ПРОЕКТ З ПРЕДМЕТУ СИСТЕМНА ІНЖЕНЕРІЯ НА ТЕМУ «ADAS»

Підготувала студентка 2-го курсу ФТІ
НТУУ «КПІ ім. Ігоря Сікорського»
Групи ФБ-95
Петренко Марина

ЗМІСТ

Лабораторна робота 1

Система – ADAS

ADAS включає в себе такі підсистеми: підсистема сповіщення водія, підсистема попередження зіткнень, підсистема попередження виїзду за смугу руху, підсистема автоматичного паркування, адаптивний круїз контроль, підсистема контролю стану водія, підсистема екстреного виклику.

Підсистема сповіщення водія виконує такі функції:

- Приймає сигнал від інших підсистем на виводить відповідні повідомлення на дисплей, включає звуковий сигнал, вібрацію, тощо.

Підсистема попередження зіткнень виконує такі функції:

- Повідомляє про можливе зіткнення з іншим авто або об’єктом при швидкому наближенні до нього.
- Вмикає гальмівну підсистему автомобіля при наближенні до транспорту, що їде попереду.

Підсистема попередження виїзду за смугу руху виконує такі функції:

- Відслідковує розмітку на дорозі і сигналізує у разі виїзду транспортного засобу за її межі.
- Виконує послідовність операцій для того, щоб утримати транспортний засіб на своїй смузі руху.

Підсистема автоматичного паркування виконує такі функції:

- Автоматичне паркування автомобілю без участі водія або контрольоване водієм за допомогою дисплею в салоні або додатку на смартфоні.

Адаптивний круїз-контроль виконує такі функції:

- Підтримує встановлену швидкість.
- Аналізує швидкість руху автомобіля і зіставляє її з дозволеною швидкістю на даній ділянці дороги. Сповіщає у разі невідповідності швидкості і адаптує її до допустимої.
- Вмикає гальмівну підсистему автомобіля при наближенні до транспорту, що їде попереду.

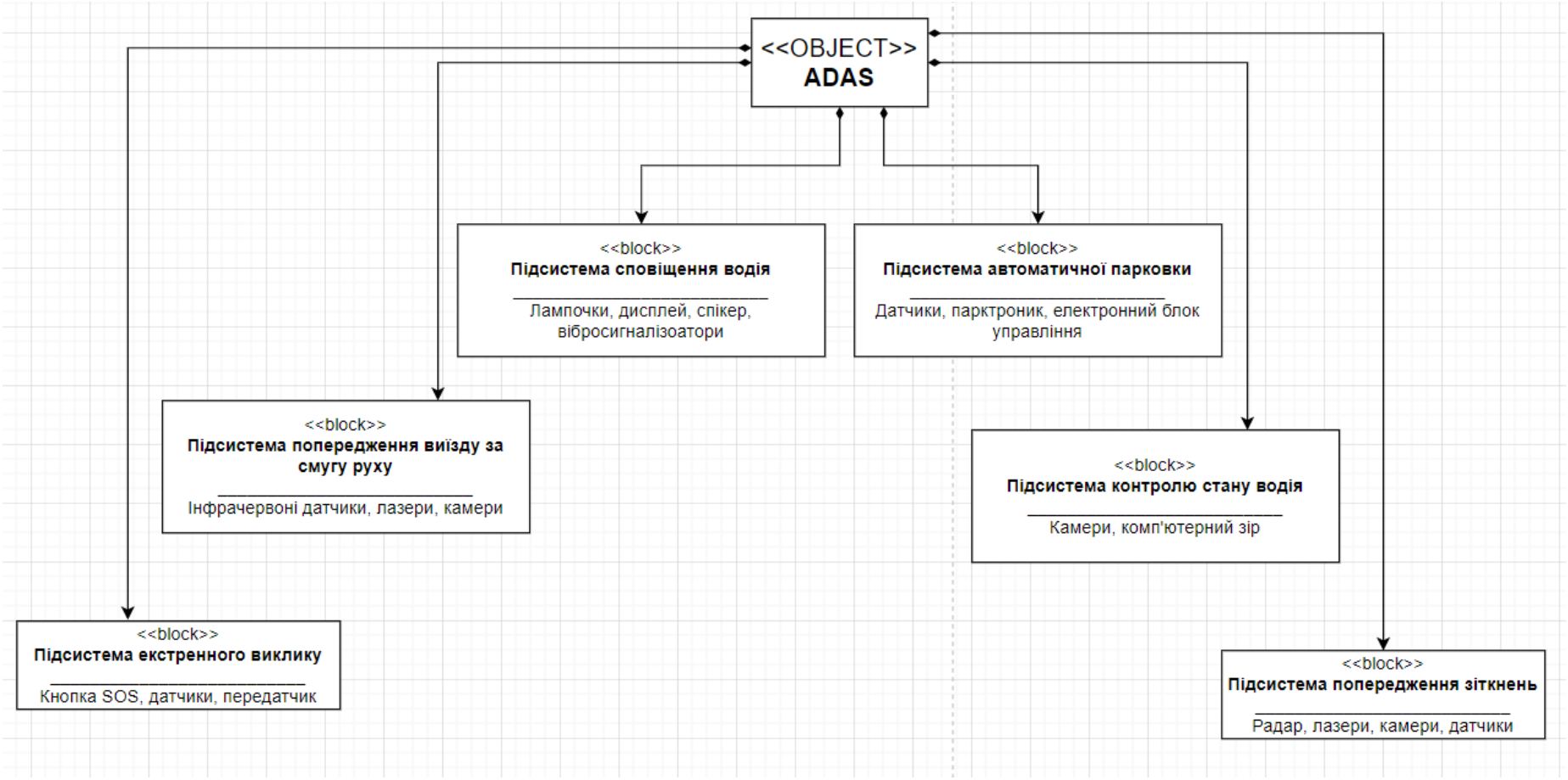
Підсистема контролю стану водія виконує такі функції:

- Контроль стану водія і сигналізація про наявні проблеми у разі їх наявності.

Підсистема екстреного виклику виконує такі функції:

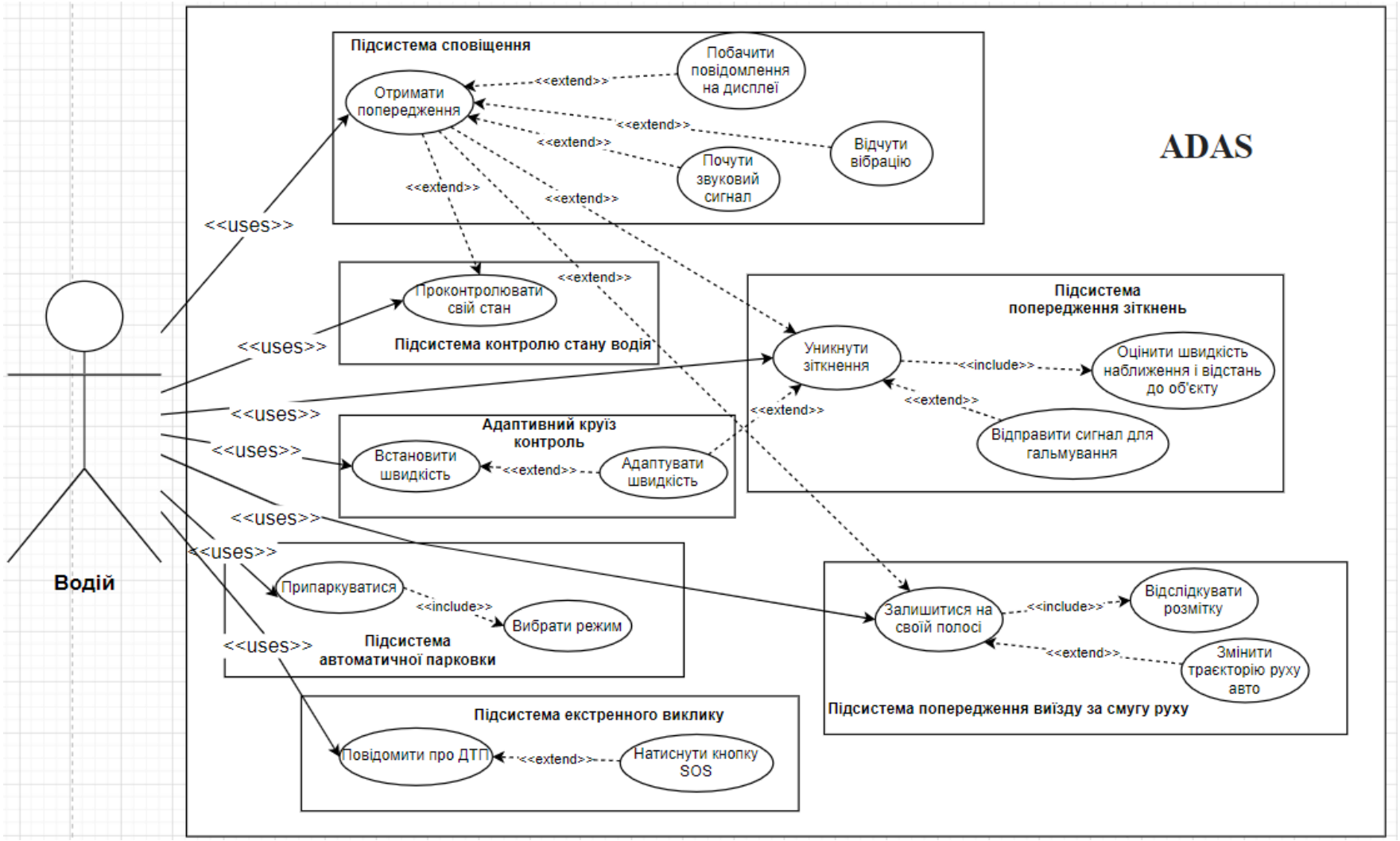
- Передає сигнал в екстрену службу через телефонне повідомлення або інтернет, при різких поворотах, гальмуванні, ударах, сприймаючи їх як ознаки ДТП

STRUCTURE DIAGRAM

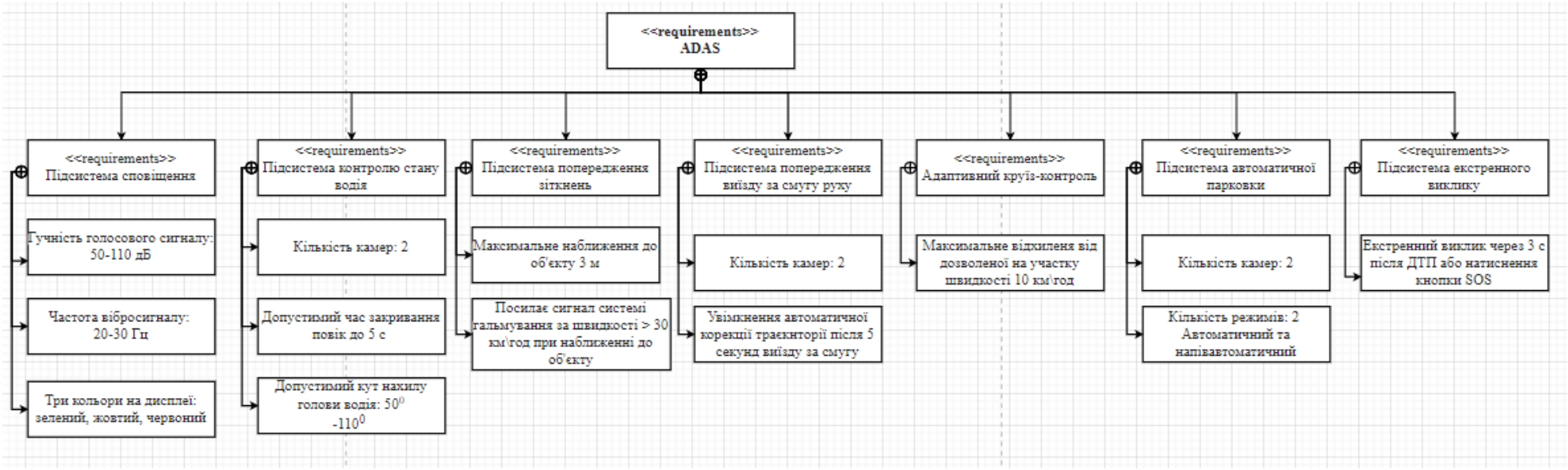


Лабораторна робота 2

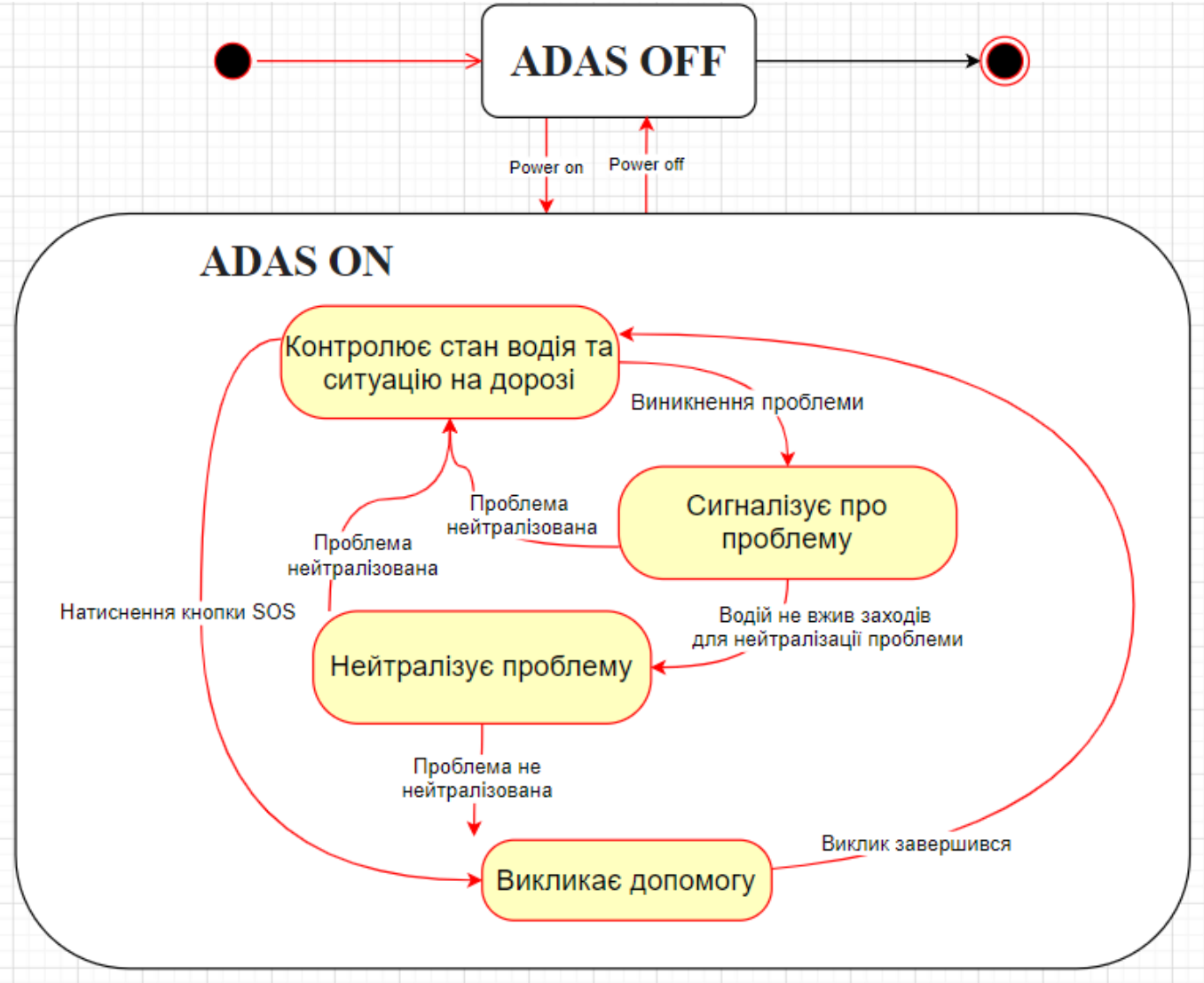
USE CASES

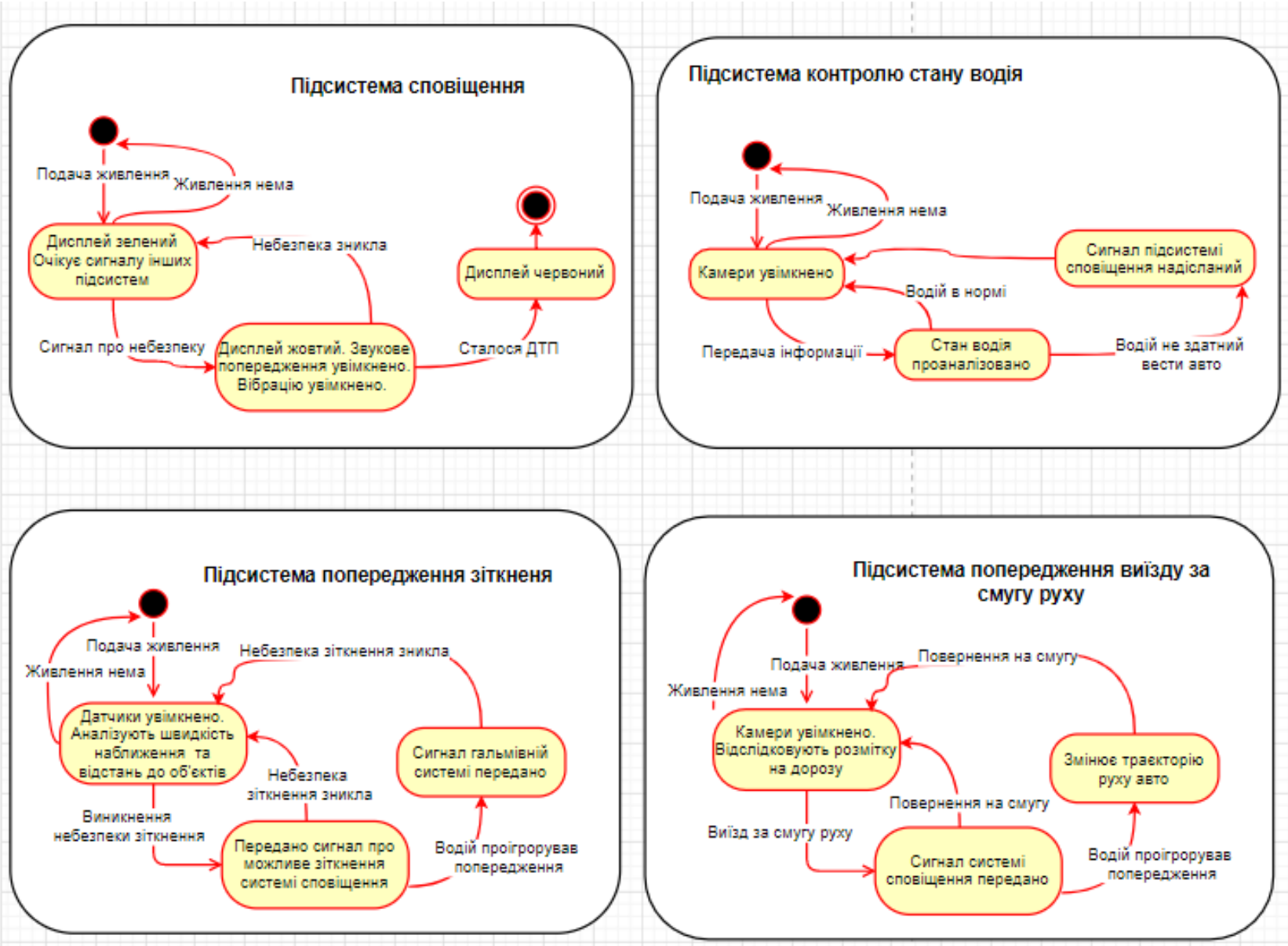


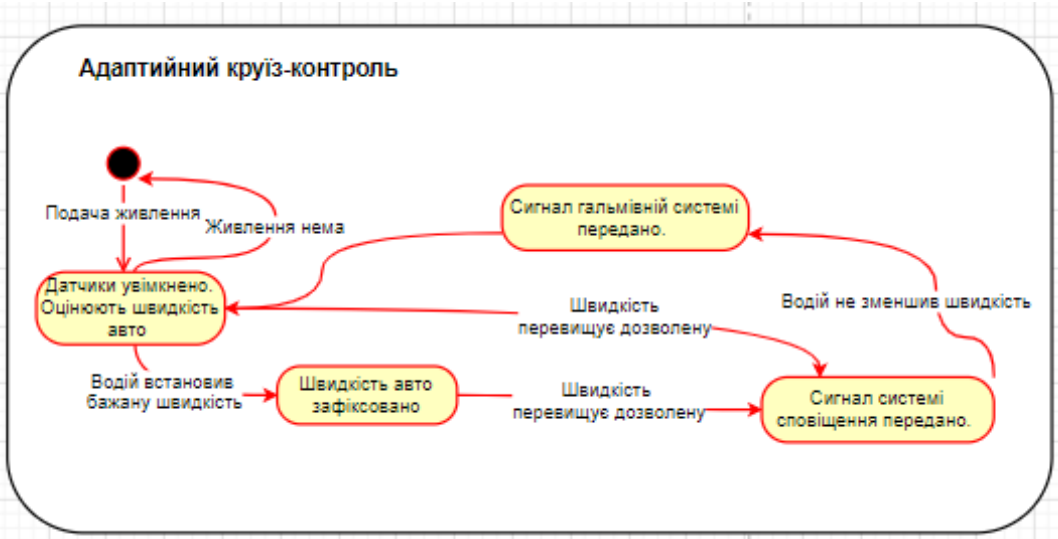
REQUIREMENTS DIAGRAM

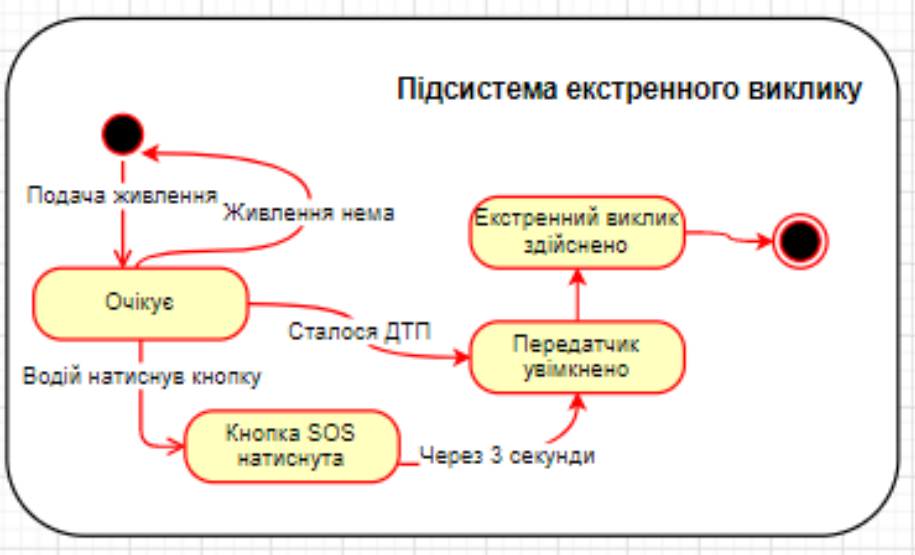


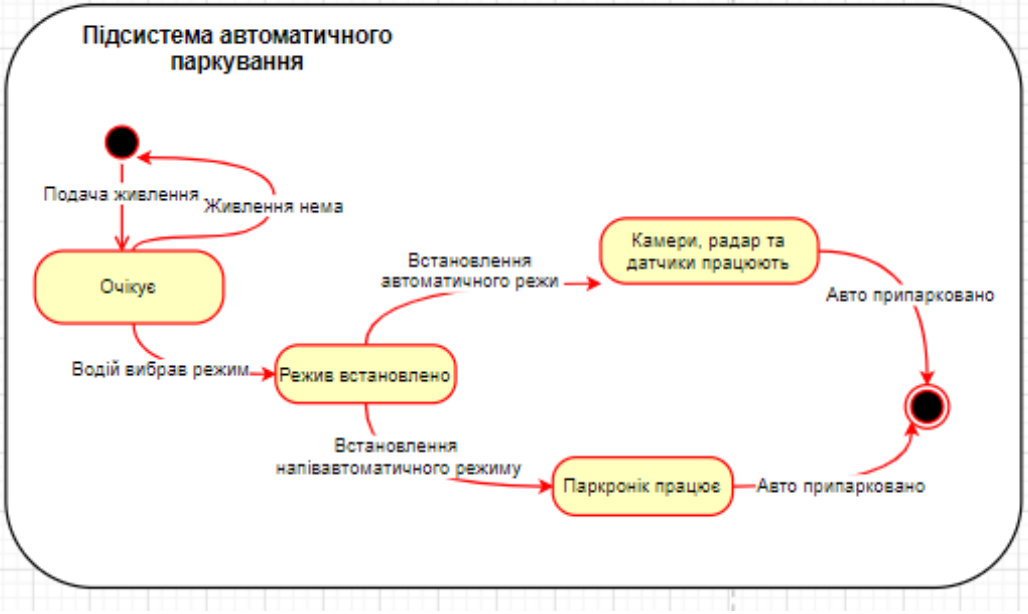
STATE MACHINE DIAGRAM



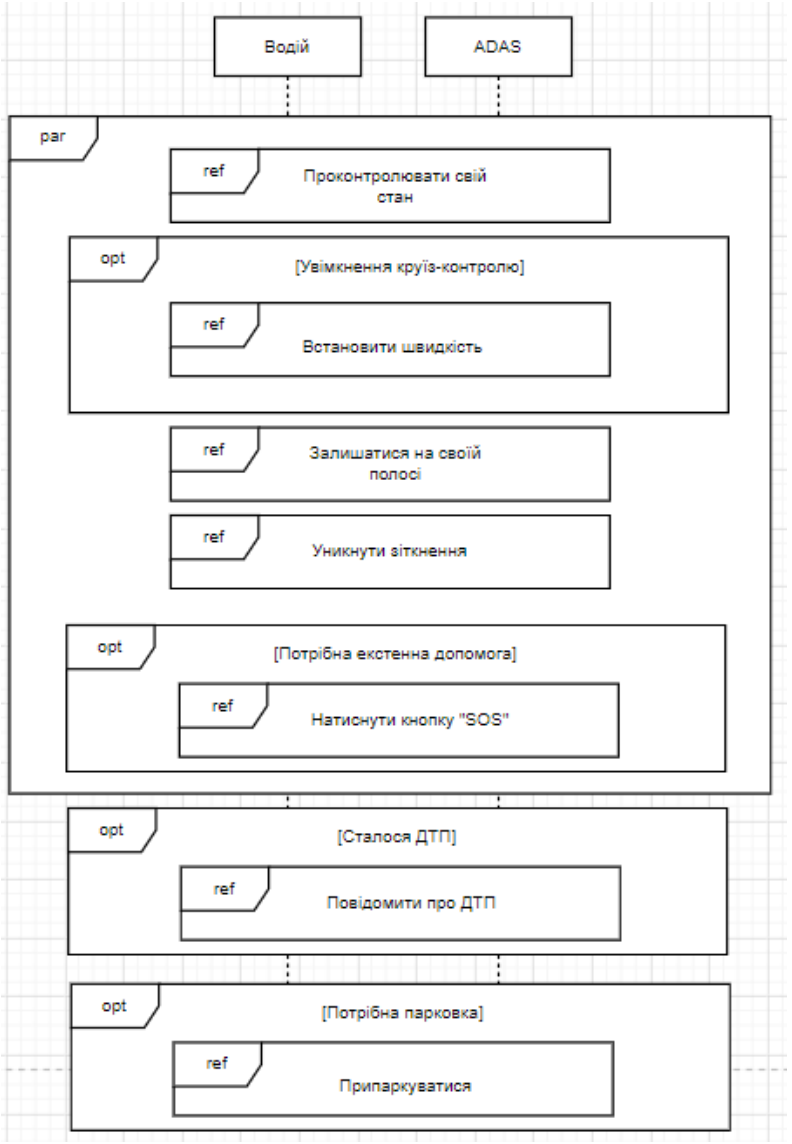

Підсистема сповіщення
Подача живлення
Живлення нема
Дисплей зелений
Очікує сигналу інших підсистем
Небезпека зникла
Дисплей червоний
Сигнал про небезпеку
Дисплей жовтий. Звукове попередження увімкнено. Вібрацію увімкнено.
Сталося ДТП
Підсистема контролю стану водія
Подача живлення
Живлення нема
Сигнал підсистемі сповіщення надісланий
Камери увімкнено
Водій в нормі
Передача інформації
Стан водія проаналізовано
Водій не здатний вести авто
Підсистема попередження зіткнення
Подача живлення
Небезпека зіткнення зникла
Живлення нема
Датчики увімкнено. Аналізують швидкість наближення та відстань до об'єктів
Небезпека зіткнення зникла
Сигнал гальмівній системі передано
Виникнення небезпеки зіткнення
Передано сигнал про можливе зіткнення системі сповіщення
Водій проігрорував попередження
Підсистема попередження виїзду за смугу руху
Подача живлення
Повернення на смугу
Живлення нема
Камери увімкнено. Відслідковують розмітку на дорозу
Змінює траєкторію руху авто
Повернення на смугу
Виїзд за смугу руху
Сигнал системі сповіщення передано
Водій проігрорував попередження

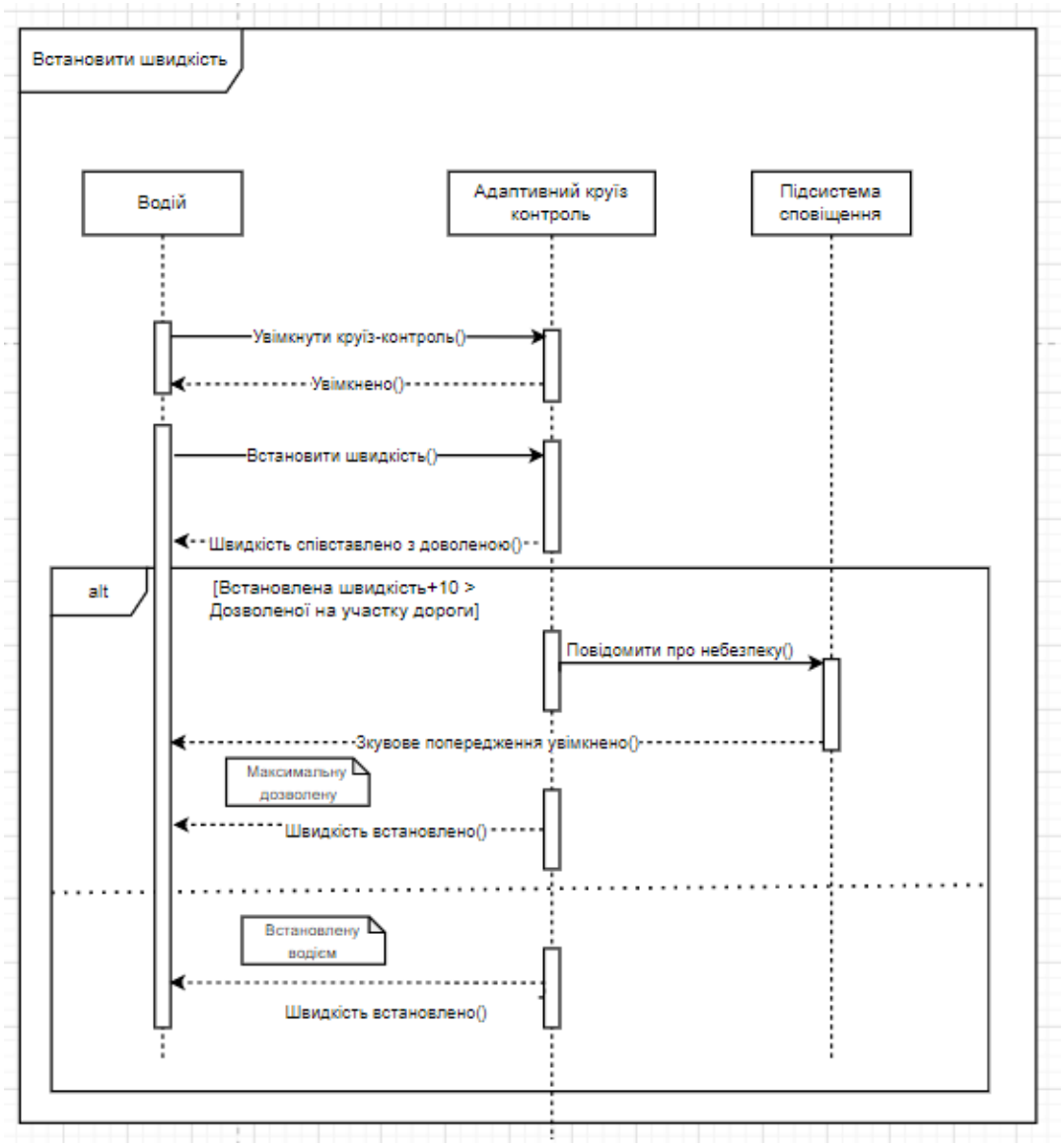

Адаптивний круїз-контроль
Подача живлення
Живлення нема
Датчики увімкнено. Оцінюють швидкість авто
Сигнал гальмівній системі передано.
Швидкість перевищує дозволену
Водій не зменшив швидкість
Водій встановив бажану швидкість
Швидкість авто зафіксовано
Швидкість перевищує дозволену
Сигнал системі сповіщення передано.

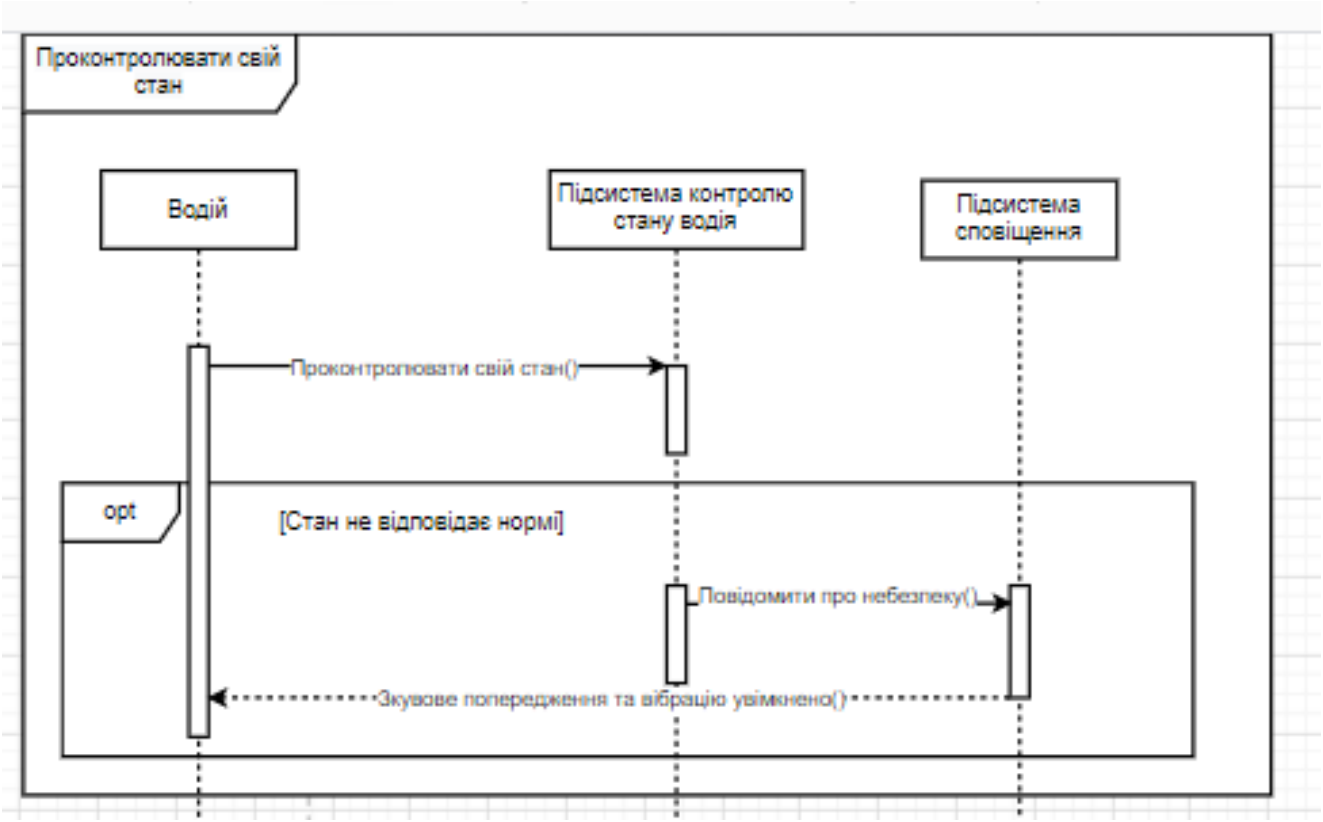

Підсистема екстренного виклику
Подача живлення
Живлення нема
Очікує
Екстренний виклик здійснено
Сталося ДТП
Передатчик увімкнено
Водій натиснув кнопку
Кнопка SOS натиснута
Через 3 секунди

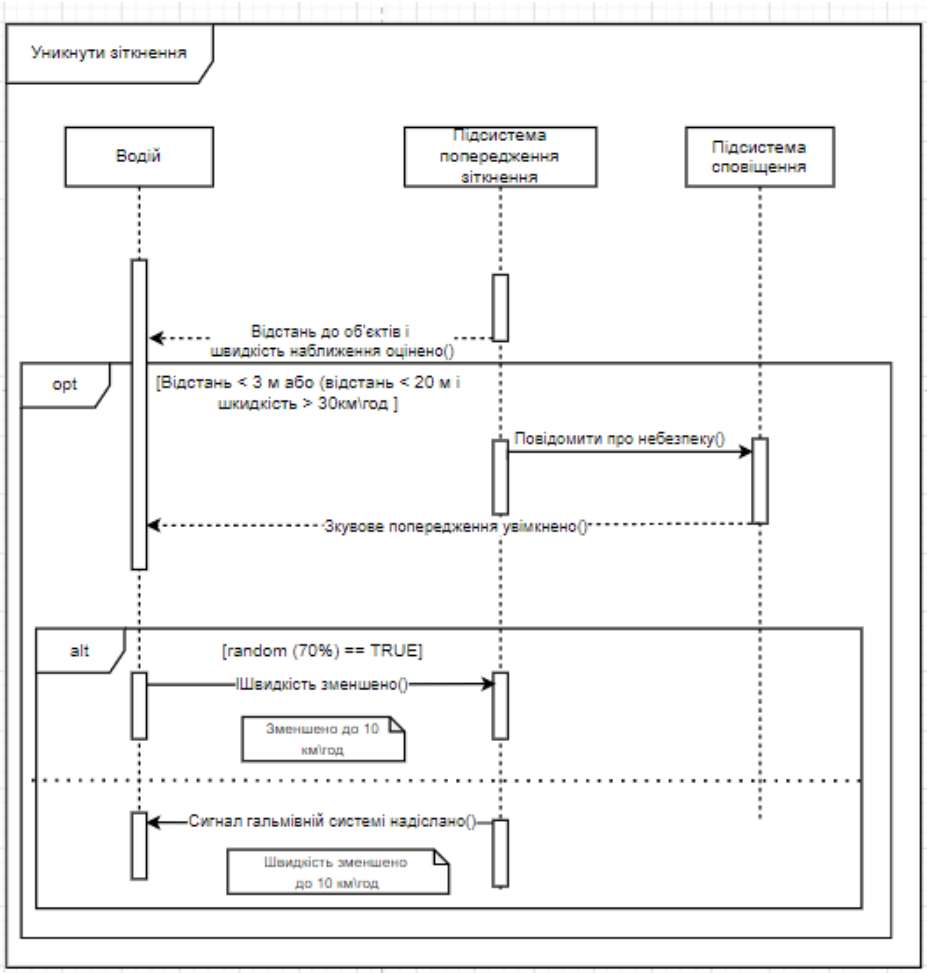

Підсистема автоматичного паркування
Подача живлення
Живлення нема
Очікує
Встановлення автоматичного режи
Камери, радар та датчики працюють
Авто припарковано
Водій вибрав режим
Режив встановлено
Встановлення напівавтоматичного режиму
Паркронік працює
Авто припарковано

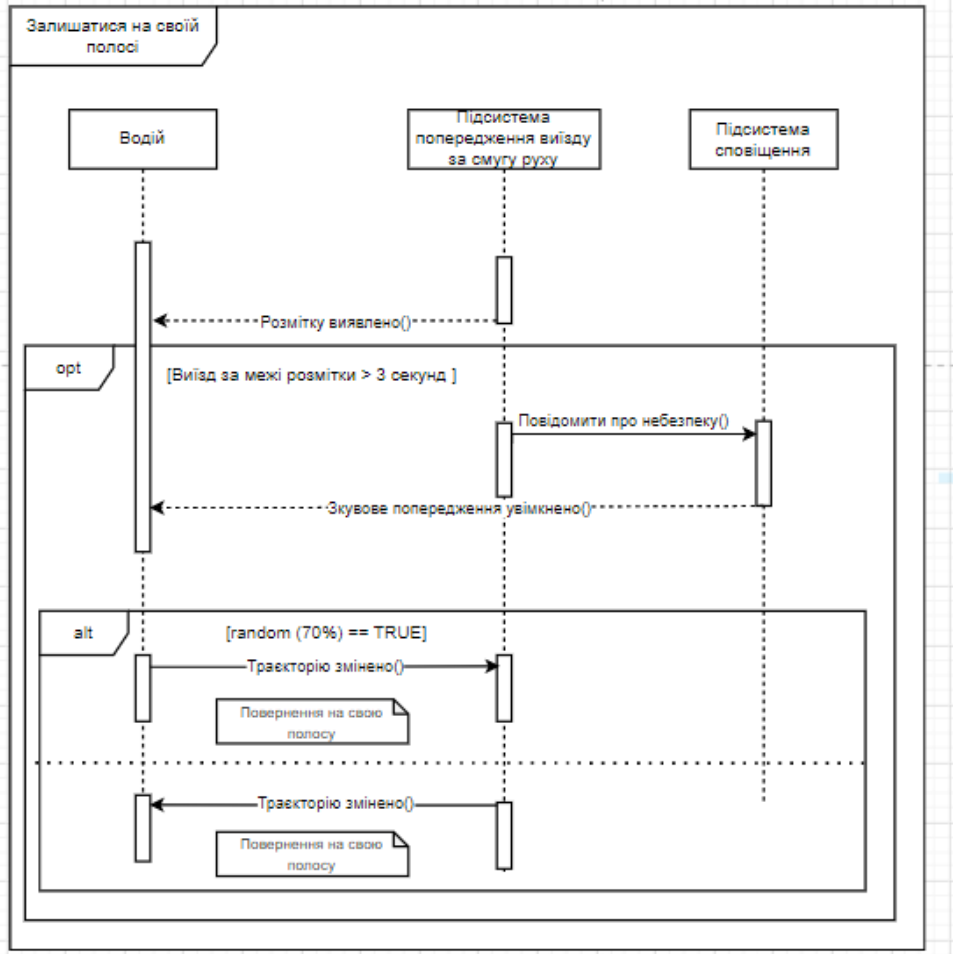
Лабораторна робота 3
SEQUENCE DIAGRAM

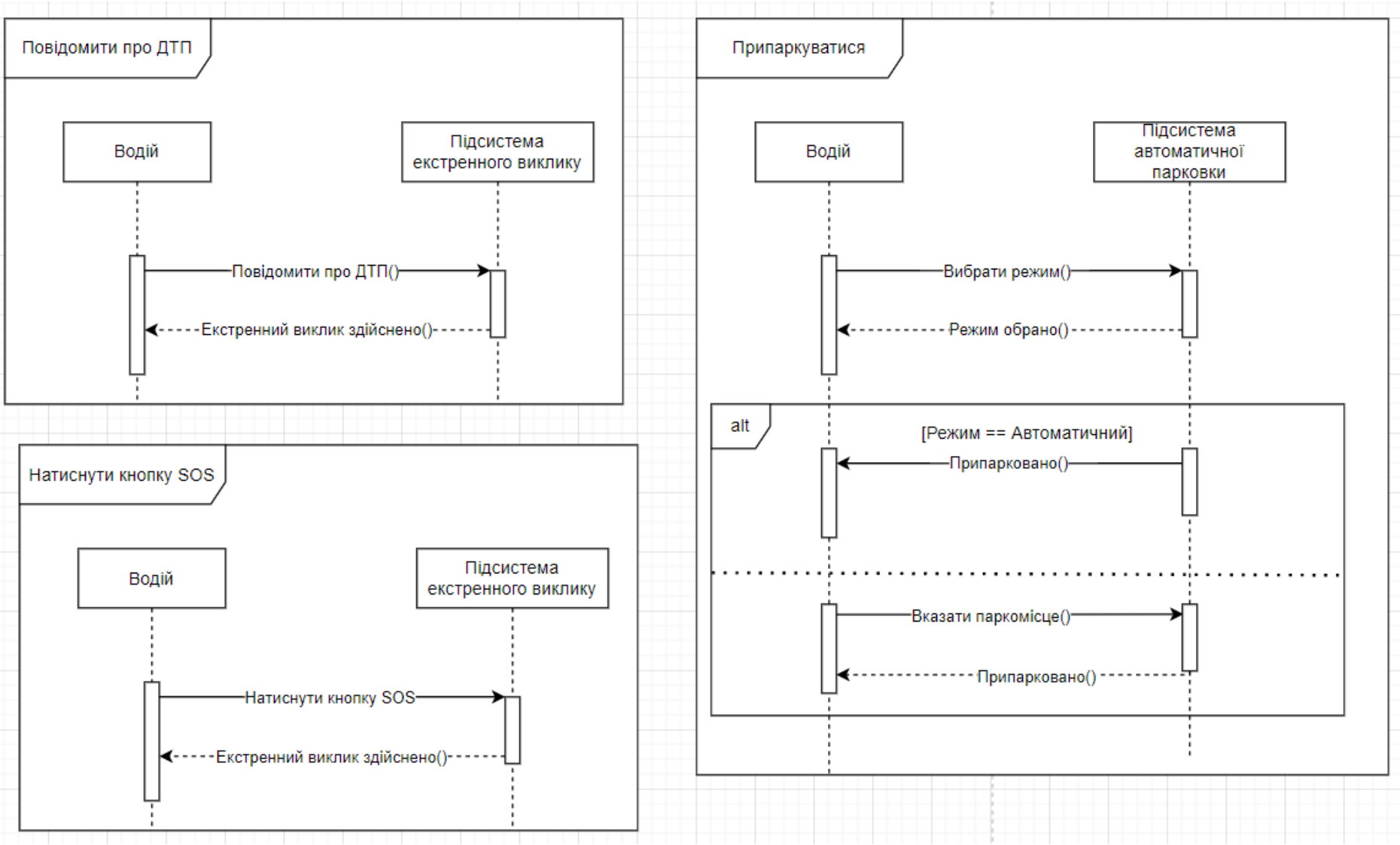



Встановити швидкість
Водій
Адаптивний круїз контроль
Підсистема сповіщення
Увімкнути круїз-контроль()
Увімкнено()
Встановити швидкість()
Швидкість співставлено з доволеною()
alt
[Встановлена швидкість+10 >
Дозволеної на участку дороги]
Повідомити про небезпеку()
Звукове попередження увімкнено()
Максимальну дозволену
Швидкість встановлено()
Встановлену водієм
Швидкість встановлено()

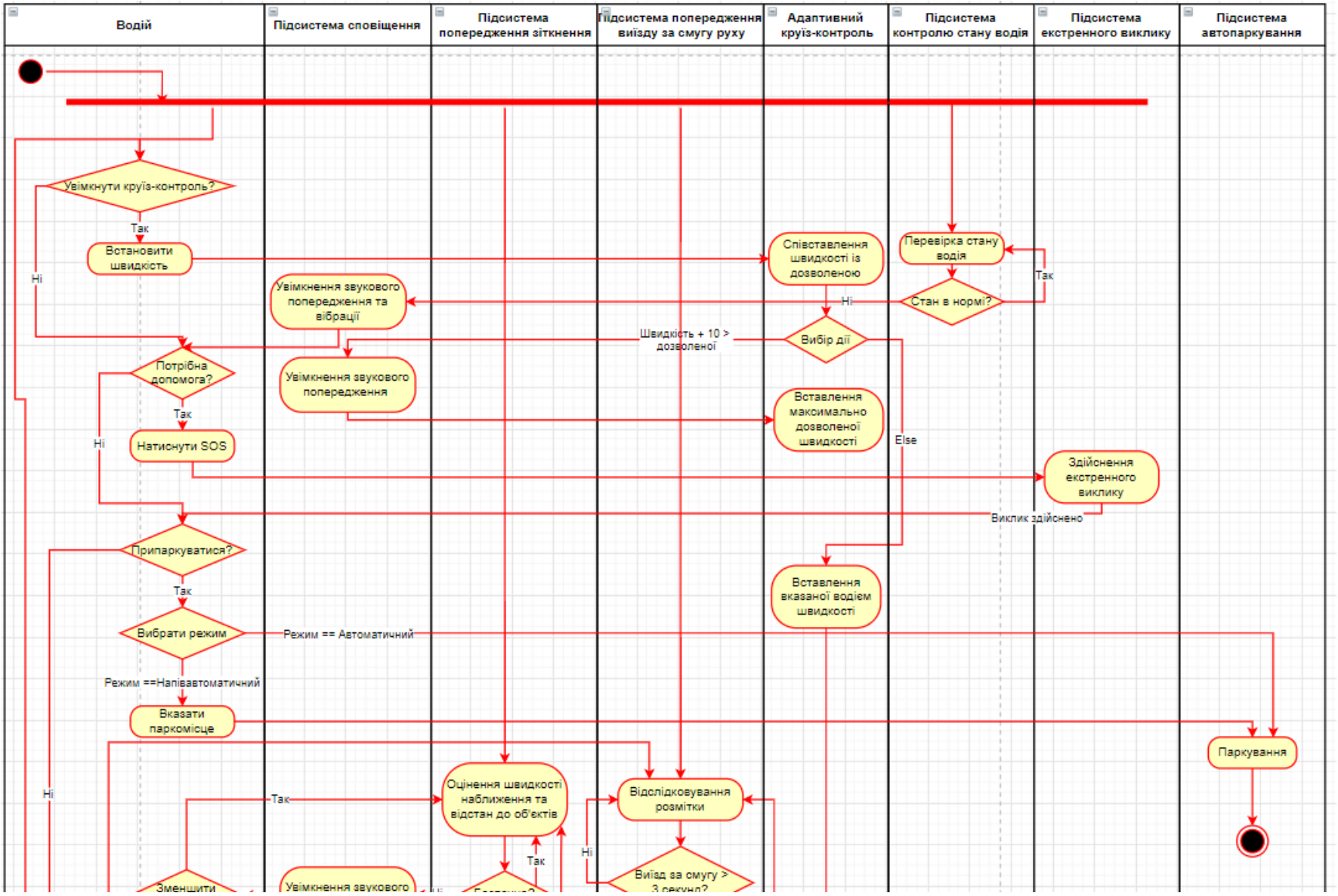

Проконтролювати свій стан
Водій
Підсистема контролю стану водія
Підсистема сповіщення
Проконтролювати свій стан()
opt
[Стан не відповідає нормі]
Повідомити про небезпеку()
Звукове попередження та вібрацію увімкнено()

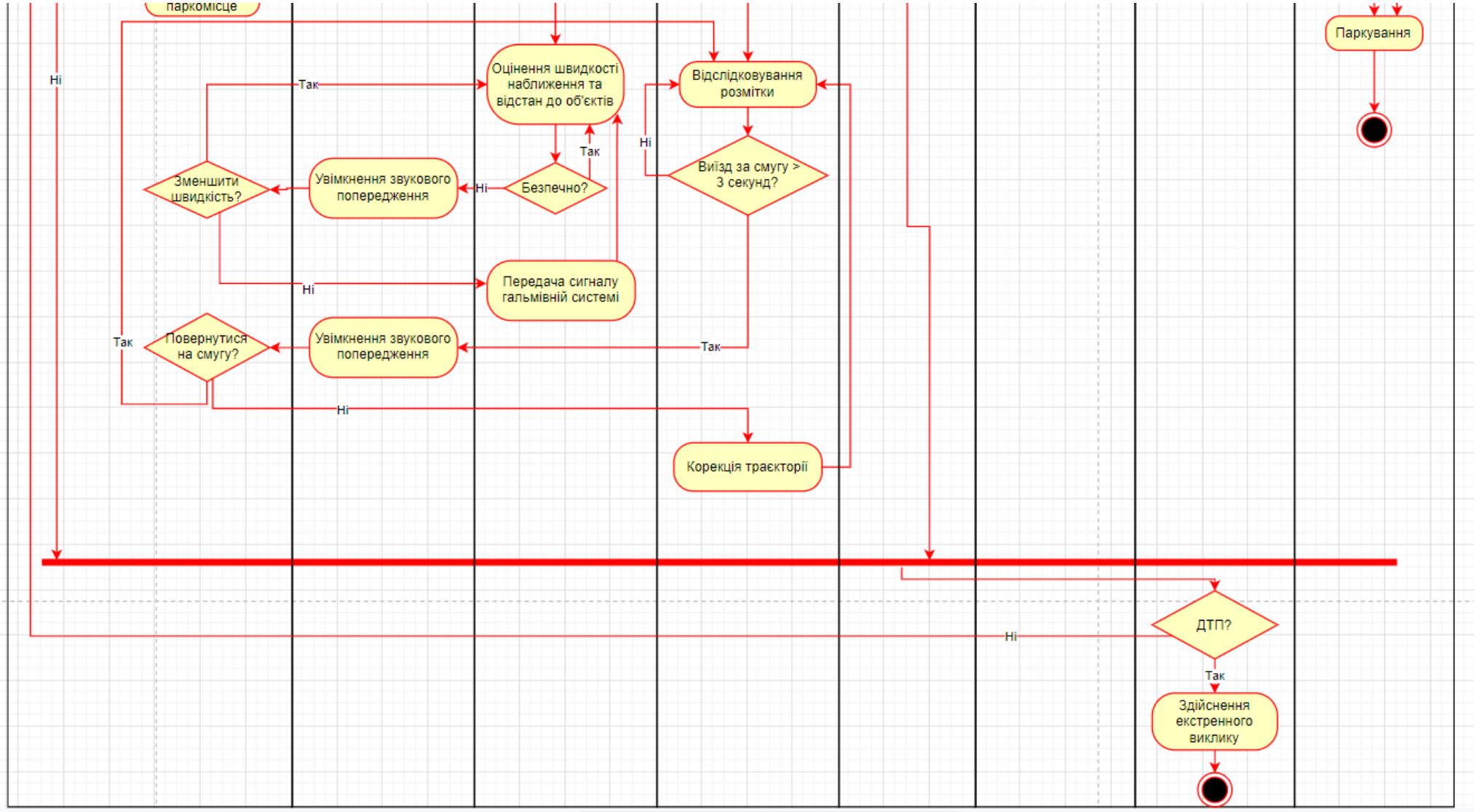

Уникнути зіткнення
Водій
Підсистема попередження зіткнення
Підсистема сповіщення
Відстань до об'єктів і швидкість наближення оцінено()
opt
[Відстань < 3 м або (відстань < 20 м і шкидкість > 30км\год]
Повідомити про небезпеку()
Зкувове попередження увімкнено()
alt
[random (70%) == TRUE]
ІШвидкість зменшено()
Зменшено до 10 км\год
Сигнал гальмівній системі надіслано()
Швидкість зменшено до 10 км\год


Залишатися на своїй полосі
Водій
Підсистема попередження виїзду за смугу руху
Підсистема сповіщення
Розмітку виявлено()
opt
[Виїзд за межі розмітки > 3 секунд]
Повідомити про небезпеку()
Зкувове попередження увімкнено()
alt
[random (70%) == TRUE]
Траєкторію змінено()
Повернення на свою полосу
Траєкторію змінено()
Повернення на свою полосу

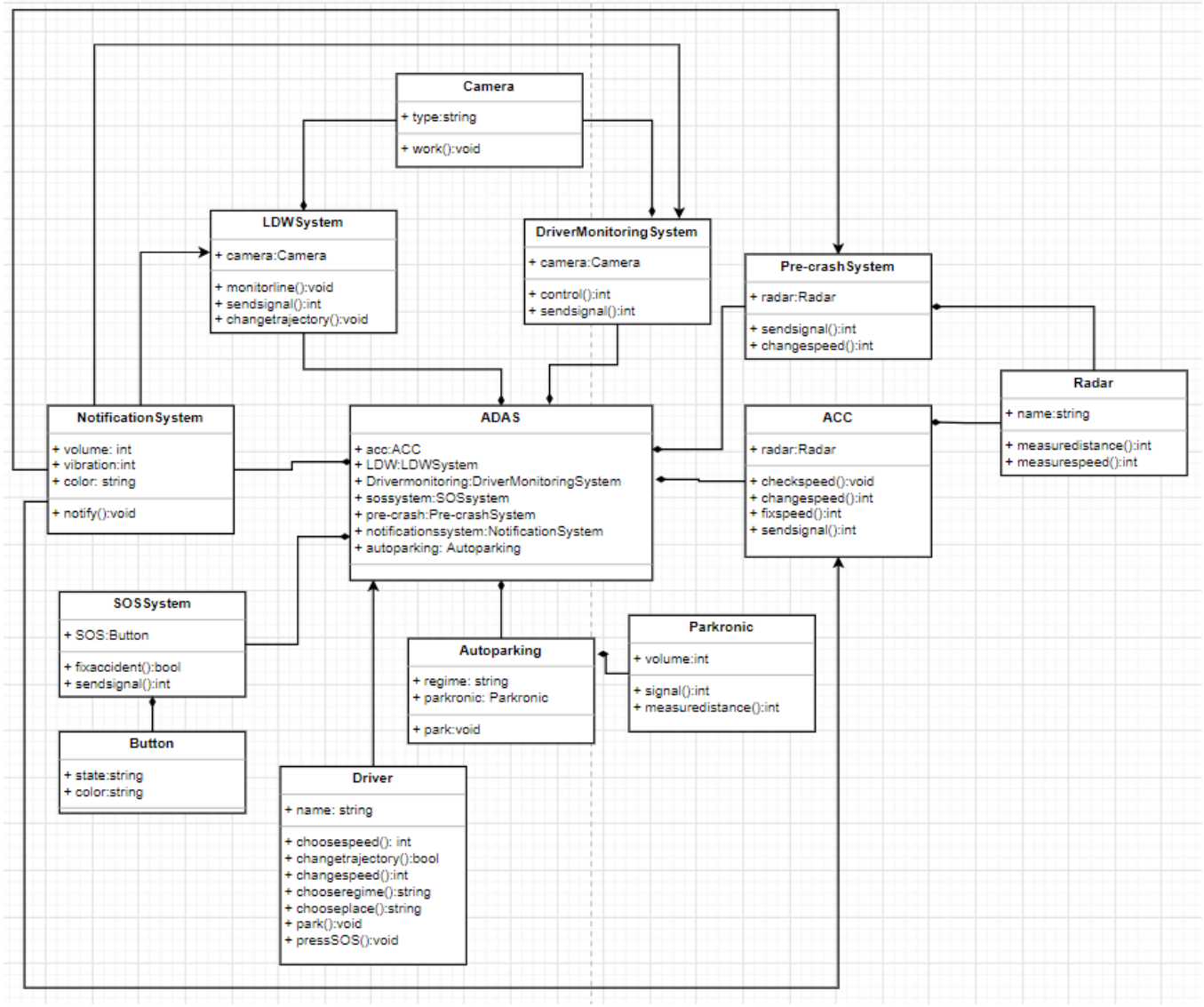

Повідомити про ДТП
Водій
Підсистема екстренного виклику
Повідомити про ДТП()
Екстренний виклик здійснено()
Натиснути кнопку SOS
Водій
Підсистема екстренного виклику
Натиснути кнопку SOS
Екстренний виклик здійснено()
Припаркуватися
Водій
Підсистема автоматичної парковки
Вибрати режим()
Режим обрано()
alt
[Режим == Автоматичний]
Припарковано()
Вказати паркомісце()
Припарковано()

ACTIVITY DIAGRAM




паркомісце
Ні
Так
Оцінення швидкості наближення та відстан до об'єктів
Відслідковування розмітки
Паркування
Так
Ні
Безпечно?
Увімкнення звукового попередження
Зменшити швидкість?
Ні
Виїзд за смугу > 3 секунд?
Передача сигналу гальмівній системі
Так
Повернутися на смугу?
Увімкнення звукового попередження
Так
Ні
Корекція траєкторії
ДТП?
Ні
Так
Здійснення екстренного виклику

CLASS DIAGRAM



Проект

Опис

Проект написаний на об’єктно-орієнтованій мові програмування С++. У процесі написання проекту було створено 13 класів(12 для системи допомоги водію та ще 1 – сам водій) , пов’язаних зв’язками композиції, асоціацій, використання.

Код програми

```
#include <iostream>
#include <string>
#include <vector>

using namespace std;

class Button {
public:
	string state;
	string color;
	Button() {
		state = "notpressed";
		color = "grey";
	}
};

class SOSSystem {
public:
	Button SOS;
	SOSSystem() {
		cout << "SOSSystem is ready to work!";
		cout << "\n\tSOS button is " << SOS.state << " and " << SOS.color << endl;
	}
	bool fixaccident(bool accident) {
		if (accident == true) {
			cout << "SOSSystem: Accident is fixed!" << endl;
			return true;
		}
		else{
			cout << "SOSSystem: All right!" << endl;
			return false;

		}
	}
	int sendsignal() {
		cout << "Help is called!" << endl;
		return 911;
	}
};



class Parkronic {
public:
	int volume;
	Parkronic();
	int signal();
	int measuredistance();
};

Parkronic::Parkronic() {
	volume = 0;
	cout << "Parcronic is ready to work" << endl;
}
```

```
int Parkronic::signal() {

	cout << "Parcronic:\n\tWarning! Car is very close to some object!";
		volume = 50;
	cout << "\n\tSignal: " << volume << endl;
	return volume;
}

int Parkronic::measuredistance() {
	cout << "Parcronic measure distance..." << endl;
	int distance = rand() % 100;
	return distance;
}

class Autoparking {
public:
	string regime;
	Parkronic parkronic;
	Autoparking() {

		cout << "Autoparking are ready to work!" << endl;
	}
	void park(string regime, string place) {
		this->regime = regime;
		cout << "Autoparking:\n\tCar is parking in " << regime << " regime"<< endl;
		if (this->regime != "auto") {
			int d = parkronic.measuredistance();
			if (d <= 30) {
				parkronic.signal();
			}
		}
		else {
			cout << "\tCar is parked on: " << place << endl;
		}
	}

};

class Radar {
public:
	string name;
	Radar() {
		name = "Someradar";
		cout << "Radar is ready to work!" << endl;
	}
		int measuredistance() {
		cout << "Radar messure distance!" << endl;
		int distance = rand() % 100;
		cout << "Distance: " << distance<<endl;
		return distance;
	}
	int measuresspeed() {
		cout << "Radar messure speed!" << endl;
		int  speed = rand() % 200;
		cout << " speed: " << speed<<endl;
		return  speed;

	}
};

class ACC {
public:
	Radar radar;
	ACC() {
		cout << "ACC is ready to work!" << endl;
	}
	int checkspeed(int speed, int maxspeed) {
		cout << "ACC check speed!" << endl;
```

```
		if (maxspeed+10 > speed) {
			cout << "All right!" << endl;
			return 0;
		}
		else {
			cout << "Speed is more that maxspeed on this way!" << endl;

			return sendsignal();

		}
	}
	int changespeed(int speed, int maxspeed) {
		speed = radar.measuresspeed();
		cout << "ACC must change speed!" << endl;
		speed = maxspeed;
		cout << "New speed is " << speed << endl;
		return speed;
	}
	int fixspeed(int speed) {

			cout << "Speed is fixed!" << endl;
			return speed;

	}

	int sendsignal() {
		return 1;
	}

};
class PrecrashSystem{
public:
	Radar radar;
	PrecrashSystem() {
		cout << "PrecrashSystem is ready to work!" << endl;
	}
	int sendsignal() {
		return 1;
	}
	int changespeed(int speed, int maxspeed) {
		speed = maxspeed;
		cout << "PrecrashSystem change speed! New speed: " << speed<<endl;
		return speed;
	}
};

class Camera {
public:
	string type;
	Camera() {
		type = "sometype";
		cout << "Camera exist!" << endl;

	}
	void work() {
		cout << "Camera is working..." << endl;
	}
};



class DriverMonitorSystem {
public:
	Camera camera;
	int control(int driver) {
		camera.work();
		cout << "DriverMonitorSystem control driver" << endl;
		int dr = driver;
```

```
		return dr;
	}
	int sendsignal() {
		return 2;
	}
};
class LDWSystem {
public:
	Camera camera;

	LDWSystem() {
		cout << "LDWSystem is ready to work!" << endl;
	}
	int monitorline(int line) {
		cout << "LDWSystem monitor line!" << endl;
		if (line != 1) {
			return sendsignal();
		}
		else return 1;
	}
	int sendsignal() {
		return 4;
	}
	int changetrajectory() {
		cout << "Trajectory is changed!" << endl;
		return 1;
	}

};
class NotificationSystem {
public:
	int volume;
	int vibration;
	string color;
	NotificationSystem() {
		vibration = 0;
		volume = 0;
		color = "green";
		cout << "NotificationSystem is ready to work!\nColor of display is " << color
<< endl;
	}
	void notify(int i) {
		vibration = 10;
		volume = 10;
		color = "red";
		cout << "NotificationSystem: Warning!\n";
		cout << "Vibration on! Color of display is " << color << endl;
		if (i == 1) {
			cout << "Choosed speed is more than maxspeed on this way!" << endl;
		}

		else if (i == 2) {
			cout << "Dear driver, you can not drive! Stop a car!" << endl;

		}
		else if (i == 3) {
			cout << "Warning! Some object is near! Speed must be decrease!" <<
endl;
		}
		else if (i == 4) {
			cout << "Warning! Car leaves line! Driver, change trajectory!" << endl;
		}


	}

};
```

```
class ADAS {
public:
	Autoparking autoparking;
	SOSSystem sossystem;
	PrecrashSystem  precrashsystem;
	ACC acc;
	DriverMonitorSystem drivermonitorsystem;
	LDWSystem ldwssystem;
	NotificationSystem  notificationsystem;

	ADAS() {
		cout << "ADAS is ready to work!" << endl;
	}
};



class Driver {
public:
	string name;
	int state;
	Driver() {
		name = "Maria";
		cout << "Hello, i am " << name << " and i have car with ADAS!" << endl;
		state = 1;
	}
	void park() {
		cout << "I want to park my car!" << endl;
	}
	string chooseregime() {
		string regime;
		cout << "Choose regime: \n\t  auto or semiauto" << endl;
		cin >> regime;

		cout << "I want to park my car in " << regime << " regime" << endl;
		return regime;
	}
	string chooseplace() {
		string place;
		cout << "Choose place: ";
		cin >> place;

		cout << "Car will parked in " << place<<endl;
		return place;
	}

	int choosespeed() {
		cout << "Choose speed: ";
		int speed;
		cin >> speed;
		return speed;
	}

	int changespeed() {
		cout << "Changed speed: ";
		int speed;
		cin >> speed;
		return speed;
	}
	bool changetrajectory() {
		cout << "Change trajectory?: ";
		bool trajectory;
		cin >> trajectory;
		return trajectory;
	}

	void pressSOS(ADAS& adas) {
```

```
                cout << "SOS is pressed!" << endl;
                adas.sossystem.SOS.state = "pressed";
                adas.sossystem.SOS.color = "red";
                cout << "SOS button is: " << adas.sossystem.SOS.state << " and " <<
adas.sossystem.SOS.color << endl;
                adas.sossystem.sendsignal();
        }
};

void help(Driver driver, ADAS adas) {
        cout << "Driver, do you need help? ";
        bool help;
        cin >> help;
        if (help) {
                driver.pressSOS(adas);
        }
}

int park(Driver driver, ADAS adas) {
        cout << "Driver, do you want to park? ";
        bool park;
        cin >> park;
        if (park) {
                driver.park();
                string regime = driver.chooseregime();
                string place = driver.chooseplace();
                adas.autoparking.park(regime, place);
                return 0;
        }
        else return 1;
}

int sped(Driver driver, ADAS &adas, int nowspeed) {
        cout << "Driver, do you want enable cruise-control?";
        bool speed;
        cin >> speed;
        if (speed) {
                int speed = driver.choosespeed();
                int maxspeed = rand() % 200;

                int acc = adas.acc.checkspeed(speed, maxspeed);
                if (acc == 1) {
                        adas.notificationsystem.notify(acc);
                        speed = adas.acc.changespeed(speed, maxspeed);
                        speed = adas.acc.fixspeed(speed);
                }
                else {
                        speed = adas.acc.fixspeed(speed);
                }

        }
        return speed;

}

int main() {
        Driver driver;
        ADAS adas;
        int speed = adas.precrashsystem.radar.measuresspeed();
        int distance = adas.precrashsystem.radar.measuredistance();

        while (true) {
                int accident = 0 + rand() % 1;
                if (adas.sossystem.fixaccident(accident)) {
                        adas.sossystem.sendsignal();
                        return 0;
                }
                int speed = adas.precrashsystem.radar.measuresspeed();
```

```
        int distance = adas.precrashsystem.radar.measuredistance();
        double danger = speed / distance;

        while (danger > 0.3) {
               adas.notificationsystem.notify(3);
               speed = driver.changespeed();
               danger = speed / distance;
               if (danger > 0.3) {
                      speed = adas.precrashsystem.changespeed(speed, distance * 0.3);
                      danger = speed / distance;
               }
        }

        int line = 0 + rand() % 1;
        line = adas.ldwssystem.monitorline(line);
        while (line != 1) {
               adas.notificationsystem.notify(4);
               line = driver.changetrajectory();
               if (line != 1) {
                      line = adas.ldwssystem.changetrajectory();
               }

        }

        driver.state = 0 + rand() % 1;
        int driverstate = adas.drivermonitorsystem.control(driver.state);
        if (driverstate != 1) {
               adas.notificationsystem.notify(adas.drivermonitorsystem.sendsignal());
               help(driver, adas);
               int a = park(driver, adas);
               if (a == 0) return 0;;

        }

        speed = sped(driver, adas, speed);
        int a = park(driver, adas);
        if(a == 0) return 0;
}


        return 0;
}
```

Сценарій 1. Машина їде. Водію стало погано за рулем

Коли водій їде за рулем, то ніхто не застрахований від того, що водію раптово може стати погано. Коли водію стало зле, він може викликати екстрену допомогу за допомогою натискання кнопки “SOS”. Проте не факт, що водій буде в змозі це зробити. Спочатку система, побачивши, що водій не в нормі, видасть відповідне голосове повідомлення. Якщо водій ніяк не відреагує, рух продовжиться. Проте в разі система може сама взяти в руки керування якщо водій ніяк не реагує. Після виклику допомоги водій може(якщо фізично може і хоче) припаркувати машину у авто- або напіватоматичному режимі.

1) Водій натиснув кнопку виклику допомоги і паркує машину в автоматичному режимі.

Для паркування в автоматичному режимі водію необхідно лише вказати місця для парковки.

```
Radar messure speed!
 speed: 41
Radar messure distance!
Distance: 67
Camera is working...
DriverMonitorSystem control driver
NotificationSystem: Warning!
Vibration on! Color of display is red
Dear driver, you can not drive! Stop a car!
Driver, do you need help? 1
SOS is pressed!
SOS button is: pressed and red
Help is called!
Driver, do you want to park? 1
I want to park my car!
Choose regime:
          auto or semiauto
auto
I want to park my car in auto regime
Choose place: somelace
Car will parked in somelace
Autoparking:
        Car is parking in auto regime
        Car is parked on: somelace
```

2) Водій натиснув кнопку виклику допомоги і паркує машину в напівавтоматичному режимі

В цьому випадку включається паркронік. Він буде сигналізувати про наближення до якого об’єкту.

```
DriverMonitorSystem control driver
NotificationSystem: Warning!
Vibration on! Color of display is red
Dear driver, you can not drive! Stop a car!
Driver, do you need help? 1
SOS is pressed!
SOS button is: pressed and red
Help is called!
Driver, do you want to park? 1
I want to park my car!
Choose regime:
          auto or semiauto
semiauto
I want to park my car in semiauto regime
Choose place: someplace
Car will parked in someplace
Autoparking:
        Car is parking in semiauto regime
Parcronic measure distance...
Parcronic:
        Warning! Car is very close to some object!
        Signal: 50
```

3) Водій натиснув кнопку виклику допомоги і не паркує машину

У цьому разі продовжується рух. Система сама може вести машину, вирівнюючи траєкторію та регулюючи швидкість.

4) Водій не реагує

Видасться голосове попередження і продовжиться рух.

```
ADAS is ready to work!
Radar messure speed!
 speed: 41
Radar messure distance!
Distance: 67
Camera is working...
DriverMonitorSystem control driver
NotificationSystem: Warning!
Vibration on! Color of display is red
Dear driver, you can not drive! Stop a car!
Driver, do you need help? 0
Driver, do you want to park? 0
SOSSystem: All right!
Radar messure speed!
 speed: 169
Radar messure distance!
Distance: 24
NotificationSystem: Warning!
Vibration on! Color of display is red
Warning! Some object is near! Speed must be decrease!
Changed speed: 169
PrecrashSystem change speed! New speed: 7
LDWSystem monitor line!
NotificationSystem: Warning!
Vibration on! Color of display is red
Warning! Car leaves line! Driver, change trajectory!
Change trajectory?: 0
Trajectory is changed!
Driver, do you want enable cruise-control?0
Driver, do you want to park? 0
```

Сценарій 2. Існує загроза зіткнення.

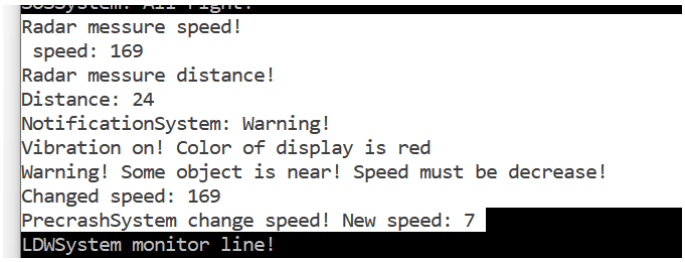
Коли авто наближається дуже близько до якогось об’єкту система спочатку видасть попередження. Якщо водій не прийме ніяких дій, не зменшить швидкість, система сама подасть сигнал гальмівній системі, щоб уникнути аварії.

1) Водій сам зменшує швидкість – реагує на попередження.

У цьому разі продовжується рух, система продовжує все моніторити і повідомляти про можливі небезпеки

```
Radar messure speed!
 speed: 169
Radar messure distance!
Distance: 24
NotificationSystem: Warning!
Vibration on! Color of display is red
Warning! Some object is near! Speed must be decrease!
Changed speed: 5
LDWSystem monitor line!
```

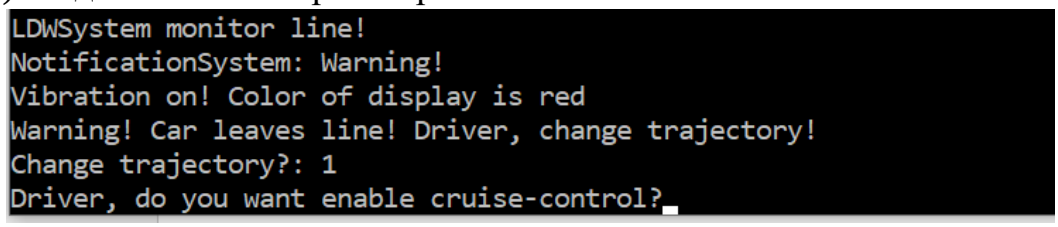
2) Водій не реагує



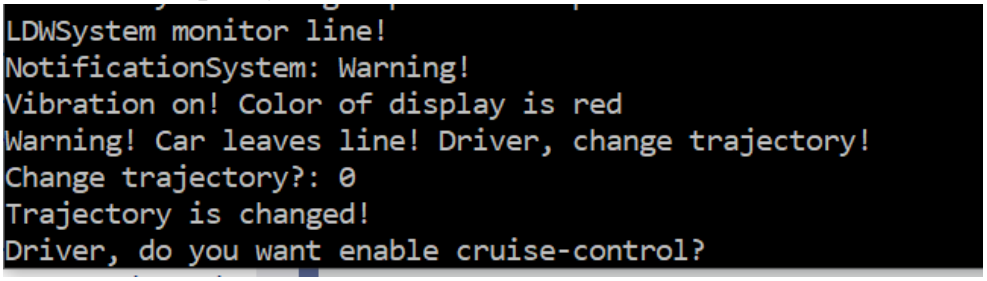
Сценарій 3. Водій виїхав за лінію розмітки.

Аналогічно як в попередньому сценарії. Водій чує попередження, реагує – система продовжує все моніторити, не реагує – система сама змінить траєкторію.

1) Водій сам міняє траєкторію



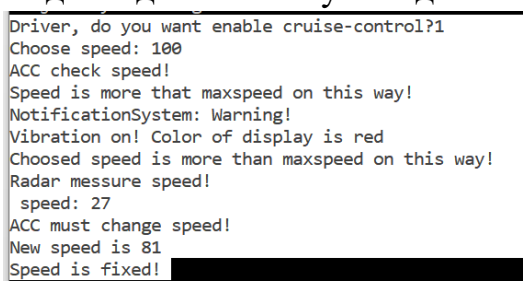
2) Водій не реагує



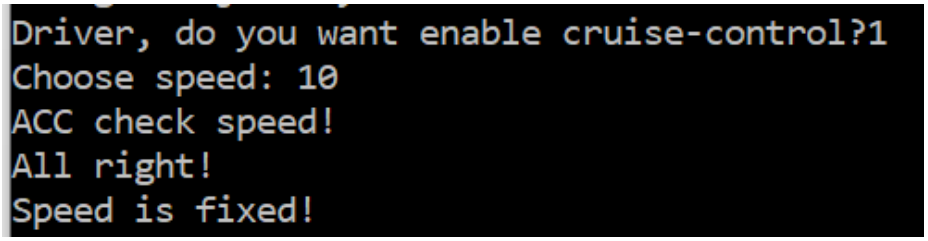
Сценарій 4. Водій вмикає круїз-контроль

Водій може увімкнути круїз-контроль, задавши певну швидкість. Якщо швидкість безпечна і не перевищує дозволену на даній ділянці дороги, то система зафіксує її. Якщо швидкість зависока – система встановить максимально можливу безпечну і дозволену швидкість

1) Водій задав зависоку швидкість

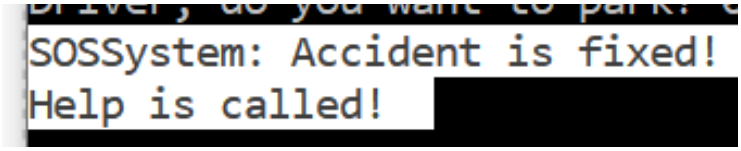


2) Водій задав дозволену безпечну швидкість



Сценарій 5. Сталася аварія

Якщо сталася аварія, система викличе екстренну допомогу.



Зміни, що були внесені в проект у ході написання коду

```
class ACC {
public:
	Radar radar;
	ACC() {
		cout << "ACC is ready to work!" << endl;
	}
	int checkspeed(int speed, int maxspeed) { // змінила void на int так, щоб функція повертала значення і сигналізувала у разі невідповідності швидкості
		cout << "ACC check speed!" << endl;
		if (maxspeed+10 > speed) {
			cout << "All right!" << endl;
			return 0;
		}
		else {
			cout << "Speed is more that maxspeed on this way!" << endl;

			return sendsignal();

		}
	}

class LDWSystem {
public:
	Camera camera;

	LDWSystem() {
		cout << "LDWSystem is ready to work!" << endl;
	}
	int monitorline(int line) {
		cout << "LDWSystem monitor line!" << endl;
		if (line != 1) {
			return sendsignal();
		}
		else return 1;
	}
	int sendsignal() {
		return 4;
	}
	int changetrajectory() { // так само поміняла тип значення, яке повертає функція, щоб знати чи діє якось система, чи ні
		cout << "Trajectory is changed!" << endl;
		return 1;
	}

};

class Driver {
public:
	string name;
	int state; //додала змінну для характеристики стану водія
	Driver() {
		name = "Maria";
		cout << "Hello, i am " << name << " and i have car with ADAS!" << endl;
		state = 1;
	}
…
```