Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий
Кафедра прикладной математики

Отчет защищен с оценкой _____

Преподаватель_____(подпись)
«___»_____2023 г.

Отчет
по расчетной работе
по дисциплине «Объектно-ориентированное программирование»

Студент гр. ПИ-02
Чередов Р.А.

Ассистент кафедры ПМ,
Рахманин Д.С.

Барнаул 2023

Оглавление

1. Задание

Суммарный отчет по лабораторным работам 2, 3 и 4 представляется в форме расчетного задания по анализу и проектированию каркаса системы.

Отчет:

Описание предметной области:

- Общая характеристика решаемых задач в предметной области;
- Характеристика поставленной задачи;
- Действующие объекты и функционал;
- Возможные расширения системы.

Объекты, интерфейсы и классы проектируемой системы:

- Перечень классов и объектов;
- Назначение классов;
- Основные методы классов;
- Отношения между классами.

Пример реализации подсистемы на основе принципа делегирования:

- Диаграмма классов;
- Назначение классов;
- Логи работы программы.

2. Описание предметной области:

2.1. Общая информация

В настоящее время более чем когда-либо актуальной проблемой является вопрос повышения уровня сохранности человеческих жизней за счет оптимизации расположения сотовых вышек и обработки данных сотовой связи. Система сети сотовой связи – это емкостное понятие в состав которого входит большое количество подсистема и мероприятий.

В основном системы сети сотовой связи делятся на подсистемы:

- РТ (радиотелефон сети сотовой связи)
- БС (базовая станция)
- ЛС (линия связи)
- ЦКС (центр коммутации связи)

Система ЦКС состоит:

- Сервер
- Терминалы
- Оборудование передачи данных (мосты, маршрутизаторы и т.д.)

Система ЛС:

- Коаксиальные провода

Система БС:

- Антенно-фидерное устройство
- Модули приемопередатчиков
- Контролер для каждого модуля приемопередатчика
- Базовый контроллер
- Автономный ретранслятор
- Соединительная линия с сетью общего пользования
- Контроллеры
- Терминал «менеджер системы»
- Пульт диспетчера

Система РТ:

- Стандартный телефон пользования

2.2. Перечень задач

Основными задачами перед разрабатываемым ПО являются:

- Нужно реализовать управление всеми элементами системы сотовой связи.
- Нужно реализовать возможность наблюдения за текущими статусами элементов сотовой связи
- Нужно реализовать возможность получения данных о местонахождения абонента сотовой связи
- Нужно реализовать возможность получения о технических неисправностях сети сотовой связи
- Хранения объема данных о выходов абонента в сеть за последние полгода, а именно ближайшее расположение с вышкой

2.3 Анализ обрабатываемых данных

Каждая подсистема будет работать со своим набором данных, поступающим из входного потока.

- Конфигурационный файл для системы комплексной обработки данных включает: дата и время, время прохода и предположительное расстояние абонента до вышек.
- Входные данные с пульта диспетчера состоящие из: запроса с пульта и ФИО пользователя.
- Входные данные с коаксиальных кабелей, то есть раз в час происходит проверка кабелей путем PING, если доходит до конца и не происходит разрыв, возвращается PING с сообщением enable, иначе False с примерным местом пробоя.
- Данные в базе данных о последнем выходе в сеть абонентов, каждая сотовая вышка сохраняет эти данные и только одна из них помечается как ближайшая вышка последнего выхода абонента в сеть, если эта вышка находилась в +-100м от абонента, то такие данные помечаются флагом, а также в эти данные входит направление, для определения сектора сот. Данные хранятся полгода.

2.4 Краткая характеристика применяемых алгоритмов

Быстрый поиск в сети сотовой связи:
По запросу в базу данных, находится последняя ближайшая сотовая вышка. У каждой сотовой вышки есть её координаты. Если последний выход абонента в сеть помечен флагом, то начинается поиск человека в +-100м от вышки.
Полный поиск в сети сотовой связи:
Для обнаружения местоположения абонента сотовой сети, подается запрос в базу данных, найти данного абонента, по последнему зарегистрированному выходу в сеть.
После этого производится нахождения ближайшей сотовой вышки к последнему выходу абонента.
Каждая вышка имеет 3 подключенных вышки, формируется треугольник с нужной нам вышкой в центре.
В данных о последнем выходе абонента в сеть имеется направление, что дает нам понять в каком секторе из сотовых вышек начать поиски.
После определения сектора, подается запрос на данный сектор сотовых вышек, с получением данных о данном пользователе.
Представить данные можно в виде правильного шестиугольника.
Проведем ближайшее расстояние до вышек из вершин и получаем точное местоположение с последним выходом абонента в сеть. Погрешность 25 метров.

3. Объекты, интерфейсы и классы проектируемой системы

3.1. Перечень классов и объектов и их краткое описание

Модель состоит из нескольких объектов:

- Телефон
- Базовая станция
- Линяя связи
- Коммутационный центр
- Система контроля центра
- Система безопасности

Можно выделить следующие классы:

- Класс pager
    - Поля
        - Id
        - Три вышки
        - Флаг на дальность
    - Методы
        - Связаться с ближайшей вышкой
        - Связаться с тремя другими вышками

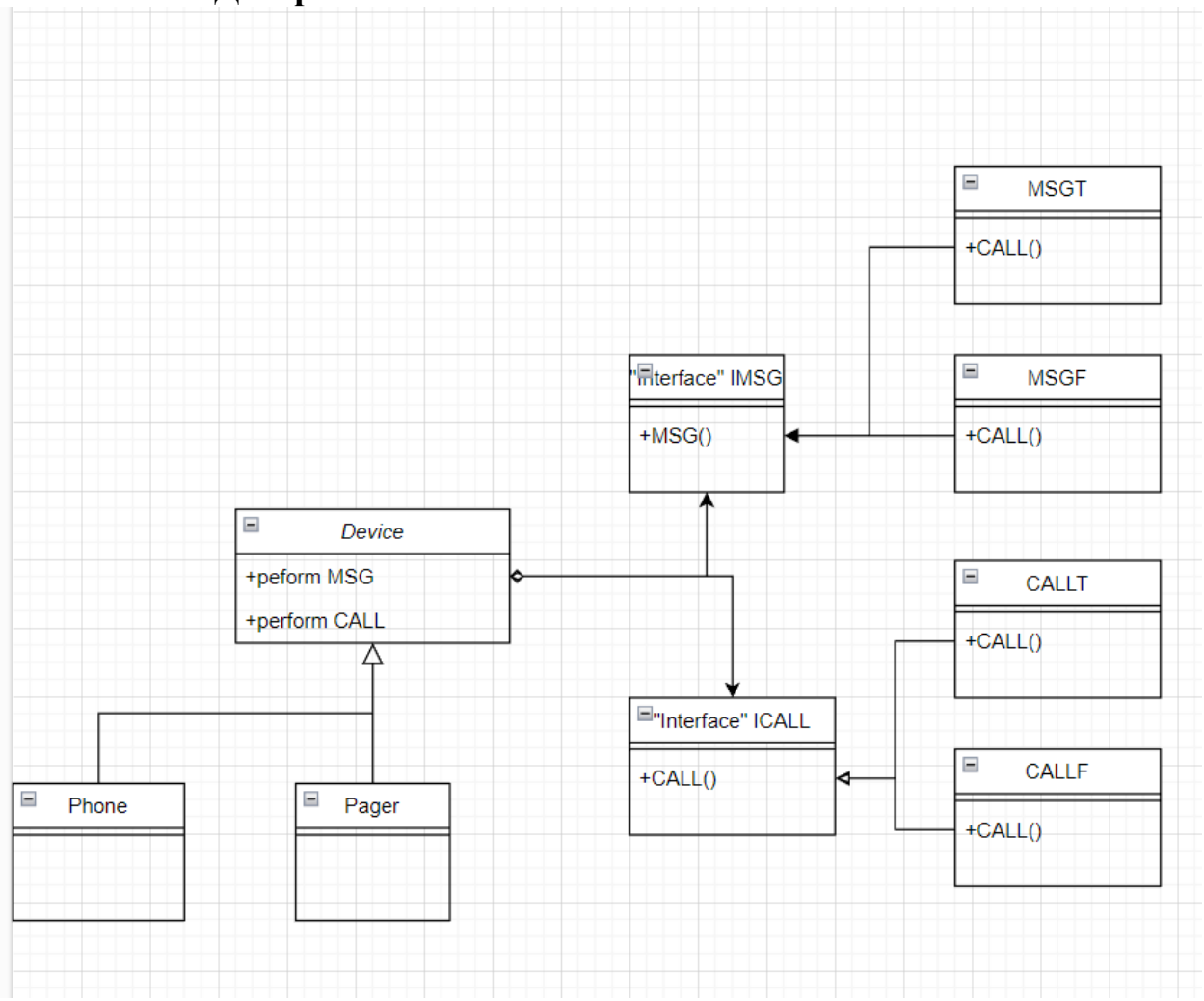
- Класс phone
  - Поля
    - Id
    - Три вышки
    - Флаг на дальность
  - Методы
    - Связаться с ближайшей вышкой
    - Связаться с тремя другими вышками
- Класс base station
  - Поля
    - Id
    - Абоненты в сети
  - Методы
    - Связь с телефоном
    - Проверка на дальность
    - Сформировать пакет данных
    - Передать данные по линии связи
- Класс commutation_line
  - Поля
    - Id
    - Пакет данных
  - Методы
    - Передача пакета данных в центр связи
- Класс commutation_centre
  - Поля
    - Пакет данных с линии связи
    - Пакет данных для решения
    - Сигнал
  - Методы
    - Распаковка пакета данных
    - Передача пакета данных в систему безопасности
    - Обработка пакета данных для решения
    - Передача данных для решения
    - Реагирование на Систему контроля
    - Передача сигналу от системы контроля к системе безопасности
- Класс security_system
  - Поля

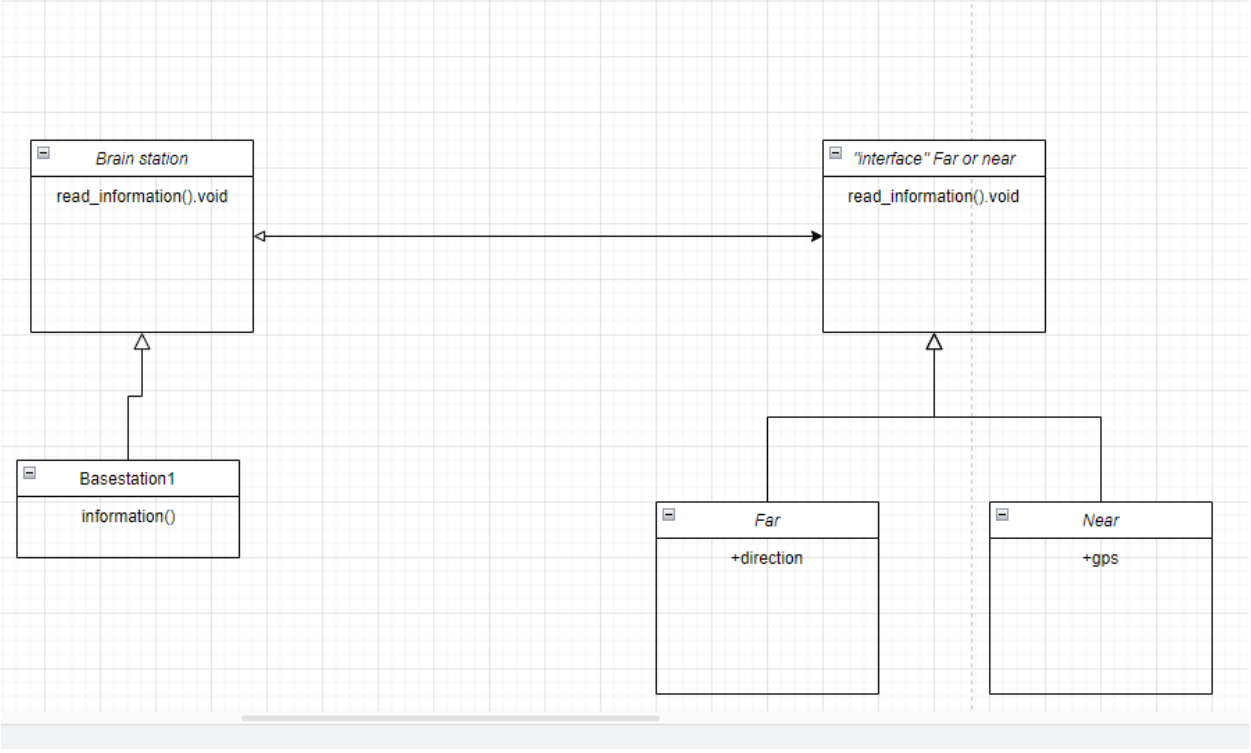
        - Телефон
        - Занятая вышка
        - Флаг 100 метров
        - Координаты вышек
    - Методы
        - Формирования пакета данных по запросу
        - Передача пакета данных

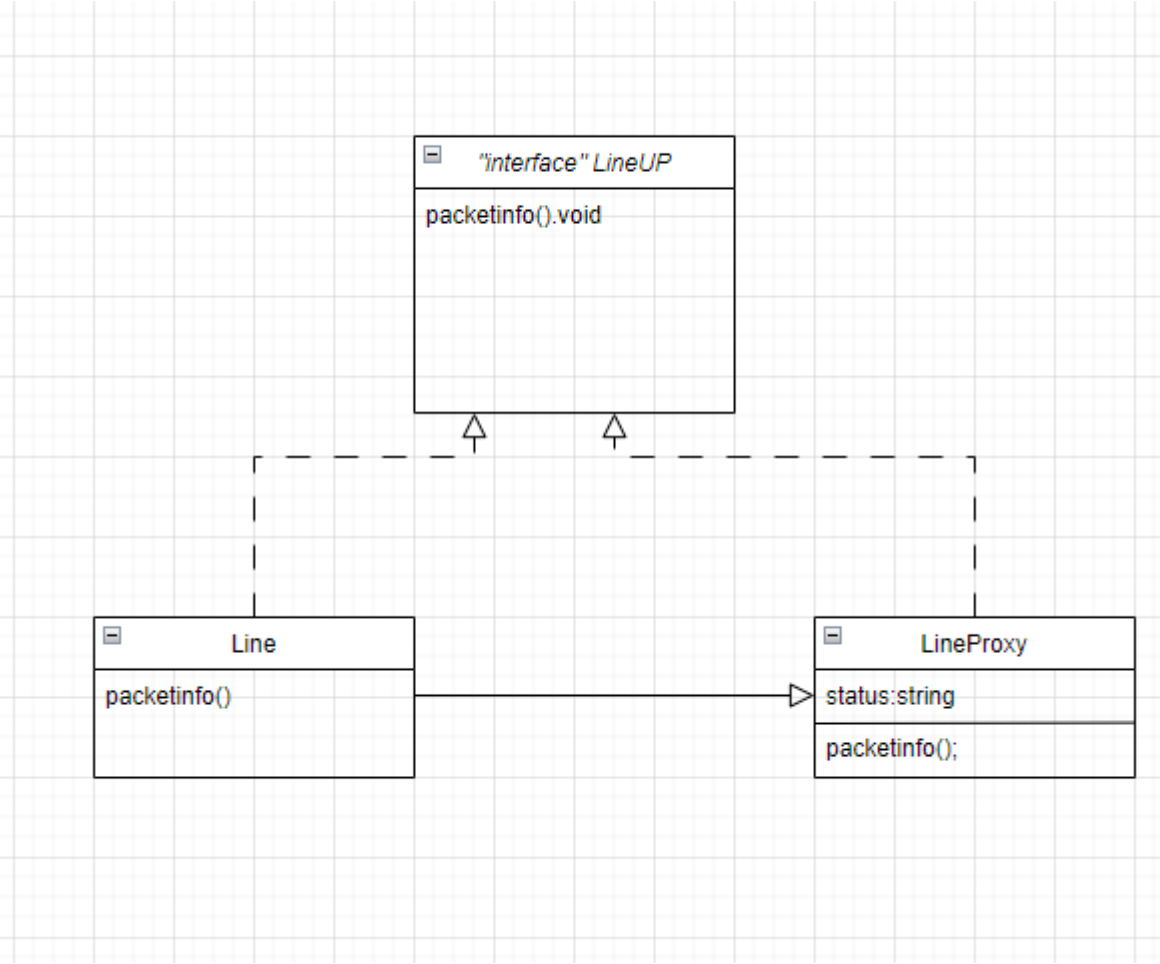
- Класс Complex_control_system
    - Поля
        - Сигнал
    - Методы
        - Формирование сигнала из класса event
        - Передача сигнала в коммутационный цетр
- Класс Event
    - Поля
        - Выбор эвента
    - Методы
        - Передача эвента в систему контроля
- Класс result
    - Поля
        - Абонент
        - Местоположение
    - Методы
        - Получение данных

4. Пример реализации подсистемы на основе принципа делегирования

4.1. Диаграмма классов






"interface" LineUP
packetinfo().void
Line
packetinfo()
LineProxy
status:string
packetinfo();

4.2. Описание классов

Интерфейс IMSG и ICALL реализуют функцию звонка и смс для класса Device, в который наследуют Phone и Pager

Интерфейс FarorNear. Наследующие его класс Far и near отвечающие за внутренности пакета данных. BrainStation- мозг базовой станции.

Line – класс описывающий основную функцию работы линии передачи. Прокси этого класса LineProxy проверяет статус линии связи.

4.3. Текст программы

Статическое делегирование

```
using namespace std;
class IMSG {
public:
	virtual  void   msg() = 0; // интерфейс не имеет реализации
};
class ICALL {
public:
	virtual  void   call() = 0; // интерфейс не имеет реализации

};
class  MSGT : public IMSG {
	// класс поведения для устройств, которые умеют писать смс
public:
	void msg() {
		printf("I am message!\n");
	}
};
class  MSGF : public IMSG {
	// класс поведения для устройств, которые не умеют писать смс
public:
	void msg() {
		printf("I am dont message!\n");
	}
};
class  CALLT : public ICALL {
	// класс поведения для устройств, которые НЕ умеют call
public:
	void call() {
		printf("I can call...\n");
	}
};
class  CALLF : public ICALL {
	// класс поведения для устройств, которые НЕ умеют call
public:
	void call() {
		printf("I can not call...\n");
	}
};
class Device {        // абстрактный класс устройства
public:
	IMSG* msgaction;
	ICALL* callaction;
	Device() {}
	~Device();
	// делегируем выполнение операции классам поведения :
	void performcall() { callaction->call(); }
	void performmsg() { msgaction->msg(); }

};
class Phone : public  Device {
public:
	Phone() {
```

```
                callaction = new CALLT();
                msgaction = new MSGT();

        }
};
class Pager : public  Device {
public:
        Pager() {
                callaction = new CALLF();
                msgaction = new MSGT();

        }
};
int main() {   // создаем объекты устройств
        printf("  devices\n");
        Phone Phone1;
        Pager Pager2;
        Phone1.performcall();
        Phone1.performmsg();
        Pager2.performcall();
        Pager2.performmsg();
}
```

Динамическое делегирование

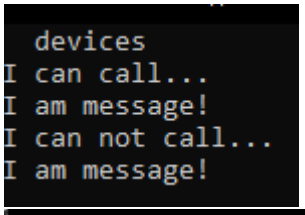
```
#include <iostream>
#include <string>
using namespace std;
class farornear
{
public:
        virtual string readinformation() = 0;
};
class BrainStation
{
public:
        void ChekInfo(farornear* rc)
        {
                info = rc;
        }
        string readinformation()
        {
                if (info)
                {
                        return info->readinformation();
                }
                return "Не установлен ";
        }
private:
        farornear* info = nullptr;;
};

class Far : public farornear
{
public:
        string readinformation() override
        {
                return "Направление";
        }
};
class Near : public farornear
{
public:
        string readinformation() override
        {
                // create pdf report logic
                return "Gps";
        }
};
```

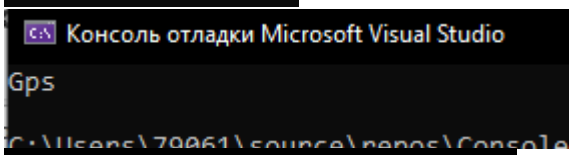
```
int main()
{
	setlocale(LC_ALL, "Russian");
	Far Far;
	Near Near;
	BrainStation BrainStation;
	BrainStation.ChekInfo(&Far);
	BrainStation.ChekInfo(&Near);
	cout << BrainStation.readinformation() << endl;


	return 0;
}
```

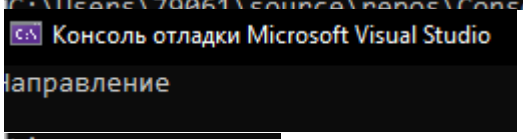
Прокси

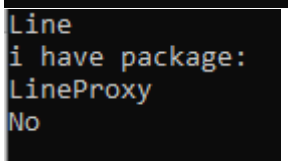
```
class LineUP {

    // класс, для которого создадим Proxy

public:
    virtual void packetinfo() = 0;

};
class Line : public LineUP {
    //  настоящий класс для обработки данных
public:
    int a;

    virtual void packetinfo() {
        if (a == 1) {
            cout << "i dont have package\n";
        }
        else {
            cout << "i have package: " <<  endl;
        }
    }
    Line(int inA) { a = inA;}
};
class LineProxy : public LineUP {
private:
    Line* prox;
    void status() { cout << "im ready" << endl; }

public:

    virtual void packetinfo() { cout << "No " << endl; }

    LineProxy(int inA ) {
        prox = new Line(inA);
        // здесь Proxy создает реальный объект M1
    }
    ~LineProxy() { delete prox; }
};
int main() {
    LineUP* t = new Line(0);
    LineUP* p = new LineProxy(1);
    cout << "Line\n";
    t->packetinfo();
    cout << "\LineProxy\n";
    p->packetinfo();
    delete p;
    delete t;
    return 0;}
```

4.4. Тесты программы









5. Список литературы

1. Тихвинский В.О., Терентьев С.В., Юрчук А.Б. Сети мобильной связи LTE. Технологии и архитектура. – М: Эко-Трендз, 2010.– 284 с.
2. Степутин, А. Н., Николаев А.Д. Мобильная связь на пути к 6G. В 2 томах. Том2 / А.Н. Степутин, А.Д. Николаев. –3-е изд. – Москва-Вологда: Инфра-Инженерия, 2020. – 420 с. : ил.
3. Рыжков А.Е., Сиверс М.А., Воробьев В.О., Гусаров А.С., Слышков А.С., Шуньков Р.В. Системы и сети радиодоступа 4G: LTE, WiMax. – СПб: Линк, 2012. – 226 с.