симуляторів дозволяють впроваджувати власні алгоритми поведінки з дотриманням внутрішніх правил системи. Кожен запуск сесії відпрацьовує по заданому сценарію який вбудовується розробниками, а по закінченню оцінюється його ефективність. Такий процес вимагає не тільки багато часу на коригування команд роботу під час випробувань, але й практичних знань та навичок для налаштування симуляції. Концепція програми, що розробляється, розв'язує цю проблему шляхом створення посередника між користувачем і симулятором Webots [3]. Зв’язок із середовищем Webots встановлюється за протоколом TCP/IP, де сервером виступає клієнтський додаток у якому виконуються обчислення, робота з базою даних, впровадження нових модулів і т.д. Клієнтом стає сесія симуляції у Webots. Аби мати можливість тестувати власні розробки управління роботехнікою, було імплементовано технологію динамічного завантаження класів що визначає модульний тип поведінки робота. Таким чином користувач розробляє власні Java-класи/бібліотеки в будь-якому редакторі, а потім через додаток імпортує створену поведінку в робототехніку й тестує власний алгоритм [4]. Керуючись імпортованим патерном, робот надсилає усі відомості з сенсорів і датчиків в реляційну базу даних SQL.

Результат роботи ПЗ - відображення пройденого маршруту, демонстрація руху техніки в реальному часі, карти місцевості, статистичних графіків та запис показників в спроєктовану базу [5]. Оскільки додаток сприймає лише колісну техніку з диференціальним приводом, на перспективу планується розширити набір робототехніки, а для оптимізації процесів обчислення планується впровадити технології багатопотоковості додатку.

Список використаної літератури:

1. «Промислові роботи: тренди й типи» [Електронний ресурс]. Режим доступу: https://www.controleng.com/in/robototehnika

2. Робот [Електронний ресурс]. Режим доступу: https://uk.wikipedia.org/ wiki

3. Webots [Електронний ресурс]. Режим доступу: https://en.wikipedia.org/ wiki/Webots

4. Why webots [Електронний ресурс]. Режим доступу: http://dspace.tneu.edu.ua/handle/316497/25072

5. Динамічне завантаження класів [Електронний ресурс]. Режим доступу: https://habr.com/ru/sandbox/62803

УДК 004.891.3

Литвинюк В.С., *студент 4 курсу спеціальності «Комп’ютерні науки»*
Антонов Ю.С., *доцент кафедри комп’ютерних наук та інформаційних технологій*

ОСОБЛИВОСТІ АРХІТЕКТУРИ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ УСУНЕННЯ ПРОБЛЕМ З ТРАНСПОРТНИМ ЗАСОБОМ

Донецький національний університет імені Василя Стуса, м. Вінниця

На сьогоднішній день IT технології міцно пов’язані з усіма сферами нашого життя. Не є виключенням в цьому плані і транспортні засобі та пов’язані з ними сфери діяльності людини [1-2]. Окрім того постійно з’являються нові або вдосконалюються старі види транспортних засобів, наприклад самокати та велосипеди обладнані електричними двигунами та сучасними видами акумуляторів. У випадку виходу з ладу транспортного засобу перед його власником постає питання «викликати відповідну ремонтну службу або проблема така, що швидше та легше її усунити власноруч?». Зрозуміло, що більшості власників може не вистачити знань для діагностики та усунення несправності. Тому розробка програм підтримки прийняття рішень у цьому напрямку є актуальною.

Мета цієї роботи розглянути деякі особливості реалізації такої системи підтримки прийняття рішеь.

Так у роботі [2] авторами було запропоновано реалізувати подібну систему використовуючи фіксований набір питань та вагових коєфіцієнтів (метод електра) [3-4]. Однак недоліком такої системи є фіксований набір питань, що був жорстко прошитий у систему. У реальному житті у різних користувачів можу бути транспортні засоби різних марок або моделей, а один й той самий користувач може мати декілька транспортних засобів.

Зметою усунення вказаних вище недоліків та полегшення супроводу та розробки такої системи, автори пропонують використовувати формат JSON, який буде містити інформацію необхідну для вирішення задачі інформацію, а саме (рис. 1): питання та вагові коєфіцієнти, марку та модель транспортного засобу, номер версії. Рішення із використанням JSON формату надає можливість використати один шаблон для створення багатьох діагностик без змін коду програми [5].

Для користувацького інтерфейсу буде розроблено декілька екранів. На першому буде відображено лише назву додатку, інформацію про нього та кнопку, що перенаправляє вас до самих тестувань. Другий екран тестувань буде містити коротку інформацію про саме тестування та кнопку перейти до тестування. Третій екран самого тестування міститиме на собі лише запитання та результат.

```
{
    "question": "Чи є паливо в паливному баці?",
    "value": 0.2
},
{
    "question": "Чи присутня іскра у свічках запалювання?",
    "value": 0.1
}...
```

Рисунок 1 Питання та коефіцієнти в форматі JSON

Дуже великим плюсом буде можливість використовувати цей додаток без інтернету та при кожній можливості оновлювати його базу запитань, що чітко видно з діаграми розгортування на (рис. 2).

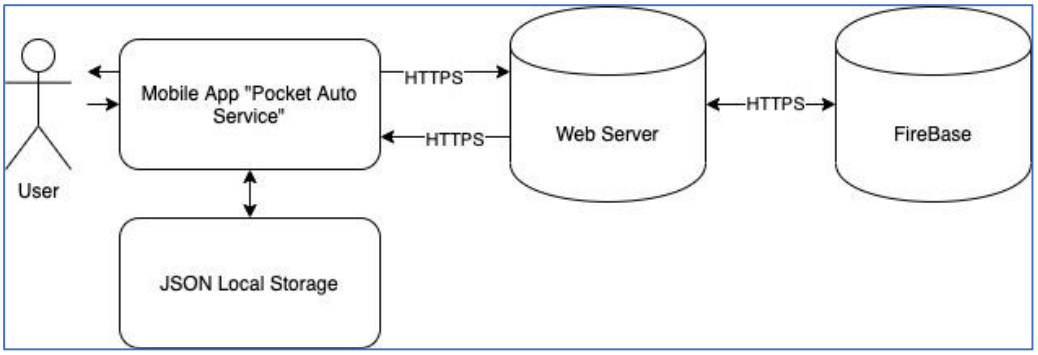

Рисунок 2 Діаграма розгортування

Розглянута система підійде для швидкої діагностики незначних та тривіальних проблем з автомобілем або іншим транспортним засобом, але при складних несправностях з великою вирогідністю буде необхідним візит на СТО.

Список віикористаної літератури

1. Черненко К.С., Макаров М.В., Антонов Ю.С., Бібліотеки комп'ютерного зору та проблеми керування транспортними засобами. *Матеріали наукової конференції професорсько-викладацького складу, наукових працівників і здобувачів наукового ступеня за підсумками науково-дослідної роботи за період 2017–2018 рр. (16–17 травня 2019 р.): у 2-х томах*. Том 2. Вінниця: Донецький національний університет імені Василя Стуса, 2019. с. 115-117. URL: http://jpvs.donnu.edu.ua/article/view/7247/7275 (дата звертання: 21.04.2021)
2. В.С. Литвинюк, Ю.С. Антонов, Розробка системи підтримки приняття рішень для усунення проблем з транспортним засобом. *Матеріали всеукраїнської науково-практичної конференції для студентів, аспірантів та молодих вчених "Прикладні інформаційні технології" (29 квітня 2020 року) - Вінниця: ДонНУ імені Василя Стуса*. URL: https://jait.donnu.edu.ua/article/view/8893 (дата звертання: 21.04.2021)
3. Метод электра. URL: https://naparah.com/intellektualnye-sistemy-prinyatiya-reshenij/0804870.html (дата звертання: 21.04.2021)
4. ELECTRE URL: https://en.m.wikipedia.org/wiki/ELECTRE (дата звертання: 21.04.2021)
5. Jeremy McPeak. Beginning javascript. Wrox, 2015. 768 p.