется на блоки. Избыточные элементы размещаются в определенном порядке между информационными.

Равномерные блочные коды делятся на разделимые и неразделимые. В разделимых кодах элементы информационной и проверочной частей кодовой комбинации всегда стоят на определенных местах. В неразделимых кодах деление на информационные и проверочные разряды отсутствует.

Разделимые коды, в свою очередь, делятся на систематические (линейные) и несистематические (нелинейные). Систематическими кодами называются блочные разделимые (n,k)-коды, в которых проверочные элементы представляют собой линейные комбинации информационных, несистематические коды таким свойством не обладают.

Помехоустойчивое кодирование предполагает введение в передаваемое сообщение, наряду с информационными, так называемых проверочных разрядов, формируемых в устройствах защиты от ошибок (кодерах-на передающем конце, декодерах — на приемном). Избыточность позволяет отличить разрешенную и запрещенную (искаженную за счет ошибок) комбинации при приеме, иначе одна разрешенная комбинация переходила бы в другую.

Помехоустойчивый код характеризуется тройкой чисел (n, k, d0), где n— общее число разрядов в передаваемом сообщении, включая проверочные (г), k=n-r - число информационных разрядов, d0— минимальное кодовое расстояние между разрешенными кодовыми комбинациями, определяемое как минимальное число различающихся бит в этих комбинациях. Иногда используются дополнительные показатели избыточности, производные от приведенных выше характеристик n, k:R = r/n- относительная избыточность, v = k / n - относительная скорость передачи.

Существующие помехоустойчивые коды можно разделить на ряд групп, только часть из которых используется для обнаружения ошибок в передаваемых по сети пакетах. В группе систематических (линейных) кодов общим свойством является то, что любая разрешенная комбинация может быть получена в результате линейных операций над линейно-независимыми векторами. Это способствует упрощению аппаратной и программной реализации данных кодов, повышает скорость выполнения необходимых операций.

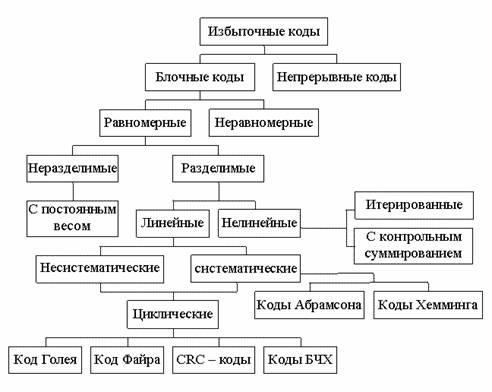
  


Рис. 1.2. Классификация помехоустойчивых кодов

Простейшими систематическими кодами являются биты четности/нечетности. Они не позволяют обнаружить ошибки четной кратности (т.е. ошибки одновременно в двух, четырех и т.д. битах) и поэтому используются при невысоких требованиях к верности принимаемых данных (или при малой вероятности ошибок в линии передачи). Примером может служить бит Parity (соответствие) в установках режимов работы последовательного порта с помощью команды MODE (MS DOS). Несмотря на ограниченные возможности обнаружения ошибок, биты четности/нечетности имеют большое значение в теории помехоустойчивого кодирования. Одни иг первых математически обоснованных и практически использовании? помехоустойчивых кодов - коды Хэмминга представляют собой простс совокупность перекрестных проверок на четность/нечетность. Циклические коды могут рассматриваться как обобщенные проверки на четность/ нечетность

Клод Шеннон сформулировал теорему для случая передачи дискретной информации по каналу связи с помехами, утверждающую, что вероятность ошибочного декодирования принимаемых сигналов может быть обеспечена сколь угодно малой путем выбора соответствующего способа кодирования сигналов. В теореме Шеннона не говорится о том, как нужно строить помехоустойчивые коды. Однако в ней указывается на принципиальную возможность кодирования, при котором может быть обеспечена сколь угодно высокая верность передачи. Это явилось стимулом к разработке помехоустойчивых кодов.

Помехоустойчивость кодирования обеспечивается за счет введения избыточности в кодовые комбинации, т.е. за счет того, что не все символы в кодовых комбинациях используются для передачи информации.

Все помехоустойчивые коды можно разделить на два основных класса: блочные и непрерывные (рекурентные или цепные).

В блочных кодах каждому сообщению (или элементу сообщения) сопоставляется кодовая комбинация (блок) из определенного количества разрядов. Блоки кодируются и декодируются отдельно друг от друга.

Блочные коды могут быть равномерными, когда длина кодовых комбинаций *п* постоянна, или неравномерными, когда *п* непостоянно.

В непрерывных кодах введение избыточности в последовательность входных символов осуществляется без разбивки ее на отдельные блоки. Процессы кодирования и декодирования в непрерывных кодах имеют также непрерывный характер.

Как блочные, так и непрерывные коды в зависимости от методов внесения избыточности подразделяются на разделимые и неразделимые. В разделимых кодах четко разграничена роль отдельных символов. Одни символы являются информационными, другие являются проверочными и служат для обнаружения и исправления ошибок. Разделимые блочные коды называются обычно *п,k-*кодами, где *п –* длина кодовых комбинаций, *k –*число информационных символов в комбинациях.

Неразделимые коды не имеют четкого разделения кодовой комбинации на информационные и проверочные символы.

Разделимые блочные коды делятся, в свою очередь, на несистематические и систематические. Несистематические разделимые коды строятся таким образом, что проверочные символы определяются как сумма подблоков длины *l,* на которые разделяется блок информационных символов.

Большинство известных разделимых кодов составляют систематические коды. У этих кодов значение проверочных символов определяется в результате проведения линейных операций над определенными информационными символами. Для случая двоичных кодов каждый проверочный символ выбирается таким, чтобы его сумма по модулю два с определенными информационными символами стала равной нулю (т.е. сумма единиц была четной). Декодирование сводится к проверке на четность определенных групп символов. В результате таких проверок дается информация о наличии ошибок, а в случае необходимости – о позиции символов, где имеются ошибки.

# Приложение, разработанное в результате преддипломной практики

## 3.1. Архитектура приложения.

Для написания программы использовались рекомендации по Code Convention указанные в стандарте PEP 8. Разрабатываемое приложение должно обеспечивать следующий функционал:

* иметь возможность симулировать передачу данных по каналу связи с помехами различных типов (одиночные, пакетные, смешанные);
* поддерживать различные режимы тестирования, а именно тестирования одиночноко кодера и каскада из двух кодеров;
* иметь возможность постановки в каскад различных кодеров;
* иметь систему логирования полученной информации;
* иметь два режима работы: графический и консольный;
* поддерживать использование перемежителей при передаче данных по каналу;
* обеспечивать визуализацию данных, полученных в результате тестирования;
* поддерживать сохранение полученных данных в базе данных.

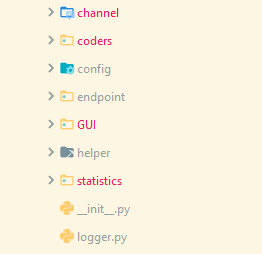
Из вышеописанных требований и была разработана архитектура приложения. На рисунке 3.1 ниже можно увидеть схему дискретного канала связи.

  
  
Рисунок 1.1. Блок схема дискретной системы связи

На базе дискретного канала связи и была построена программная модель дискретного канала связи. Были разработаны модули для следующей функциоональности:

* обработки ошибок при работе приложения;
* модуль для общих преобразований частоиспользуемых типов данных;
* модуль для обработки конфигурационного файла приложения;
* модуль для поддержки различных режимов работы приложения;
* модуль для имитации канала связи и управления им в ходе тестирования (в него входит и генерация ошибок);
* модуль для имитации кодека и управления им, а так же подсчёта результатов работы входящих в него кодеров;
* модуль для симуляции работы кодеров;
* модуль для управления событиями приложения;
* модуль для управлением отдельным потоком тестирования;
* модуль работы с графическим интерфейсом;
* модуль для сбора статистики и записи её в базу данных;
* модуль для визулизации полученных результатов.

В каждый из модулей входит от одного до нескольких управляющих классов и (или) функций. На рисунке 3.1 представлены основные пакеты приложения.



3.1. Корневые пакеты приложения с исходным кодом

При работе с графическим интерфейсом используется паттерн MVC (Model, View, Controller). MVC — схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо.

Модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя своё состояние. В контексте этого приложения моделью являются кодеры и база дынных. Она будет описана подробнее позже.

Представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели. В данном приложении — это Qt5 widgets и классы, находящиеся в пакете *windows*. Ниже на рисунке 3.2 представлен листинг кода класса управления окном приложения.

class MainWindow(QMainWindow):  
 def \_\_init\_\_(self, controller):  
 log.debug(**"Create main window"**)  
  
 super(MainWindow, self).\_\_init\_\_()  
 self.controller = controller  
 uic.loadUi(**r'.\src\GUI\UI\window.ui'**, self)  
  
 setup\_main\_window\_coder(controller, self)  
 setup\_main\_window\_second\_coder(controller, self)  
  
 self.countTestEdit.valueChanged.connect(self.controller.testParams.set\_count\_test)  
 self.countTestEdit.valueChanged.emit(self.countTestEdit.value())  
  
 self.informationEdit.textChanged.connect(self.controller.testParams.set\_test\_info)  
 self.informationEdit.textChanged.emit(self.informationEdit.text())  
  
 globalSignals.stepFinished.connect(self.singleProgress.setValue)  
 globalSignals.autoStepFinished.connect(self.autoProgress.setValue)  
  
 self.startSingleFirstCoderButton.clicked.connect(self.controller.testParams.start\_first\_single\_test)  
 self.startTestsFirstCoderButton.clicked.connect(self.controller.testParams.start\_first\_test\_cycle)  
 self.startSingleCascadeCoderButton.clicked.connect(self.controller.testParams.start\_cascade\_single\_test)  
 self.startTestsCascadeCoderButton.clicked.connect(self.controller.testParams.start\_cascade\_test\_cycle)  
  
 setup\_main\_window\_noise(controller=controller, window=self)  
 self.show()

Рисунок 3.2. Класс отрисовки главного окна приложения

Класс MainWindow наследуется от класса из модуля PyQt5 QMainWindow. Можно увидеть из рисунка выше следующее: класс отрисовки загружает предварительно приготовленной файл с разметкой и маппит компоненты в класс окна, а затем подписывает определённые методы контроллера на события изменения компонентов окна.

Контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений. Контроллеры отвечают за корректность переданных данным с пользовательского интерфейса, а так же за создания нового потока тестирования. Дальше на рисунке 3.3. показана диаграмма класса контролера отвечающего за создание кодера из полученных данных от пользователя. Как можно увидеть класс наследуется от класса GeneralCoderSimulate, который реализует общую логику

В приложении реализована небольшая система управления событиями и подписка на них. Её реализует класс \_\_GlobalSignals с помощью событийной системы модуля PyQt5. В приложении имеются пять глобальных событий, которые вызываются во время выполнения тестирования: начало тестирования, завершения одного теста, завершение одного цикла тестов, завершение тестирования, произошедшая ошибка в различных потоках.

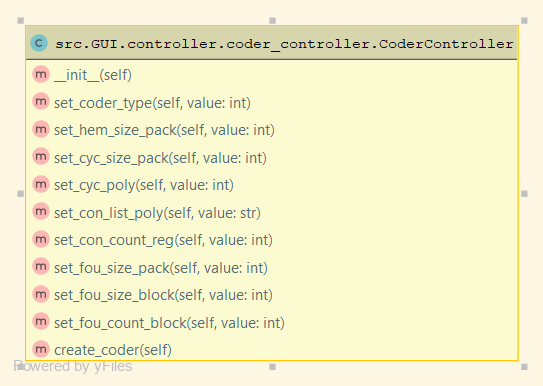


Рисунок 3.3. Класс управлением создания кодера

Этот класс содержит кроме всего прочего, фабричный метод, который создаёт конкретную инстанцию класса кодера и возвращает его в виде абстрактного класса кодера, от которого наследуются все классы кодеров.

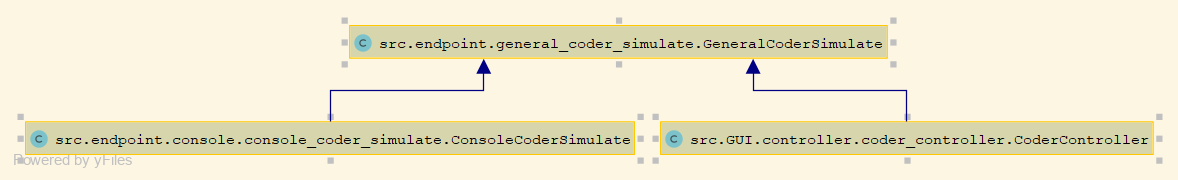


Рисунок 3.4. Схема наследования классов управления создания кодеров

На рисунке 3.4. видна схема наследования в классе управления создания кодеров, класс GeneralCoderSimulate является абстрактным и не должен создавать инстанции классов. Класс ConsoleCoderSimulate используется для аналогичного функционала, что и CoderController при создании кодеров при консольном режиме работы приложения.

Так же присутствует класс контроллер для управления процесса создания канала связи и начала тестирования TestController. Далее на рисунке 3.5 можно увидеть его

# Заключение

В процессе прохождения преддипломной практики было разработано приложение для анализа эффективности работы помехоустойчивых кодеков с определёнными набором кодеков:

* 1. FPM
  2. FBI
  3. BOPF

В работе использовались различные аппаратно-программные конфигурации. Использовались операционные системы: Windows 8, Windows 10, MacOS.

В процессе прохождения производственной практики в отделе корпоративных решений компании ООО "LeverX" все поставленные цели и задачи, связанные с ознакомлением методов обработки информации и их особенностями, были выполнены.

Также был сделан вывод, что решение о выборе варианта построения аналитической системы и методики обработки информации зависит от задач и особенностей каждой конкретной компании.

Полученные в ходе проделанной работы знания считаю полезными и актуальными.

# Список использованных источников информации