Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)

(МАИ)

Факультет №3 «Системы управления, информатика и электроэнергетика»

Кафедра 307 «Цифровые технологии и информационные системы»

«Архитектура информационных систем»

Робот доставщик

Выполнили:

студенты группы М3О-213Б-23

**К.С. Колеватов**

**А.В. Дылевская**

Принял:

магистр кафедры 307

**А.Н. Максимов**

**1. Введение**

В современном мире автоматизация процессов доставки становится всё более актуальной задачей. Роботы доставщики могут значительно повысить эффективность логистических операций, снизить затраты и улучшить качество обслуживания. В данном отчёте рассматривается разработка робота доставщика, предназначенного для использования в коворкинге МАИ.

**2. Основания для разработки**

Разработка робота доставщика обусловлена необходимостью оптимизации доставки напитков студентам.

**3. Назначение разработки**

Робот доставщик предназначен для автоматической транспортировки напитков студентам.

**4. Требования к программе или программному изделию**

Программное обеспечение робота должно соответствовать следующим требованиям:

–Поддержка автономного навигационного модуля.

–Возможность интеграции с системами управления и мониторинга.

–Наличие алгоритмов для обхода препятствий и оптимизации маршрута.

–Поддержка удалённого управления и диагностики.

**5. Требования к программной документации**

Программная документация должна включать:

–Полное описание архитектуры программного обеспечения.

–Руководство пользователя и администратора.

–Документацию по установке, настройке и эксплуатации.

–Описание алгоритмов и протоколов взаимодействия.

**6. Стадии и этапы разработки**

Разработка робота доставщика включает следующие этапы:

–Анализ требований и проектирование.

–Разработка прототипа.

–Тестирование и отладка.

–Внедрение и эксплуатация.

–Мониторинг и сопровождение.

**7. Порядок контроля и приёмки**

Контроль и приёмка робота доставщика осуществляются в несколько этапов:

–Техническая экспертиза и проверка соответствия требованиям.

–Испытания в реальных условиях эксплуатации.

–Оценка надёжности и безопасности.

–Заключительная приёмка и ввод в эксплуатацию.

**8. Требования к функциональным характеристикам**

Робот доставщик должен обладать следующими функциональными характеристиками:

–Максимальная грузоподъёмность: 3 кг.

–Время автономной работы: не менее 8 часов.

–Возможность зарядки от стандартных источников питания.

**9. Требования к надёжности**

**10. Условия эксплуатации**

**11. Требования к составу и параметрам технических средств**

Робот должен быть оснащён следующими техническими средствами:

–Система навигации (GPS, лидары, камеры).

–Датчики для обнаружения препятствий.

–Система связи для удалённого управления.

13. Требования к информационной и программной совместимости

Робот должен быть совместим с:

–Системами управления и мониторинга.

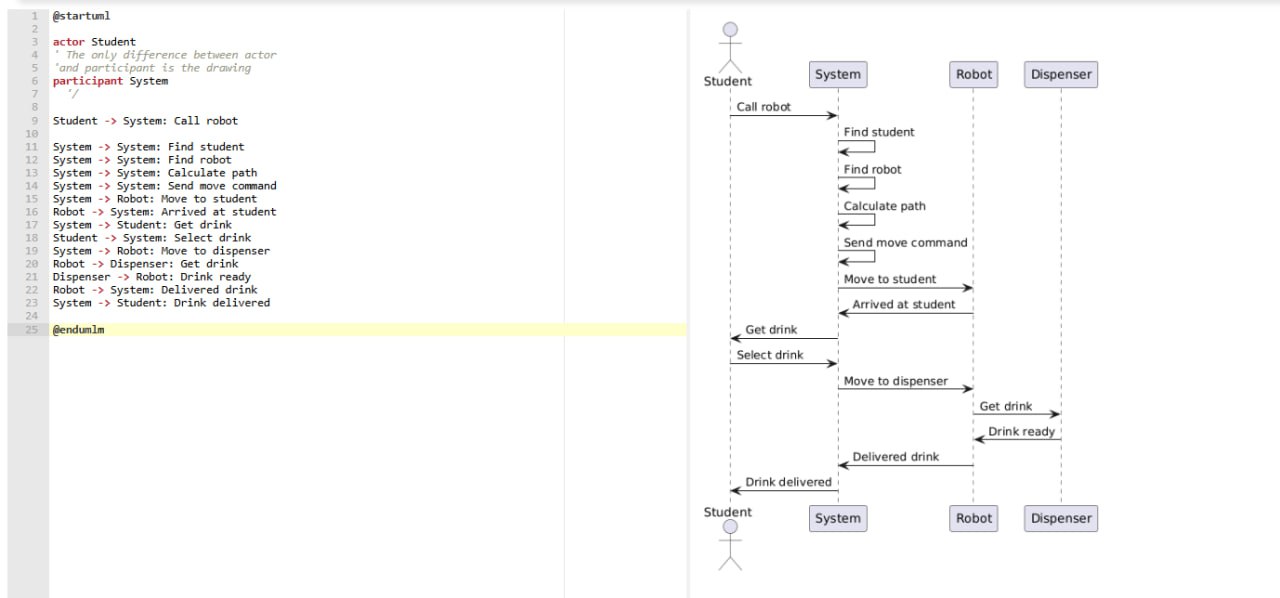
–Протоколами связи Wi-Fi и Bluetooth.

–Операционными системами Windows и Linux.

**Заключение**

Разработка робота доставщика для коворкинга МАИ является важным шагом в направлении автоматизации внутренних процессов доставки. Внедрение данного решения позволит повысить эффективность доставки и улучшить качество обслуживания.

**Приложение**

****

@startuml

actor Student

' The only difference between actor

'and participant is the drawing

participant System

'/

Student -> System: Call robot

System -> System: Find student

System -> System: Find robot

System -> System: Calculate path

System -> System: Send move command

System -> Robot: Move to student

Robot -> System: Arrived at student

System -> Student: Get drink

Student -> System: Select drink

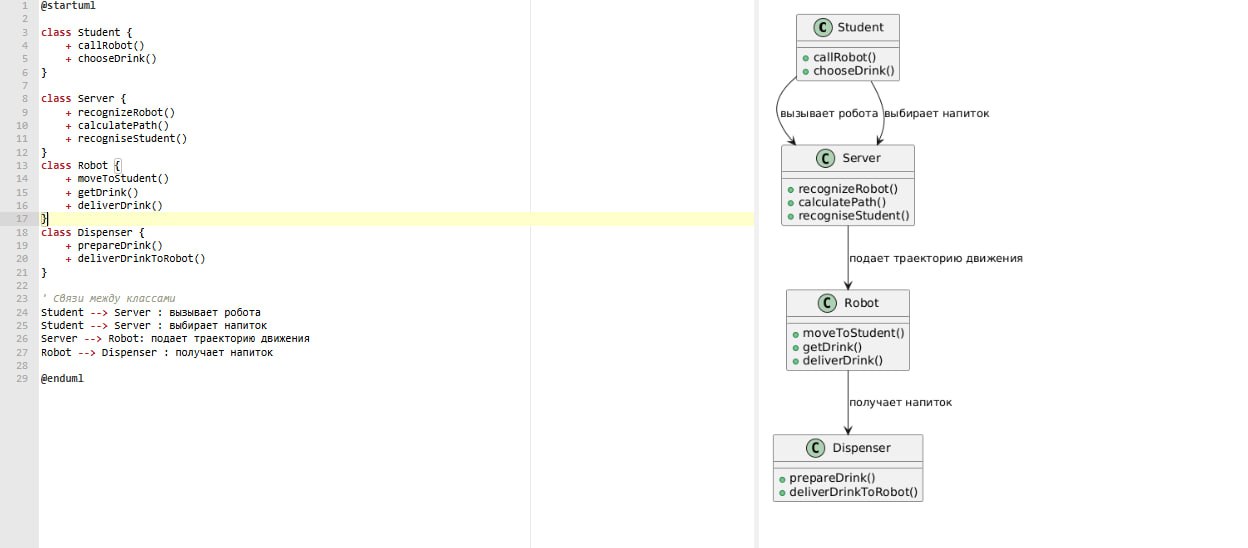
System -> Robot: Move to dispenser

Robot -> Dispenser: Get drink

Dispenser -> Robot: Drink ready

Robot -> System: Delivered drink

System -> Student: Drink delivered

@endumlm****

@startuml

class Student {

+ callRobot()

+ chooseDrink()

}

class Server {

+ recognizeRobot()

+ calculatePath()

+ recogniseStudent()

}

class Robot {

+ moveToStudent()

+ getDrink()

+ deliverDrink()

}

class Dispenser {

+ prepareDrink()

+ deliverDrinkToRobot()

}

' Связи между классами

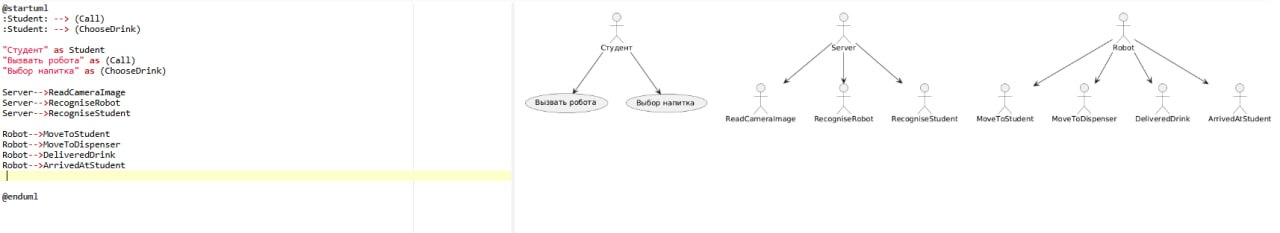
Student --> Server : вызывает робота

Student --> Server : выбирает напиток

Server --> Robot: подает траекторию движения

Robot --> Dispenser : получает напиток

@enduml



@startuml

:Student: --> (Call)

:Student: --> (ChooseDrink)

"Студент" as Student

"Вызвать робота" as (Call)

"Выбор напитка" as (ChooseDrink)

Server-->ReadCameraImage

Server-->RecogniseRobot

Server-->RecogniseStudent

@enduml

**Список используемой литературы**

**1.** ГОСТ 19.201-78 Единая система программной документации. Техническое задание. Требования к содержанию и оформлению. 1978. Режим доступа: http://protect.gost.ru/document.aspx?control=7&id=155153

**2.** Создание проекта форм интерфейса и карты диалоговых окон в PLANTUML [Электронный ресурс]. Режим доступа: https://habr.com/ru/post/279373/ (27.09.2020)