МАГІСТЕРСЬКА РОБОТА

МР. ІПм - 46.00.00.000 ПЗ

Група ІПм-24-3

Петриків Дмитро

2025

Івано-Франківський національний технічний університет нафти і газу

Факультет інформаційних технологій

Кафедра інженерії програмного забезпечення

Петриків Дмитро Андрійович

(прізвище, ім'я, по батькові)

УДК 004.9

(індекс)

МАГІСТЕРСЬКА РОБОТА

Програмні моделі та методи безпечної взаємодії автономних транспортних сутностей

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Петриків Д.А.

(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Юрчишин Володимир Миколайович, д.т.н., професор

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту

Завідувач кафедри

доц. Бандура В.В.

(посада) (підпис) (дата) (ініціали та прізвище)

Нормоконтроль

доц. Вовк Р.Б.

(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Івано-Франківськ – 2025

Івано-Франківський національний технічний університет нафти і газу

Факультет інформаційних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень магістр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Зав. кафедрою **ІПЗ**
доц. В.В. Бандура
“ 04 ” вересня 2025 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Петриківу Дмитру Андрійовичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи “ Програмні моделі та методи безпечної взаємодії автономних транспортних сутностей ”

керівник проекту (роботи) Юрчишин В.М., д.т.н., професор кафедри ІПЗ

затверджені наказом закладу вищої освіти від “ 05 ” листопада 2025 р. № 695/7

2. Строк подання студентом проекту (роботи) 15 грудня 2025 р.

3. Вихідні дані до проекту (роботи) Концепції та формальні моделі і методи побудови програмних технологій взаємодії транспортних сутностей

4. Зміст розрахунково - пояснювальної записки(перелік питань, які потрібно розробити)

1. Дослідження предметної області безпечної взаємодії автономних транспортних сутностей
2. Методи та алгоритми безпечної взаємодії сучасних автономних транспортних систем
3. Методологія проектування безпечної взаємодії автономних транспортних сутностей
4. Застосування фреймворку Arrowhead для забезпечення захищеності SOA

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Загальна архітектура системи платонування вантажівок (рис. 1.1)
2. Спрощена архітектура IoT, застосована до транспортних засобів (рис. 1.2)
3. Транспортні засоби та дорожня система, підключені через VANET (рис. 1.3)
4. Архітектура системи автономного керування з точки зору технічної перспективи (рис. 1.4)
5. Загальний огляд різних рівнів автоматизації керування (рис. 1.5)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата
Перевірка на плагіат	доц., к.т.н. Вовк Р.Б.	

7. Дата видачі завдання 04 вересня 2025 р.

Керівник _____
(підпис)

Завдання прийняв до виконання_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назви етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Підбір і вивчення літератури по темі магістерської роботи	15.09.2025	виконано
2	Дослідження предметної області безпечної взаємодії автономних транспортних сутностей	01.10.2025	виконано
3	Методи та алгоритми безпечної взаємодії сучасних автономних транспортних систем	17.10.2025	виконано
4	Методологія проектування безпечної взаємодії автономних транспортних сутностей	02.11.2025	виконано
5	Застосування фреймворку Arrowhead для забезпечення захищеності SOA	19.11.2025	виконано
6	Імплементація методології на прикладі колони автономних транспортних засобів	02.12.2025	виконано
7	Затвердження пояснювальної записки роботи завідувачем кафедри	15.12.2025	виконано

Студент – магістр _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 77 с., 27 рис., 7 табл., 41 джерело.

Тема: Програмні моделі та методи безпечної взаємодії автономних транспортних сутностей

Метою роботи є розроблення програмних моделей, алгоритмів та методології проєктування, які забезпечують безпечну та захищену взаємодію автономних транспортних сутностей.

Об'єктом дослідження є процес взаємодії автономних та підключених транспортних сутностей у динамічному середовищі дорожнього руху.

Предметом дослідження є методи, програмні моделі та алгоритми забезпечення безпечної кооперативної взаємодії автономних транспортних систем на основі V2V-комунікацій, відносного позиціонування та формальних моделей верифікації.

Результати дослідження

В роботі виконано інтеграцію контрактного аналізу з фреймворком Arrowhead, що забезпечує формальну верифікацію та підвищення рівня кіберзахищеності транспортних систем

Висновок

Виконано практичну імплементацію методології на прикладі колони автономних транспортних засобів, що дозволило продемонструвати ефективність запропонованих рішень у задачах платонування.

АВТОНОМНІ ТРАНСПОРТНІ СИСТЕМИ; ПІДКЛЮЧЕНІ ТРАНСПОРТНІ ЗАСОБИ; ВІДНОСНЕ ПОЗИЦІОНУВАННЯ; ПРОГНОЗУВАННЯ ТРАЄКТОРІЙ; СИСТЕМИ ПОПЕРЕДЖЕННЯ; КОНТРАКТНИЙ ПІДХІД; ІНТЕЛЕКТУАЛЬНІ ТРАНСПОРТНІ СИСТЕМИ.

ABSTRACT

Master Thesis: 77 pp., 27 fig., 7 tab., 41 sources.

Topic: Software models and methods for safe interaction of autonomous transport entities

The purpose of the work is to develop software models, algorithms and design methodologies that ensure safe and secure interaction of autonomous transport entities.

The object of the study is the process of interaction of autonomous and connected transport entities in a dynamic traffic environment.

The subject of the study is methods, software models and algorithms for ensuring safe cooperative interaction of autonomous transport systems based on V2V communications, relative positioning and formal verification models.

Research results

The work integrates contract analysis with the Arrowhead framework, which provides formal verification and increases the level of cybersecurity of transport systems

Conclusion

The methodology was implemented in practice using the example of a convoy of autonomous vehicles, which allowed demonstrating the effectiveness of the proposed solutions in platooning tasks.

AUTONOMOUS TRANSPORT SYSTEMS; CONNECTED VEHICLES; RELATIVE POSITIONING; TRAJECTORY PREDICTION; WARNING SYSTEMS; CONTRACT APPROACH; INTELLIGENT TRANSPORT SYSTEMS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AV - Autonomous Vehicle

BCL - Behavioural Contract Language

CAT - Contract Analysis Tool

DSRC - Dedicated Short-Range Communication

FCW - Forward Collision Warning

PAG - Platoon Agent

RLR - Red Light Running

WWD - Wrong-Way Driving

HV - Host Vehicle

PA - Platoon Accepted

CPS - Cyber-Physical Systems

BSM - Basic Safety Message

PJRQ - Platoon Join Request

PL - Platoon Leader

CV - Connected Vehicle

WWEW - Wrong-Way Entry Warning

BSW - Blind Spot Warning

CAV - Connected Autonomous Vehicle

ВСТУП

Актуальність теми.

Стрімкий розвиток автономних транспортних систем, поява концепцій підключених транспортних засобів (Connected Vehicles), розширення можливостей V2X-комунікацій та зростання рівня автоматизації дорожнього руху створюють нові передумови для підвищення безпеки та ефективності транспортних процесів. У таких умовах особливого значення набувають методи та програмні моделі, що забезпечують надійну, передбачувану та захищену взаємодію автономних транспортних сутностей у динамічному та потенційно небезпечному середовищі.

Незважаючи на значні досягнення у галузі автономного транспорту, забезпечення безпечної колективної поведінки транспортних засобів залишається складним науково-технічним завданням. Системи, що працюють у реальному часі, повинні коректно обробляти великі обсяги даних, враховувати багатофакторну динаміку дорожньої обстановки, прогнозувати розвиток ситуацій та гарантувати коректність виконання поведінкових сценаріїв. Додаткові виклики виникають при реалізації технологій платонування, кооперативного маневрування та обміну критично важливою інформацією між транспортними агентами.

Сучасні архітектурні моделі IoT та IoV створюють підґрунтя для організації взаємодії між транспортними засобами та інфраструктурою, проте їх використання потребує розроблення спеціалізованих алгоритмів відносного позиціонування, прогнозування та виявлення небезпечних ситуацій. Також актуальною є проблема формальної перевірки коректності поведінки автономних транспортних систем, що вимагає впровадження контрактних моделей, методів поведінкової верифікації та сервісно-орієнтованих архітектур із високими вимогами до кіберзахисту.

У цьому контексті дана магістерська робота спрямована на розроблення програмних моделей, алгоритмів та методологічних засад, які

забезпечують безпечну взаємодію автономних транспортних сутностей, підвищують рівень ситуаційної обізнаності та створюють можливість масштабованої і перевірюваної кооперації транспортних агентів у спільному середовищі.

Актуальність дослідження зумовлена зростаючою потребою у створенні безпечних, надійних та керованих автономних транспортних систем, які здатні функціонувати у багатофакторному середовищі дорожнього руху. Відповідно до глобальних тенденцій розвитку інтелектуальних транспортних систем, на перший план виходять питання забезпечення безпеки, прогнозованості поведінки та стійкості до відмов під час взаємодії автономних агентів.

Проблема ускладнюється тим, що автономні транспортні засоби повинні не лише самостійно приймати рішення, а й координувати власні дії з іншими учасниками руху, враховуючи їх відносне положення, швидкість, траєкторії, наміри та можливі ризики. Наявні підходи не завжди забезпечують необхідну точність визначення відносних параметрів та своєчасність виявлення небезпек, особливо у випадках високої щільності руху, різких змін ситуації або складних маневрів.

Крім того, формалізація поведінкових вимог до автономних систем, перевірка коректності їх роботи та гарантування відповідності стандартам безпеки й кіберзахисту потребують використання нових методологічних рішень. У цьому аспекті особливої ваги набувають контрактні моделі, формальні методи верифікації та фреймворки на кшталт Arrowhead, які забезпечують високий рівень структурованості та безпеки інформаційної взаємодії.

Таким чином, актуальність роботи підтверджується необхідністю створення комплексного підходу, який охоплює алгоритмічні, архітектурні та методологічні аспекти безпечної взаємодії автономних транспортних систем та враховує сучасні вимоги до масштабованості, надійності й кіберзахищеності.

Метою роботи є розроблення програмних моделей, алгоритмів та методології проєктування, які забезпечують безпечну та захищену взаємодію автономних транспортних сутностей.

Об'єктом дослідження є процес взаємодії автономних та підключених транспортних сутностей у динамічному середовищі дорожнього руху.

Предметом дослідження є методи, програмні моделі та алгоритми забезпечення безпечної кооперативної взаємодії автономних транспортних систем на основі V2V-комунікацій, відносного позиціонування та формальних моделей верифікації.

У ході виконання роботи поставлено такі завдання:

1. Проаналізувати предметну область безпечної взаємодії автономних транспортних систем та визначити ключові технологічні підходи і виклики.

2. Дослідити архітектурb IoT та IoV, методи платонування та концепції організації групової взаємодії транспортних засобів.

3. Розробити методи визначення відносного положення транспортних засобів на основі повідомлень безпеки.

4. Створити алгоритми прогнозування траєкторій та виявлення небезпечних ситуацій.

5. Сформувати методологію проєктування безпечної взаємодії автономних систем на основі контрактного аналізу та сервісно-орієнтованих архітектур.

У роботі застосовано такі методи:

- аналітичні методи для дослідження теоретичних моделей та алгоритмів автономних транспортних систем;

- математичне моделювання для опису відносного позиціонування та прогнозування траєкторій;

- алгоритмічні та програмні методи для реалізації V2V- і V2I-додатків;

- методи формальної верифікації для перевірки коректності контрактних моделей;

- методи системного аналізу для побудови комплексної методології проектування.

Наукова новизна отриманих результатів полягає у hозробленні алгоритмів відносного позиціонування та прогнозування руху, що інтегрують кооперативні повідомлення безпеки та кутові параметри відносної орієнтації транспортних засобів та сформуванні комплексної методології проєктування безпечної взаємодії автономних транспортних сутностей на основі контрактного підходу та сервісно-орієнтованих архітектур.

Практичне застосування результатів

Отримані результати можуть бути використані для:

- проєктування систем кооперативної взаємодії автономних транспортних засобів;

- створення програмних модулів для систем попередження небезпечних ситуацій у V2V- та V2I-комунікаціях;

- підвищення ефективності та безпеки технологій платонування;

- розроблення архітектур автономних транспортних систем із підтримкою верифікації поведінкових сценаріїв.

Структура магістерської роботи. Представлена робота складається зі вступу, трьох розділів та висновків. Загальний обсяг роботи становить 77 сторінок, і містить 27 рисунків, 7 таблиць, перелік використаних джерел із 41 позиції.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ БЕЗПЕЧНОЇ ВЗАЄМОДІЇ АВТОНОМНИХ ТРАНСПОРТНИХ СУТНОСТЕЙ

1.1. Особливості прийняття рішень водіями для зменшення аварійності

Впровадження систем допомоги водіям у процесі прийняття рішень (Driver Decision Support Systems, DDSS) має потенціал для зниження частоти дорожньо-транспортних пригод (ДТП). Технологічні рішення, що надають інформацію про безпечність виконання маневру, можуть ефективно запобігати летальним випадкам на дорогах.

Технології підключених транспортних засобів (Connected Vehicles, CV), реалізовані через комп'ютерний зір (Computer Vision, CV), дозволяють створювати додатки Vehicle-to-Vehicle (V2V) та Vehicle-to-Infrastructure (V2I). Ці додатки забезпечують інформування та попередження водіїв про потенційну небезпеку або інциденти.

В роботі [2] представлено проєкт який, залучив понад 1000 волонтерів для оцінки ефективності технології CV у додатках V2V та V2I. У рамках дослідження було згенеровано понад 84 000 попереджень V2V та V2I, з яких 388 були ідентифіковані як потенційно справжні конфлікти. Своєчасне попередження водіїв у цих критичних ситуаціях підтвердило підвищення безпеки. Крім того, результати дослідження вказують на покращення мобільності та безпеки як пасажирів, так і пішоходів. Ці статистичні дані свідчать про гостру необхідність розробки ефективної технології CV для значного підвищення безпеки дорожнього руху.

Для забезпечення точності попереджень одним із ключових викликів є точне визначення відносного положення сусідніх транспортних засобів. Існуючі підходи або використовують відеокамери, або припускають рух CV в одній смузі. Підходи на основі комп'ютерного зору демонструють зниження

продуктивності в несприятливих погодних умовах [12]. Крім того, припущення про рух в одній смузі є непрактичним у реальних умовах.

З огляду на це, актуальною є розробка V2V додатків попередження, які використовують виключно обмін базовими повідомленнями про безпеку (Basic Safety Messages, BSM) для точного прогнозування відносних позицій основного транспортного засобу (Host Vehicle, HV) та віддаленого транспортного засобу (Remote Vehicle, RV). Наскільки відомо, це перша робота, спрямована на обчислення відносного положення CV лише на основі даних BSM.

Існуючі методи CV для запобігання проїзду на червоне світло (Red Light Running, RLR) включають моделі прогнозування, розроблені на основі даних радарних датчиків, які не є даними CV і є нереалістичними для впровадження з фактичними CV. Інший підхід використовує технологію CV для виявлення RLV та попередження водія про порушення. У цьому методі червоний сигнал світлофора виявляється за допомогою встановленого на транспортному засобі датчика, на роботу якого можуть впливати змінні погодні умови. Крім того, існує невизначеність щодо того, чи виявлений червоний сигнал світлофора дійсно знаходиться на шляху наближення транспортного засобу. Це може призводити до хибних попереджень, негативно впливаючи на поведінку водія.

Це мотивувало необхідність розробки ефективного алгоритму виявлення світлофорів, який може бути використаний як основа для створення алгоритму виявлення RLV. Аналогічно, обмежена надійність існуючих методів прогнозування руху проти зустрічного потоку (Wrong-Way Driving, WWD) з використанням технології CV в усіх ситуаціях обумовила потребу в розробці алгоритму виявлення WWD, який був би ефективним у будь-якій заданій ситуації.

Для верифікації розроблених алгоритмів V2V та V2I було розширено симулятор CARLA для підтримки функціональності підключених транспортних засобів. Початкові експерименти проводилися з невеликою

кількістю транспортних засобів, а потім були здійснені широкомасштабні симуляції з використанням функції ко-симуляції SUMO, доступної в CARLA.

1.2. Технологія платонування на основі підключених транспортних засобів

1.2.1. Особливості технології платонування

Платонування транспортних засобів (Vehicle Platooning) — це технологія, що дозволяє групі транспортних засобів, зазвичай вантажівок або легкових автомобілів, рухатися разом в єдиному формуванні (колоні) на дуже малих і контрольованих відстанях між ними.

Ця система використовує технології зв'язку Vehicle-to-Vehicle (V2V) та автоматизованого керування (круїз-контроль, гальмування та рульове керування) для синхронізації швидкості та маневрів усіх транспортних засобів у колоні. Платонування вантажівок (truck platooning) — це форма, за якої певна кількість вантажівок рухається як автоколона (fleet) з малою міжтранспортною відстанню, використовуючи зв'язок V2V (Vehicle-to-Vehicle). На рисунку 1.1 показано архітектуру системи платонування вантажівок.

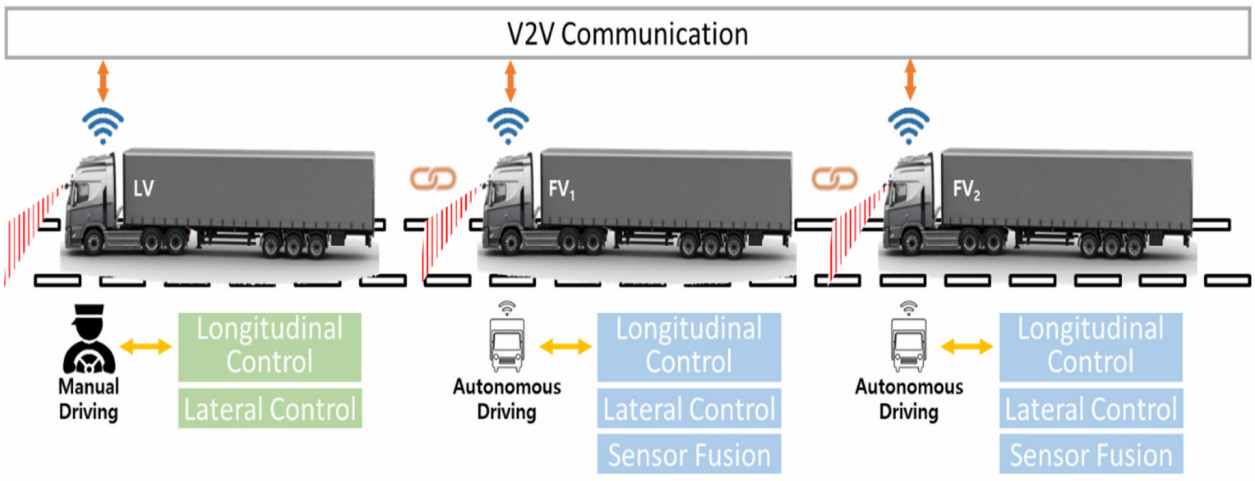


Рис. 1.1. Загальна архітектура системи платонування вантажівок

Вантажний транспортний засіб-лідер (Leading Vehicle, LV) керується вручну досвідченим водієм, а наступні транспортні засоби (Following Vehicles, FVs) рухаються за допомогою автономного керування. Наступний транспортний засіб (FV) використовує сенсори навколишнього середовища, такі як радар та камера, для сприйняття транспортних засобів та смуг попереду, а також здійснює автономне керування за допомогою поздовжнього та поперечного контролю руху.

Алгоритм автономного керування FV не покладається на GPS, оскільки транспортний засіб може не отримувати коректні GPS-сигнали в деяких умовах, наприклад, під час руху через тунелі.

У випадку платонування великих вантажних автомобілів довжина колони може легко сягати 100 м. Таким чином, кількість вантажівок в одному взводі зазвичай обмежується 3 або 4, з огляду на безпеку сусідніх транспортних засобів.

Ключові особливості та принципи:

1. Зменшення аеродинамічного опору. Це головна мета платонування. Рухаючись близько один до одного, транспортні засоби, що йдуть позаду (особливо другий та наступні), потрапляють у зону зниженого тиску, створену транспортним засобом-лідером. Це значно зменшує аеродинамічний опір (опір повітря), подібно до того, як птахи літають клином.

2. Підвищення паливної ефективності. Зниження аеродинамічного опору призводить до зменшення споживання палива (до 10–15% для вантажівок) та, відповідно, до скорочення викидів парникових газів.

3. Використання V2V та автоматизації:

- Перший транспортний засіб (лідер колони) управляється водієм або повністю автономно.

- Транспортні засоби, що йдуть позаду, автоматично прискорюються, гальмують і, у деяких системах, керують рухом, синхронізуючись із лідером

через V2V зв'язок. Це дозволяє підтримувати безпечні інтервали, які часто є меншими, ніж ті, що може підтримувати людина-водій.

4. Покращення пропускної здатності дороги. Зменшення інтервалів між транспортними засобами дозволяє розмістити більше автомобілів на одній ділянці дороги, підвищуючи пропускну здатність трас.

Платонування є ключовим додатком Інтернету транспортних засобів (IoV). Воно вимагає високонадійного та швидкого зв'язку, який забезпечується технологією CV. Система має бути здатною швидко реагувати на раптові зміни, передаючи інформацію про гальмування лідера до всіх наступних транспортних засобів практично миттєво. Платонування об'єднує автономне водіння (автоматичне керування швидкістю та дистанцією) та підключеність (обмін даними BSM). Платонування розглядається як важливий крок до повністю автономного та екологічно ефективного транспорту.

1.2.2. Архітектура Інтернету речей (IoT) у транспорті

Концепція Інтернету речей (IoT) визначається як обмін даними між фізичними об'єктами, підключеними через Інтернет. У зв'язку зі стрімким зростанням кількості підключених фізичних об'єктів, розроблено численні архітектури IoT. Згідно з [4], спрощена архітектура IoT включає шари краю, туману та хмари:

- Шар краю (Edge Layer) містить вузли краю (розумні об'єкти, датчики, виконавчі механізми). Тут здійснюється обчислення на краю (edge computing).

- Шар туману (Fog Layer) містить мережеві комунікаційні та обробні блоки (туманові вузли). Ці вузли забезпечують завантаження даних із вузлів краю до хмари (туманові обчислення).

- Хмарний шар (Cloud Layer). На цьому рівні створюються програмні додатки та аналітика з використанням даних, отриманих від туманових вузлів.

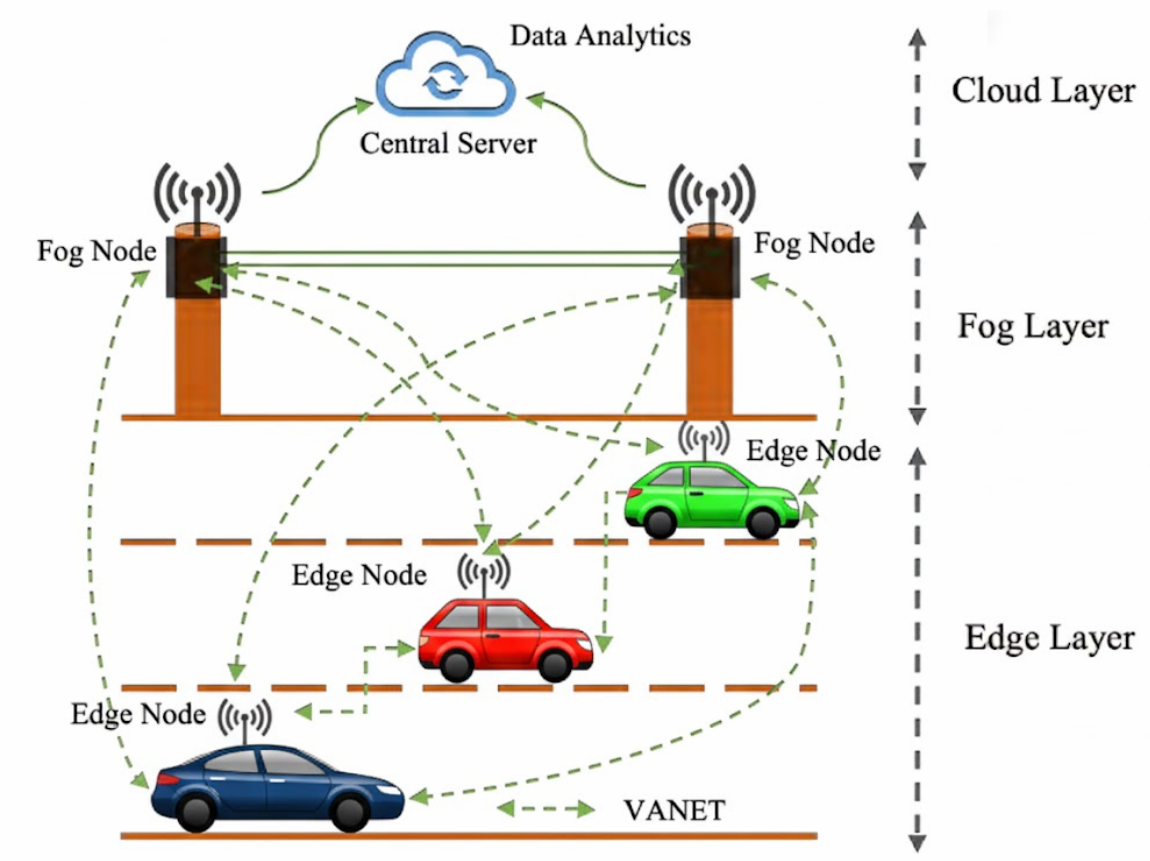


Рис. 1.2. Спрощена архітектура IoT, застосована до транспортних засобів, підключених через спеціальну мережу

Як проілюстровано на рис. 1.2, ця архітектура IoT може бути застосована до автономних транспортних засобів (Autonomous Vehicles, AV) у транспортному секторі, де кожен AV функціонує як вузол на краю для розв'язання різноманітних задач.

Аналіз експлуатаційних витрат [5] на вантажні перевезення показав незначне покращення паливної ефективності, що частково пояснюється використанням обмежувачів швидкості.

Таким чином, існує значний потенціал для зниження витрат на паливо та викидів парникових газів через подальше підвищення паливної економічності. Платонування транспортних засобів (Vehicle Platooning) — це стратегія, за якої група транспортних засобів рухається на мінімальних безпечних відстанях один від одного, що дозволяє досягти зниження аеродинамічного опору і, як наслідок, покращити паливну ефективність [20].

1.2.3. Інтернет транспортних засобів (IoV) та платонування

Технології автономного водіння активно розробляються провідними компаніями (Tesla, Waymo, Uber та ін.), а технологія CV тестується багатьма установами [7]. Для підвищення ефективності та безпеки транспорту ці дві технології інтегруються. Платонування є одним з ключових транспортних додатків, що розробляються на цій основі.

Платонування реалізується через Інтернет транспортних засобів (Internet of Vehicles, IoV), який є підмножиною IoT. У рамках IoV транспортні засоби та дорожні системи з'єднані через спеціальні самоорганізовані мережі транспортних засобів (Vehicular Ad Hoc Networks, VANET) [3].

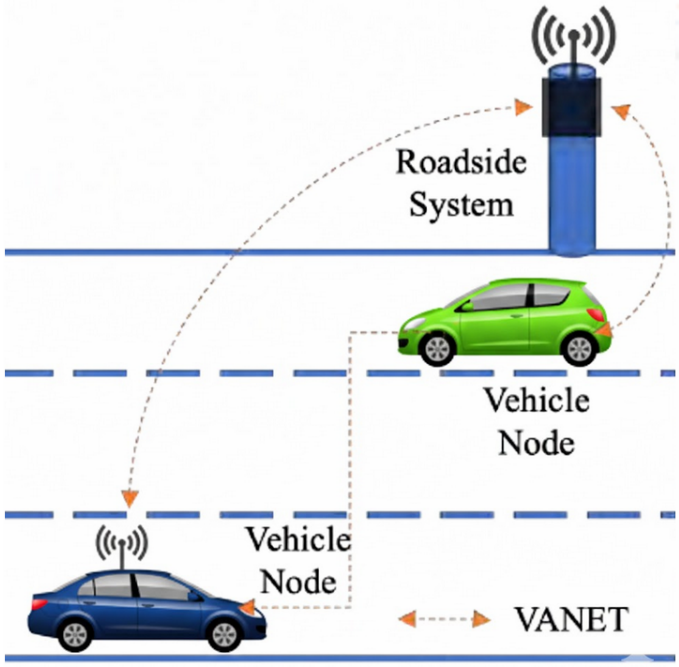


Рис. 1.3. Транспортні засоби та дорожня система, підключені через VANET

Кожен транспортний засіб в IoV розглядається як розумний об'єкт, оснащений сенсорними платформами, обробними пристроями та блоками управління, що забезпечує зв'язок Vehicle-to-Everything (V2X). Очікується, що ця технологія задовольнить сучасні потреби транспорту шляхом

покращення безпеки, зменшення заторів, зниження споживання палива та забруднення, а також уможливить послуги спільного використання автомобілів. Крім того, кожен транспортний засіб може функціонувати як інформаційний вузол (хаб) або крайовий сервер для людей та пристроїв IoT на дорозі. Ця технологія має значний вплив на широкий спектр галузей, включаючи транспорт, автомобільне виробництво, енергетику, автоматизацію та інформаційно-комунікаційні технології.

1.3. Архітектура, виклики та симуляційне моделювання технології автономних та підключених транспортних засобів

1.3.1. Архітектура та обмеження автономних транспортних засобів

Автономний транспортний засіб (Autonomous Vehicle, AV), або безпілотний автомобіль, являє собою інтелектуальну систему, що використовує технологію автоматизованої системи водіння (Automated Driving System, ADS). Ця система призначена для сприйняття навколишнього середовища за допомогою сенсорів (зокрема, LIDAR, камер), вибору оптимальної траєкторії до заданого пункту призначення та подальшого самостійного руху.

Впровадження AV має потенціал для зменшення рівня викидів, запобігання дорожньо-транспортним пригодам (ДТП) та зниження психофізіологічного напруження водія.

Типова архітектура ADS складається з кількох ключових модулів:

- Сприйняття (Perception). Інтерпретує зовнішнє середовище на основі даних, отриманих від сенсорних систем.

- Планування руху (Motion Planning). Виконує низькорівневі операції для досягнення цілей, визначених на вищому рівні.

- Навігація (Navigation). Визначає поточне та прогнозує майбутнє положення транспортного засобу (локалізація) у дорожньому середовищі.

- Поведінка (Behavior). Приймає рішення та генерує необхідні команди для виконавчих механізмів (дросельна заслінка, гальмо, кермо) після оцінки навколишнього середовища та завершення планування руху й навігації [28].

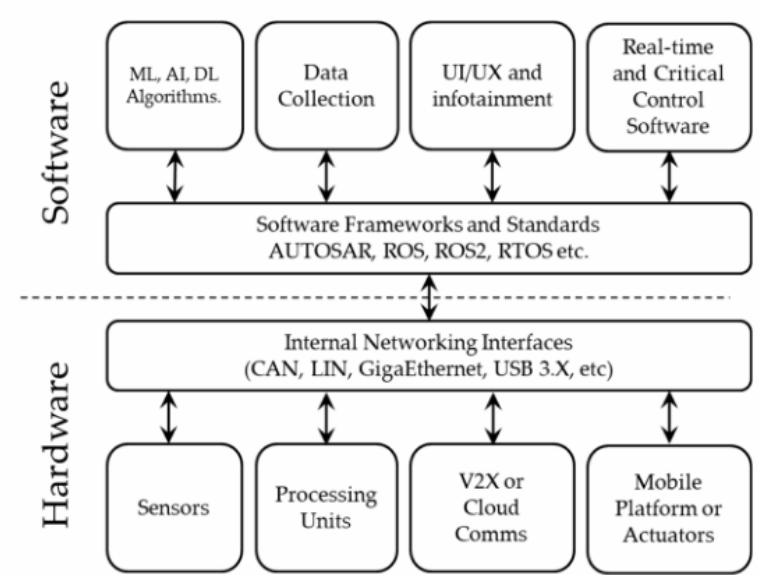


Рис. 1.4. Архітектура системи автономного керування з точки зору технічної перспективи, що описує основні апаратні та програмні компоненти та їх реалізацію

З моменту першого представлення цієї технології досягнуто значного прогресу в розвитку ADS. Товариство автомобільних інженерів (SAE) класифікує автономність за шістьма рівнями (від Рівня 0 – відсутність автоматизації, до Рівня 5 – повна автономність) [9]. Проте, наразі на дорогах відсутні транспортні засоби з повним п'ятим рівнем автономності.

Поточна ADS функціонує як автономна система, призначена виключно для основного транспортного засобу (Host Vehicle), виконуючи всі операції ADS в межах його сенсорного поля. Незважаючи на значний прогрес, залишається низка відкритих проблем.

Чинники, такі як складні дорожні та транспортні умови, перешкоди для радарних систем, проблеми на пішохідних переходах, програмні помилки та атаки вторгнення, створюють різноманітні технічні, етичні та юридичні виклики.

SAE Level 0	SAE Level 1	SAE Level 2	SAE Level 3	SAE Level 4	SAE Level 5
NO AUTOMATION	**DRIVER ASSISTANCE**	**PARTIAL AUTOMATION**	**CONDITIONAL AUTOMATION**	**HIGH AUTOMATION**	**FULL AUTOMATION**
The human driver performs all driving aspects of driving tasks, e.g., steering, acceleration, etc.	The vehicle features a single automated system for driver assistance, such as steering or acceleration/deceleration and with the anticipation that the human driver performs all remaining aspects of the driving tasks.	ADAS. The vehicle can perform steering and acceleration/deceleration. However, the human driver is required to monitor the driving environment and can take control at any time.	The vehicle can detect obstacles in the driving environment and can perform most driving tasks. Though, human override is still required.	The vehicle can perform all aspects of the dynamic driving task under specific scenarios. Geofencing is required. Human override is still an option.	The vehicle performs all driving tasks under all conditions and scenarios without human intervention.
The human drivers monitor the driving environment			The automated system monitors the driving environment		

Рис. 1.5. Загальний огляд різних рівнів автоматизації керування

1.3.2. Технології підключених автономних транспортних засобів

Обмеження AV можуть бути подолані шляхом інтеграції з підключеними системами, де AV отримують можливість покладатися на інші транспортні засоби (ТЗ) або дорожню інфраструктуру для виконання функцій ADS (рис. 1.6).

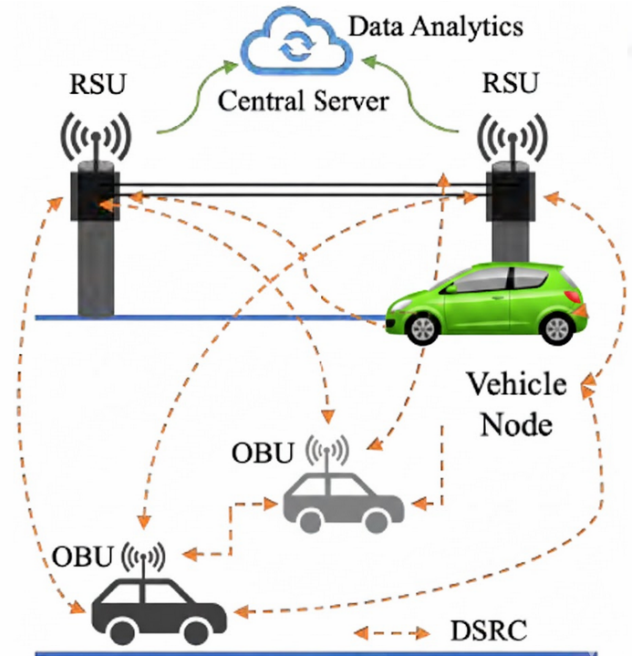


Рис. 1.6. Транспортні засоби та інфраструктура, підключені за допомогою технології DSRC

Ці транспортні засоби можуть отримувати великі обсяги даних через комунікаційні канали V2V (Vehicle-to-Vehicle), V2I (Vehicle-to-Infrastructure) та V2X (Vehicle-to-Everything). Використання цих даних дозволяє AV компенсувати недоліки їхньої поточної архітектури, орієнтованої виключно на бортові системи.

Комунікації V2V, V2I та V2X є основою технології підключених транспортних засобів (Connected Vehicles, CV). Термін CV описує інтелектуальну транспортну систему (ITS), яка включає зв'язок між ТЗ та дорожньою інфраструктурою через різні бездротові технології, такі як DSRC (Dedicated Short Range Communications) та Long Term Evolution (LTE).

Основні компоненти технології CV:

- Дорожні пристрої (Roadside Units, RSU) - встановлюються на дорожній інфраструктурі.

- Бортові пристрої (On-Board Units, OBU) - встановлюються всередині ТЗ.

OBU збирають дані про ТЗ (швидкість, місце розташування, напрямок) і передають цю інформацію сусіднім OBU або RSU у вигляді базових повідомлень про безпеку (Basic Safety Messages, BSM) з високою частотою. Отримані BSM використовуються для забезпечення безпеки водіїв та попередження у разі потенційного ризику зіткнення.

Визначено понад 40 додатків CV, класифікованих за трьома основними категоріями: безпека, мобільність та навколишнє середовище [9]. Отримуючи доступ до великої кількості локальних даних про трафік, технологія CV трансформує AV у підключений автономний транспортний засіб (Connected Autonomous Vehicle, CAV), значно підвищуючи безпеку, ефективність та мобільність транспортної системи [32].

1.3.3. Потреба в методології побудови фреймворку для симуляції

Хоча CAV вважається майбутнім автономного водіння, наразі на дорогах відсутні працюючі CAV. Розробка, впровадження та верифікація цієї

технології вимагає значних інвестицій від виробників оригінального обладнання (OEM).

Однак, прискорення розвитку може бути досягнуто за нижчої вартості за допомогою симуляційного моделювання. Інструменти візуальної 3D-симуляції дозволяють:

- Верифікувати алгоритми розробниками.
- Тестувати небезпечні критичні випадки дослідниками.
- Знижувати високі витрати для OEM.
- Прискорювати розв'язання проблем.

Отже, існує потреба в 3D-симуляторі CAV. Це мотивувало вибір популярного 3D-симулятора CARLA для забезпечення функціональності підключеності між транспортними засобами.

Ця робота досліджує програмний фреймворк для симуляції, який може використовуватися дослідниками для прискорення розвитку технологій CV та CAV. Наразі не існує симулятора, який одночасно підтримує моделювання AV, CV та CAV. Цей фреймворк є симулятором, що дозволяє симулювати підключеність V2V та V2I як для транспортних засобів з ручним керуванням, так і для повністю автономних ТЗ. Фреймворк надає можливість симулювати широкий спектр сценаріїв V2V та V2I та може бути адаптований користувачами для власних дослідницьких цілей.

Висновки до розділу

У першому розділі проведено системний аналіз предметної області безпечної взаємодії автономних транспортних сутностей, що дозволило окреслити ключові технологічні тренди та виклики у сфері автономного транспорту. Досліджено особливості прийняття рішень водіями під час уникнення аварійних ситуацій, що слугувало теоретичною основою для розуміння логіки функціонування автоматизованих систем підтримки водія

(ADAS) та підходів до формування поведінкових моделей автономних транспортних агентів.

У роботі встановлено, що технологія платонування, яка базується на взаємодії підключених транспортних засобів, потребує комплексної архітектури з підтримкою високої пропускної здатності, низької затримки та стійкості до збоїв. Проведений аналіз IoT- і IoV-архітектур показав, що вони відіграють визначальну роль у побудові ефективних систем групової взаємодії транспортних сутностей, забезпечуючи постійний обмін даними, спільне прийняття рішень та адаптивну координацію руху.

РОЗДІЛ 2. МЕТОДИ, АЛГОРИТМИ ТА КОНЦЕПЦІЇ БЕЗПЕЧНОЇ ВЗАЄМОДІЇ СУЧАСНИХ АВТОНОМНИХ ТРАНСПОРТНИХ СИСТЕМ

2.1. Огляд концептуальних основ сучасних транспортних систем

У цьому розділі представлено контекстуальний огляд технологій підключених транспортних засобів (CV), автоматизованих систем водіння (ADS), їхньої інтеграції в підключені автономні транспортні засоби (CAV) та концепції платонування.

Термін Connected Vehicles (CV) стосується систем інтелектуального транспорту, які забезпечують зв'язок між транспортними засобами та навколишнім середовищем. Основним каналом комунікації є Dedicated Short-Range Communications (DSRC). CV-технологія охоплює три основні типи зв'язку: Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I) та Vehicle-to-Everything (V2X).

Архітектура CV складається з двох ключових компонентів:

1. Бортові пристрої (On-Board Units, OBU) - встановлюються всередині транспортних засобів.

2. Дорожні пристрої (Roadside Units, RSU) - розміщуються на елементах дорожньої інфраструктури (наприклад, на перехрестях).

OBU збирають дані про ТЗ (швидкість, місце розташування, напрямок руху) і з високою частотою передають цю інформацію сусіднім OBU або RSU у формі базових повідомлень про безпеку (Basic Safety Messages, BSM) через DSRC. Отримані BSM використовуються для підвищення безпеки водіїв та попередження про потенційний ризик зіткнення [14].

Ключові V2V додатки, продемонстровані в контексті цієї роботи:

- Попередження про лобове зіткнення (Forward Collision Warning, FCW) - інформує водія про раптову зупинку транспортного засобу, що рухається попереду.

- Електронний стоп-сигнал (Emergency Electronic Brake Light, EEBL) - Попереджає водія про різке гальмування транспортного засобу, яке знаходиться попереду основного ТЗ, але закрите для прямого огляду іншими об'єктами.

- Попередження про сліпу зону (Blind Spot Warning, BSW) - попереджає водія про присутність іншого ТЗ у сліпій зоні під час спроби зміни смуги руху.

- Попередження про повільно рухомий транспортний засіб (Slow Moving Vehicle Warning, SMVW) - повідомляє водіїв про наближення до повільно рухомого ТЗ у тому ж напрямку.

- Попередження "Не обганяти" (Do Not Pass Warning, DNPW) - попереджає водія, якщо смуга, необхідна для обгону повільнішого ТЗ, зайнята зустрічним ТЗ.

- Попередження про нерухомий транспортний засіб (Stationary Vehicle Warning, SVW) - повідомляє водія про наближення до нерухомого ТЗ попереду на дорозі.

Відомо, що автономний транспортний засіб (AV) — це інтелектуальний ТЗ, що використовує автоматизовану систему водіння (ADS) для сприйняття навколишнього середовища, вибору оптимального маршруту та самостійного виконання руху.

Технологія AV продовжує стикатися з низкою невирішених проблем. Автономність на рівні 4 (рис. 1.5) і вище залишається складною відкритою проблемою. Більшість поточних рішень ADS базуються на підходах, орієнтованих виключно на бортові системи (лише для основного ТЗ) [12].

Подолання обмежень бортового підходу AV може бути досягнуто шляхом інтеграції з технологією CV. Це дозволяє AV покладатися на дорожню інфраструктуру та сусідні ТЗ для виконання функцій ADS. Такі ТЗ, відомі як підключені автономні транспортні засоби (CAV), отримують доступ до великих обсягів даних через комунікації V2V, V2I та V2X. Ця

стратегія розглядається як майбутнє автономного водіння та механізм для усунення недоліків існуючої архітектури лише основного ТЗ.

Мережа CV загалом класифікується як самоорганізована мережа (ad hoc network) і відома як Vehicular Ad Hoc Network (VANET). На комунікаційному рівні VANET найчастіше використовуються технології DSRC та Long Term Evolution (LTE).

У архітектурі Інтернету речей (IoT) OBU репрезентує шар краю, а RSU — туманний вузол (як показано на рисунку 1.1). Інформація, що передається через BSM, використовується для попередження водіїв про потенційну аварійну ситуацію

Інтеграція технології CV в AV формує CAV, що покращує безпеку, ефективність та мобільність транспортної системи за рахунок доступу до великого обсягу локальних даних про трафік. CAV можуть отримувати значні обсяги даних про сусідні ТЗ для розв'язання проблем, пов'язаних із дизайном, орієнтованим лише на основний ТЗ. Наразі працюючі CAV відсутні на дорогах, але цей дизайн є прогнозованою траєкторією розвитку ADS [12].

Платонування визначається як "спонтанне та динамічне створення конвоїв транспортних засобів". Воно формується щонайменше двома ТЗ, які рухаються на невеликій відстані один за одним. Керування може бути повністю автономним, або водії отримують допомогу для підтримки безпечної дистанції та руху в межах смуги (включно з автономним гальмуванням).

Стандартна конфігурація платонування включає Platoon Leader (PL), який керує, та учасників (Platoon Members, PM), які слідують за ним, використовуючи VANET. PL відповідає за регулювання швидкості, кількості учасників та дозволів на в'їзд/виїзд. PM виконують вказівки PL.

Зв'язок між PL і PM здійснюється через V2V. Міжтранспортний проміжок розраховується та підтримується за допомогою адаптивного круїз-контролю (Adaptive Cruise Control, ACC), який використовує сенсори

(LIDAR, радар, камери). Кооперативний ACC (Cooperative ACC, CACC), що поєднує комунікацію V2V (CV) з ACC (ADS), є необхідним для реалізації платонування.

2.2. Методологія визначення відносного положення підключених транспортних засобів на основі повідомлень безпеки

Основна мета пропонованого підходу полягає у визначенні відносного положення сусідніх підключених транспортних засобів (Connected Vehicles, CV), використовуючи виключно дані, що містяться у базових повідомленнях безпеки (Basic Safety Messages, BSM).

У рамках цієї роботи прийнято припущення, що інформація про географічне розташування транспортних засобів отримується із системи диференціального GPS (DGPS). Дослідження [17] підтверджують, що точність DGPS становить у середньому 33 см, а максимальна похибка не перевищує 70 см.

2.2.1. Обчислення відносного кута

Для прогнозування відносного положення віддаленого транспортного засобу (Remote Vehicle, RV) відносно основного транспортного засобу (Host Vehicle, HV) критичним кроком є обчислення відносного кута θ.

Нехай основний транспортний засіб (HV) розташований у точці $A=(x_1, y_1, z_1)$ і рухається в напрямку H_1, а віддалений транспортний засіб (RV) розташований у точці $B=(x_2, y_2, z_2)$.

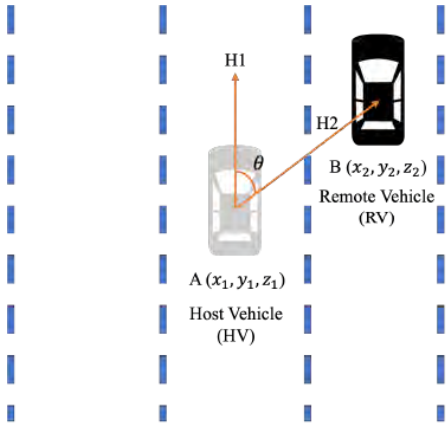
1. Спочатку обчислюється вектор AB між двома точками:

$$AB = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$$

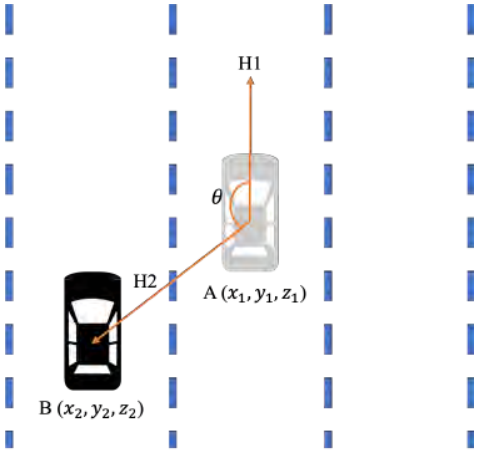
2. Визначається напрямок H_2 вектора AB відносно осі Y (або фіксованої системи координат).

3. Відносний кут θ між напрямком руху HV (H_1) та напрямком на RV (H_2) обчислюється як різниця між цими напрямками:

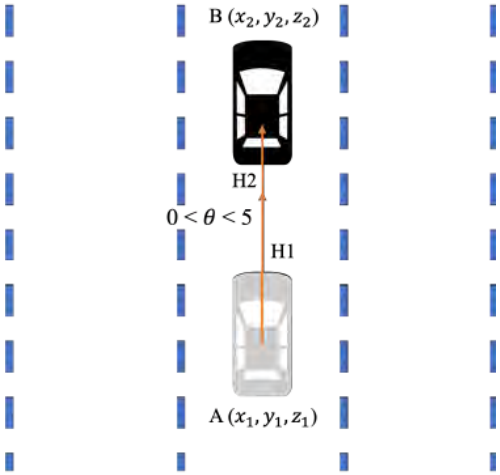
$$\theta = H_2 - H_1$$



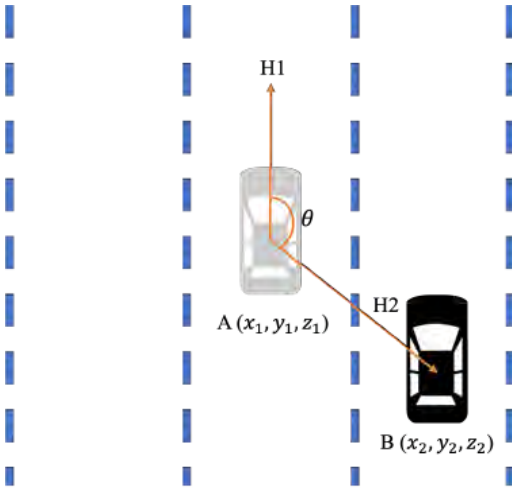
а) Загальна концепція розрахунку θ



б) Віддалений ТЗ знаходиться у лівій смузі



в) Віддалений ТЗ знаходиться у тій же смузі



г) Віддалений ТЗ знаходиться у правій смузі

Рис. 2.1. Ілюстрації відносного кута θ із віддаленим транспортним засобом у різних смугах

2.2.2. Прогнозування відносного положення

Використовуючи отриманий відносний кут θ, стає можливим передбачити відносне положення RV відносно HV у реальному часі.

На рисунку 2.1 а представлено загальну концепцію розрахунку θ. Рисунки 2.1 б – г ілюструють відносний кут θ для різних сценаріїв розташування RV відносно HV.

Ці сценарії демонструють, як кут θ дозволяє однозначно ідентифікувати латеральне (поперечне) розташування RV.

2.3. Реалізація алгоритмів попередження на основі зв'язків типу V2V та відносного кута

У цьому розділі описується, як обчислений відносний кут θ між основним транспортним засобом (Host Vehicle, HV) та віддаленим транспортним засобом (Remote Vehicle, RV) використовується для реалізації трьох ключових додатків V2V: попередження про лобове зіткнення, електронний стоп-сигнал та попередження про сліпу зону, а також

додаткових попереджень про повільно рухомий та нерухомий транспортні засоби.

2.3.1. Алгоритм попередження про лобове зіткнення

Логіка попередження про лобове зіткнення FCW оцінюється на основі відстані (d) та відносного кута (θ) між HV та RV.

Перевіряється, чи знаходиться RV безпосередньо попереду HV, що визначається двома критеріями:

- $|\theta| \in [0^\circ, 5^\circ]$. Ця умова задовольняється лише у сценарії, коли RV рухається у тій самій смузі перед HV (як проілюстровано на рисунку 2.1 в).

- d<45 м.

Якщо просторова умова задовольняється, обчислюється час до зіткнення (Time-to-Collision, TTC), використовуючи алгоритми відстані та часу.

Якщо обчислений TTC менше встановленого еталонного часу безпеки, водієві HV видається попередження FCW.

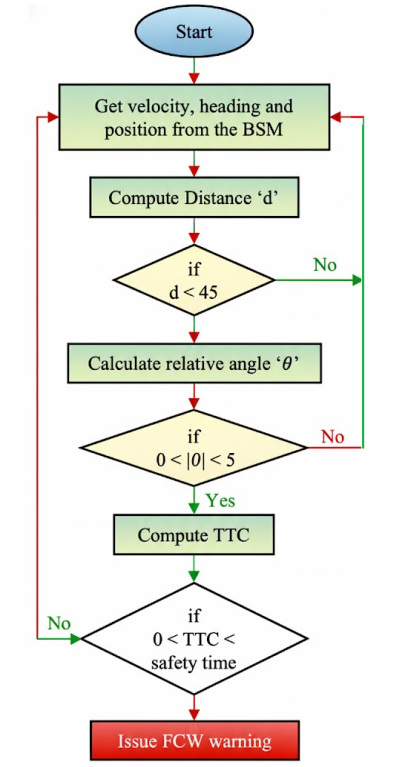


Рис. 2.2. Блок-схема логіки видачі попередження про лобове зіткнення

2.2.2. Алгоритм видачі повідомлення про електронний стоп-сигнал

Логіка EEBL, як і FCW, вимагає, щоб RV знаходився безпосередньо попереду HV та на невеликій відстані.

Використовуються ті ж самі критерії, що й для FCW:

- $|\theta| \in [0^\circ, 5^\circ]$.

- d<45 м.

Якщо просторова умова задовольняється, прискорення RV порівнюється з еталонним значенням прискорення.

Якщо прискорення RV перевищує еталонне значення (що вказує на швидке уповільнення), водієві HV видається попередження EEBL.

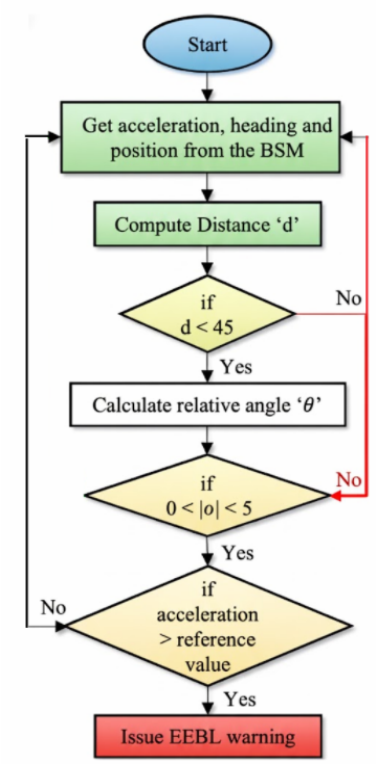


Рис. 2.3. Блок-схема логіки видачі попередження електронний стоп-сигнал

2.3.3. Алгоритм попередження про сліпу зону (BSW)

Для оцінки логіки BSW використовується θ для визначення, чи знаходиться RV діагонально поруч із HV у потенційній сліпій зоні.

Визначається, чи знаходиться RV у сліпій зоні HV за критеріями:

- $|\theta| \in [80^\circ, 145^\circ]$. Це відповідає сценаріям, де RV знаходиться у сусідній лівій або правій смузі відносно HV (як проілюстровано на рисунках 2.1 б та 2.1 г).

- $d<25$ м.

Якщо обидві умови задовольняються, водієві HV видається попередження BSW, інформуючи про наявність транспортного засобу в сліпій зоні під час спроби зміни смуги.

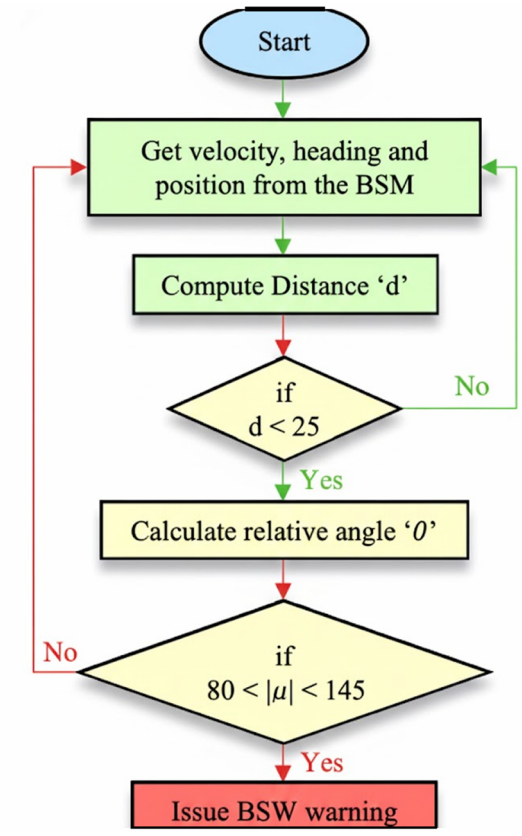


Рис. 2.4. Алгоритм видачі повідомлення про сліпу зону

2.3.4. Алгоритм попередження про повільно рухомий транспортний засіб (SMVW)

Для реалізації SMVW припускається, що обмеження швидкості для поточної дороги є відомим.

Обчислюється різниця швидкостей Δv між обмеженням швидкості дороги та швидкістю RV (отриманою з BSM).

Якщо d<45 м та Δv>10 км/год (значення Δv є довільним і може бути змінене), то виконується перевірка напрямку.

Для перевірки напрямку та смуги:

- Обчислюється різниця кутів напрямку α=H1−H3 між HV та RV.

- Якщо ∣α∣<5◦ (що означає рух HV та RV в одному напрямку), обчислюється відносний кут θ=H2−H1.

- Якщо ∣θ∣<1◦ (що означає, що HV та RV знаходяться в одній смузі), водієві HV видається попередження SMVW.

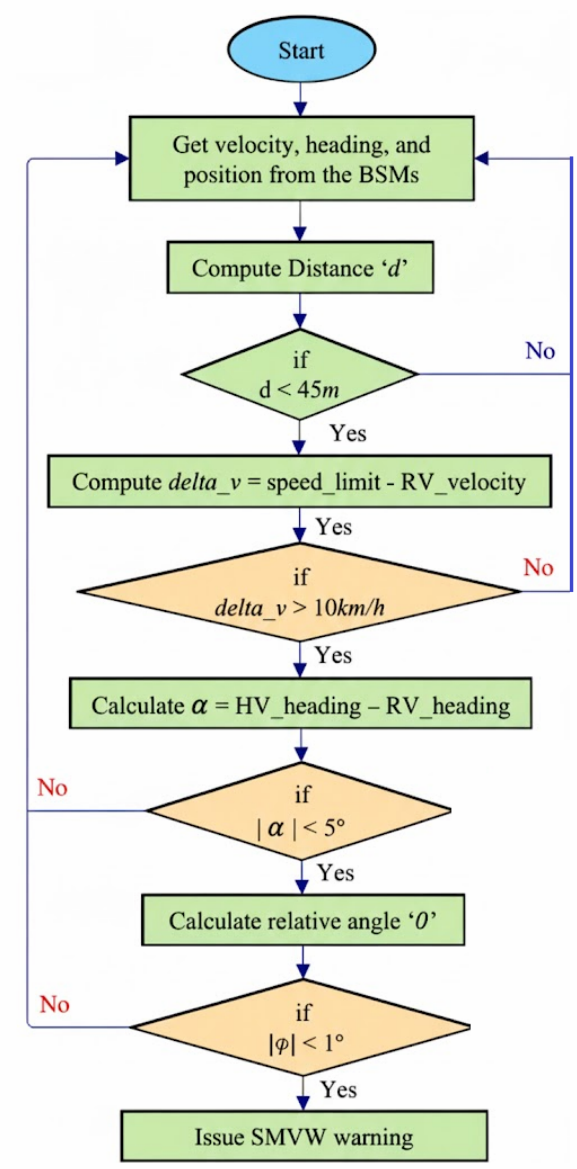


Рис. 2.5. Алгоритм попередження про повільно рухомий транспортний засіб

2.3.5. Алгоритм видачі попередження "Не обганяти" (DNPW)

Алгоритм DNPW ґрунтується на логіці SMVW та додатково перевіряє наявність зустрічного RV.

Спочатку перевіряється наявність повільно рухомого ТЗ у напрямку HV за логікою SMVW (рисунок 2.6). Якщо попереду виявлено повільно рухомий RV, перевіряється, чи рухається будь-який інший RV у протилежному напрямку шляхом обчислення різниці кутів напрямку α=H1 −H3.

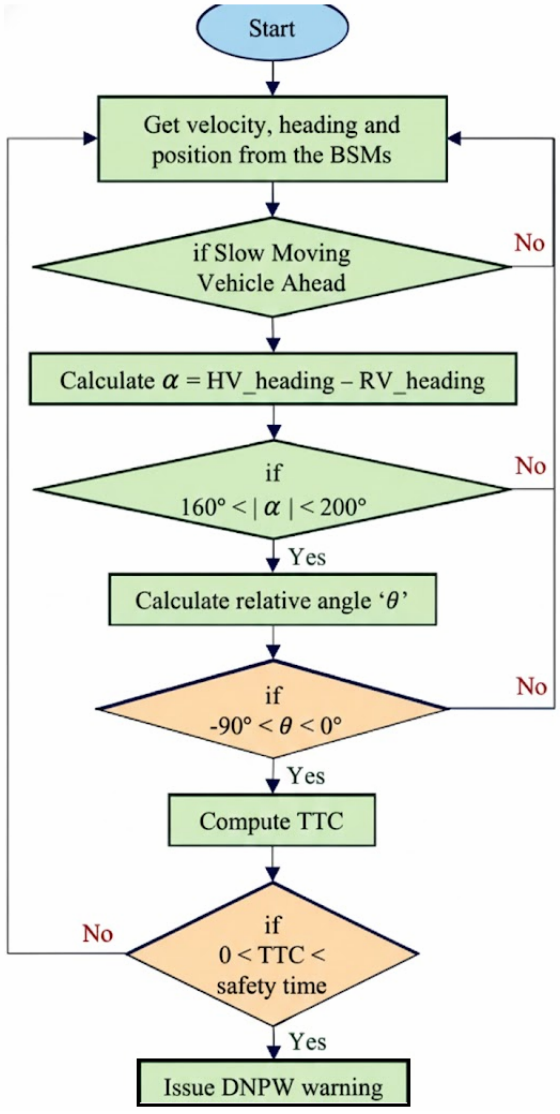


Рис. 2.6. Алгоритм видачі попередження "Не обганяти"

Якщо $|\alpha| \in [160^\circ, 200^\circ]$ (що вказує на рух у протилежних напрямках), обчислюється відносний кут θ=H2−H1.

Потім виконується просторова перевірка та TTC:

Якщо $\theta \in [-90^\circ, 0^\circ]$ (що означає, що зустрічний RV наближається до HV спереду), обчислюється TTC.

Якщо TTC менше еталонного часу безпеки, водієві HV видається попередження DNPW.

2.3.6. Попередження про нерухомий транспортний засіб (SVW)

SVW вимагає, щоб кожен CV незалежно визначав свій нерухомий статус і передавав цю інформацію через BSM.

Логіка визначення нерухомості наступна. Кожен CV використовує лічильник stationary_counter та змінну is_stationary. Якщо швидкість ТЗ дорівнює нулю, stationary_counter збільшується. Якщо швидкість ненульова, лічильник скидається. Якщо stationary_counter перевищує встановлений поріг (наприклад, 60 секунд, значення є довільним і залежить від частоти кадрів симуляції), змінна is_stationary набуває значення істина (true), і повідомлення про нерухомість передається іншим CV у межах діапазону DSRC.

Логіка генерації попередження SVW:

1. HV використовує повідомлення про нерухомість від RV для генерації SVW.

2. Просторова умова: обчислюється відстань d від нерухомого RV. Якщо $d<80$ м.

3. Перевірка напрямку руху: перевіряється, чи знаходиться RV на шляху руху HV, обчислюючи різницю кутів напрямку $\alpha = H1 - H3$ між HV та RV.

Якщо $|\alpha| < 5^\circ$ (що вказує на те, що RV знаходиться у напрямку руху HV), обчислюється відносний кут θ.

4. Визначення смуги: перевіряється модуль θ:

Якщо $|\theta| \in [0^\circ, 1^\circ]$ (HV та RV в одній смузі), видається SVW "в тій самій смузі".

Якщо ｜θ｜>1◦ (але в межах допустимого діапазону 5◦, що вказує на RV у сусідній смузі), видається SVW "в іншій смузі".

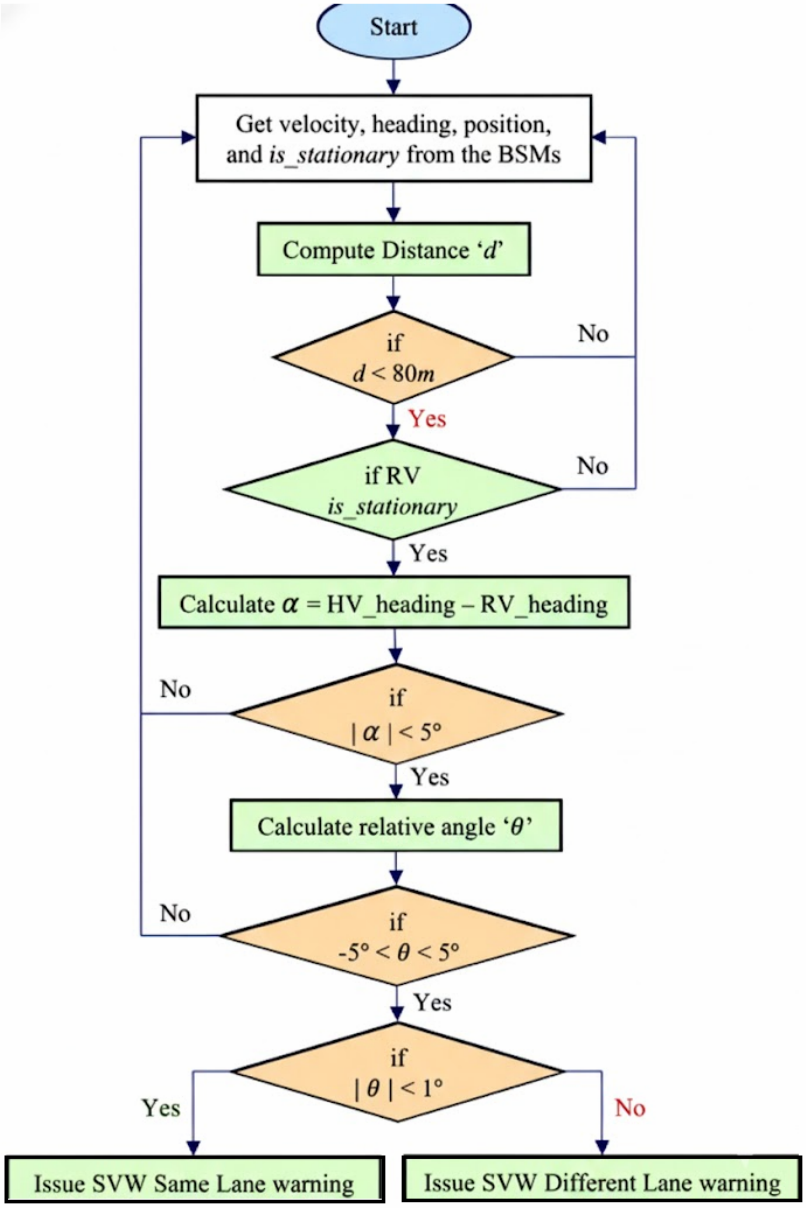


Рис. 2.7. Алгоритм попередження про нерухомий транспортний засіб

2.4. Розробка алгоритмів V2I додатків на основі відносного позиціонування

У цьому розділі представлена методологія розробки додатків Vehicle-to-Infrastructure (V2I) для підвищення безпеки дорожнього руху. Спочатку пропонується алгоритм для точного визначення поточного стану світлофора.

Ця логіка потім використовується для реалізації попередження про порушення червоних сигналів світлофора (Red Light Violation Warning, RLVW). Нарешті, представлено алгоритм для виявлення руху в неправильному напрямку (Wrong-Way Driving, WWD), який активує попередження Wrong-Way Entry Warning (WWEW). Всі алгоритми розроблені з використанням відносного кута θ між основним транспортним засобом (HV) та пристроями дорожньої інфраструктури (Roadside Units, RSU), обчисленого на основі підходу відносного положення.

2.4.1. Алгоритм виявлення світлофора

Для реалізації цього алгоритму припускається, що чотири RSU встановлені на кожному світлофорі перехрестя (точки $B(x_2,y_2,z_2)$, $C(x_3,y_3,z_3)$, $D(x_4,y_4,z_4)$ та $E(x_5,y_5,z_5)$), як показано на рисунку 2.8.

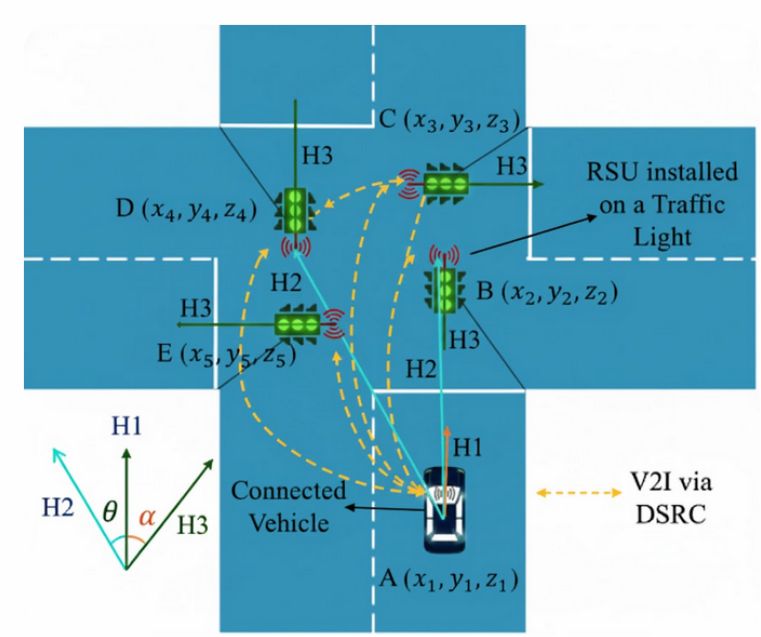


Рис. 2.8. Методика виявлення світлофора

RSU передають статус світлофора, до якого вони приєднані, разом із даними про розташування та напрямок руху (heading) сусіднім CV за допомогою зв'язку V2I. Напрямок RSU (H3) калібрується так, щоб він був протилежним до законного потоку руху.

Обчислення відносних параметрів:

- HV знаходиться у точці $A(x_1,y_1,z_1)$ з кутом напрямку H_1.

- Вектори AB, AC, AD, AE мають кут напрямку H_2.

- Відносний кут θ: різниця між кутами напрямку HV та вектором AB ($\theta=H_1 - H_2$).

- Різниця кутів напрямку α: різниця між кутами напрямку HV та RSU ($\alpha=H_1-H_3$).

Алгоритм виявлення стану світлофора подано в лістингу 2.1.

Лістинг 2.1. Алгоритм виявлення стану світлофора

```
function DETECT_TRAFFIC_LIGHT_STATE()
    // Обробка вхідних даних
    Process HV and RSU BSMs

    // Обчислення просторових параметрів
    Compute distance 'd' between HV and RSU locations
    Calculate relative angle 'θ'
    α = RSU_heading - HV_heading

    // Крок 1: Ідентифікація напрямку (перевірка RSU, що контролює смугу руху HV)
    if 135° < |α| < 180° then

        // Крок 2: Перевірка положення (чи знаходиться RSU попереду і в межах діапазону
        if 0° < |θ| < 45° and d < 100 then

            // Крок 3: Визначення стану світлофора та дія
            if RSU_light == Green then
                Traffic light ahead is green
                Check_Red_Light_Violation() // Виклик функції перевірки порушення

            else if RSU_light == Yellow then
                Traffic light ahead is yellow

            else if RSU_light == Red then
                Traffic light ahead is red
            end if
        end if
    end if
end function
```

2.4.2. Алгоритм попередження про порушення червоних сигналів світлофора

Логіка попередження про порушення червоних сигналів світлофора (RLVW) активується після того, як світлофор попереду ідентифіковано як зелений.

Обчислення фактичної відстані до перехрестя:

- Відстань d між CV та RSU не є фактичною відстанню до стоп-лінії.

- Зміщення перехрестя (intr_offset): відстань між стоп-лінією та RSU (світлофором).

- Фактична зупинна відстань (dintr): $dintr = d - intr\_off_{set}$.

Лістинг 2.2. Алгоритм попередження про порушення червоних сигналів світлофора

```
function CHECK_RED_LIGHT_VIOLATION()
    // Обчислення фактичної відстані до перехрестя (стоп-лінії)
    d_intr = d - intr_offset

    // Отримання поточної швидкості HV
    speed_cur = HV_speed

    // Отримання залишкового зеленого часу світлофора з BSM RSU
    Get remaining_green_time from RSU BSM

    // Обчислення часу, необхідного для досягнення перехрестя
    time_intr = d_intr / speed_cur

    // Перевірка умови порушення
    if remaining_green_time < time_intr then
        // Якщо зелений час закінчиться раніше, ніж ТЗ досягне перехрестя
        Issue RLVWarning // Видати попередження про порушення червоного світла
    end if
end function
```

Алгоритм RLVW (Red Light Violation Warning) призначений для попередження водія про високий ризик того, що світлофор зміниться на червоний до того, як транспортний засіб (ТЗ) зможе безпечно проїхати перехрестя. Він активується, коли система виявляє, що світлофор попереду горить зеленим (або жовтим).

Основна логіка алгоритму базується на порівнянні залишкового зеленого часу світлофора з часом, необхідним ТЗ для досягнення стоп-лінії.

Після визначення фактичної зупинної відстані обчислюється час, необхідний ТЗ, щоб досягти стоп-лінії, на основі поточної швидкості ТЗ (speedcur):

$$time_{\text{intr}} = d_{\text{intr}} / speed_{\text{cur}}$$

Наступний крок це порівняння та генерація попередження:

- алгоритм отримує залишковий зелений час (remaining_green_time) з повідомлення BSM від RSU.

- якщо remaining_green_time менше за timeintr, це означає, що світлофор стане червоним до того, як ТЗ проїде перехрестя. У цьому випадку видається попередження RLVW.

Цей підхід забезпечує своєчасне попередження водіїв, допомагаючи запобігти небезпечному проїзду на червоний сигнал світлофора.

2.4.3. Алгоритм попередження про рух у неправильному напрямку

Для додатка алгоритм попередження про рух у неправильному напрямку (WWEW) припускається, що RSU встановлений, наприклад, на стовпі вуличного освітлення (точка $B(x_2,y_2,z_2)$). Напрямок RSU (H_3) калібрується так, щоб він був протилежним (180◦) до законного напрямку руху.

Зв'язок V2I дозволяє CV (наприклад, $C(x_3,y_3,z_3)$, що рухається неправильно) отримувати дані RSU для прогнозування WWD.

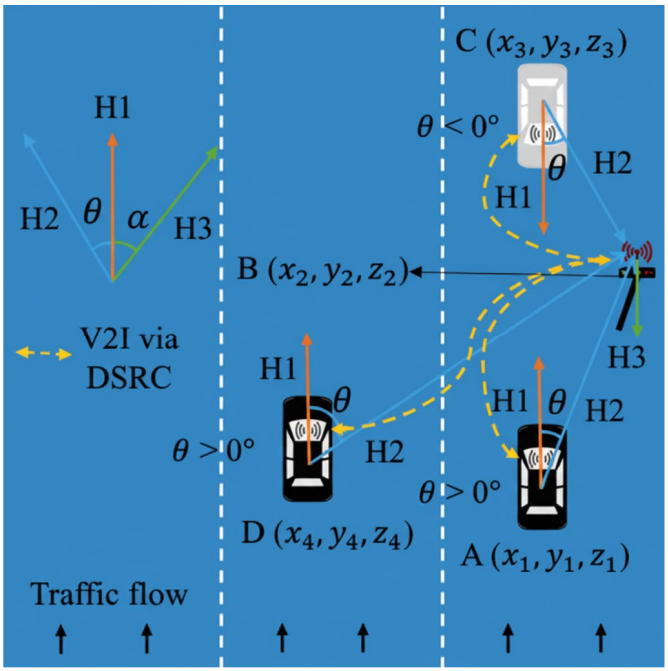


Рис. 2.9. Ілюстрація алгоритму руху в неправильному напрямку

З рисунку 2.9 видно, що для транспортних засобів, що рухаються в законному напрямку (точки A,D):

- Відносний кут θ є позитивним.

- Абсолютне значення різниці кутів напрямку $|\alpha|$ знаходиться в межах $[135^\circ, 180^\circ]$.

- Для транспортного засобу, що рухається в неправильному напрямку (точка C):

- θ буде негативним.

- $|\alpha|$ буде в межах $[0^\circ, 45^\circ]$.

Лістинг 2.3. Алгоритм попередження про рух у неправильному напрямку

```
function DETECT_WRONG_WAY_DRIVING()
    // Обробка вхідних даних
    Process HV and RSU BSMs

    // Обчислення просторових параметрів
    Compute distance 'd' between HV and RSU locations
    Calculate relative angle 'θ'
    α = RSU_heading - HV_heading

    // Перевірка умови руху в неправильному напрямку
    if 0° < |α| < 45° and θ < 0° and d < 50 then
        // Умова:
        // 1. |α| (різниця між напрямками RSU та HV) вказує на протилежний рух (0°–45°)
        // 2. θ (відносний кут) від'ємний, що означає, що RSU знаходиться позаду HV.
        // 3. d (відстань) менше 50 м (в межах критичної зони).
        Issue WWEW // Видати попередження про рух у неправильному напрямку
    end if
end function
```

Алгоритм WWEW (Wrong-Way Entry Warning), що поданий в лістингу 2.3 призначений для ідентифікації та попередження водія про те, що його транспортний засіб (ТЗ) рухається у протилежному (неправильному) напрямку щодо законного потоку руху на даній ділянці дороги, використовуючи комунікацію V2I (Vehicle-to-Infrastructure). Цей алгоритм критично залежить від калібрування дорожнього пристрою (RSU) - напрямок

RSU (H_3) має бути встановлений таким чином, щоб він був протилежним до правильного напрямку руху.

Цей підхід дозволяє швидко та з низькими обчислювальними витратами виявити WWD, використовуючи лише інформацію про відносне положення та напрямок.

Висновки до розділу

Другий розділ присвячено розробці та вдосконаленню алгоритмічних методів визначення відносного положення транспортних засобів та реалізації програмних засобів попередження потенційно небезпечних ситуацій. Доведено, що обчислення відносного кута, прогнозування положення та аналіз траєкторій на основі кооперативних повідомлень безпеки (BSM) дозволяють істотно підвищити точність оцінки ситуацій дорожнього руху та забезпечити основу для створення координаційних V2V- та V2I-додатків.

Розроблені алгоритми попередження про критичні сценарії (лобове зіткнення, наявність сліпої зони, поява повільного транспортного засобу, заборона обгону) демонструють можливість ефективного застосування аналітичних методів відносного позиціонування у системах забезпечення ситуаційної обізнаності. На основі формалізації правил дорожнього руху та параметрів динаміки руху побудовано універсальні схеми прийняття рішень, придатні для реалізації в автономних транспортних системах різних рівнів автономності.

Реалізовані алгоритми демонструють, що інтеграція даних з інфраструктурних об’єктів у поєднанні з відносним позиціонуванням підвищує точність оцінки дорожньої ситуації та сприяє прийняттю більш обґрунтованих рішень у режимі реального часу.

РОЗДІЛ 3. МЕТОДОЛОГІЯ ПРОЕКТУВАННЯ БЕЗПЕЧНОЇ ВЗАЄМОДІЇ АВТОНОМНИХ ТРАНСПОРТНИХ СУТНОСТЕЙ

3.1. Особливості побудови архітектури та методології верифікації для безпеки та захисту взаємодії автономних транспортних засобів

Тенденція до розробки та впровадження автономних транспортних засобів (АТЗ) суттєво зросла протягом останніх років. Очікування, пов'язані з цією технологією, включають значне зниження кількості дорожньо-транспортних пригод (унаслідок мінімізації людського фактора), підвищення паливної економічності та загальне збільшення пропускної здатності транспортної системи

Наразі дорожні випробування АТЗ активно проводяться у різних регіонах (наприклад, Uber, Valeo). Наприклад, у випробуваннях Valeo автономний автомобіль функціонує на об'їзній дорозі, динамічно адаптуючи свою швидкість до умов руху. Під час таких випробувань регуляторні органи вимагають обов'язкової присутності водія-супервайзера. Його основна функція полягає у втручанні в надзвичайних ситуаціях, а його присутність також сприяє психологічному комфорту інших учасників руху, які можуть не усвідомлювати автономний статус ТЗ. Кінцевою метою галузі, однак, є досягнення повної автономності (Full Autonomy), що вимагатиме застосування відповідних розподілених методів контролю для максимізації як безпеки (safety), так і середньої пропускної здатності.

Аналіз поточних дорожніх випробувань виявляє низку відкритих проблем на науковому, технічному, економічному та соціальному рівнях. Оптимальне вирішення завдань контролю руху передбачає кооперативний підхід, де ТЗ обмінюються інформацією для спільного формування деталізованої картини поточної дорожньої ситуації. Це, у свою чергу, вимагає ефективного та захищеного механізму комунікації V2V (Vehicle-to-

Vehicle) для обміну критично важливою інформацією (наприклад, про перешкоди).

Гетерогенність проблеми та швидка еволюція технології вимагають стандартизованої інфраструктури та методології проектування. Це дозволить розробникам однозначно формулювати вимоги та властивості системи, забезпечуючи функціональні та нефункціональні гарантії safety. Такий підхід необхідний для сприяння впровадженню технології та зниженню сприйнятого користувачем ризику. З огляду на це, зростає потреба у спільному розгляді властивостей безпеки (safety) та захищеності (security). Однак ця інтеграція ускладнюється тим, що ці властивості традиційно специфікуються, аналізуються та розробляються різними командами, що використовують різний досвід та інструментарій. Крім того, у багатьох критичних застосуваннях властивості safety повинні відповідати специфічним галузевим стандартам для цілей сертифікації.

Основна новизна та внесок даної роботи полягає у розробці методології, яка забезпечує повний цикл проектування, починаючи від вимог, сформульованих природною мовою, до етапу прототипування системи автономних ТЗ, з особливим акцентом на інтегрованих вимогах safety та security.

Запропонована методологія відповідає вимогам стандарту safety ISO 26262. Хоча новий стандарт ISO/PAS 21448:2019 (SOTIF — Safety of the Intended Functionality), що стосується систем надзвичайного втручання, не є предметом прямого обговорення, як розширення ISO 26262, наша робота може бути адаптована до нього як перспективний напрямок.

Ми демонструємо, що реалізація захищеного протоколу комунікації між АТЗ може викликати конфлікти з вимогами safety та потенційно знижувати загальний рівень безпеки системи. Завдяки використанню фреймворку Arrowhead [14] ми отримуємо можливість вибору оптимального алгоритму security, який є сумісним із критичними вимогами safety.

Фреймворк Arrowhead (Arrowhead Framework) — це відкритий, комплексний архітектурний фреймворк, розроблений для підтримки та спрощення створення великих, розподілених, сервіс-орієнтованих систем (Service-Oriented Architectures, SOA) в Інтернеті речей (IoT) та складних кіберфізичних системах, зокрема в автономних транспортних засобах.

У контексті даної роботи його основна роль полягає у забезпеченні вимог захищеності (security) системи.

Розглянемо його ключові функції та роль у проектуванні:

1. Сервіс-орієнтована архітектура (SOA).

Arrowhead надає стандартизовану структуру, де різні компоненти системи (наприклад, автономні транспортні засоби V1,V2,V3) взаємодіють шляхом обміну послугами. Це робить систему більш модульною та гнучкою.

2. Автентифікація та авторизація.

Це основна функція фреймворку в даній роботі. Arrowhead гарантує, що лише автентифіковані (перевірені) транспортні засоби можуть брати участь у комунікації і лише авторизовані (дозволені) транспортні засоби можуть надсилати або отримувати критично важливі команди безпеки, такі як сигнал про екстрене гальмування.

3. Вирішення конфлікту Safety-Security.

Процеси автентифікації зазвичай вимагають певного часу (затримки). У критичних системах, таких як колона автономних авто, ця затримка може поставити під загрозу безпеку (safety) (наприклад, спричинити зіткнення).

Arrowhead вирішує цю проблему, забезпечуючи транспортні засоби токенами безпеки заздалегідь. Це мінімізує затримку, необхідну для автентифікації при обміні терміновими повідомленнями.

4. Відповідність вимогам.

Інтеграція Arrowhead у методологію проєктування дозволяє інженерам із самого початку враховувати вимоги захищеності та верифікувати їх за допомогою контрактного підходу.

Пропонована робота зосереджена на поєднанні двох ключових аспектів:

А. Контрактний підхід для Safety.

Ми вирішуємо проблему специфікації та верифікації коректної та безпечної поведінки системи за допомогою контрактного підходу. Цей підхід забезпечує розділення припущень (assumptions) та гарантій (guarantees), розподіляючи властивості системи між різними компонентами та командами. Це особливо ефективно для розподілених систем, де множина ТЗ колективно сприяє досягненню кінцевого результату. Ми інтегруємо контракти зі стандартизованими мовами, такими як SysML, для полегшення їхнього застосування в традиційних процесах проектування, а також із тривимірними анімованими сценаріями для покращення розуміння кібер-фізичної системи, що аналізується.

Б. Сервісно-орієнтована архітектура для Security.

Контрактна техніка поєднується з SOA, що базується на фреймворку Arrowhead, для керування всіма V2V комунікаційними взаємодіями, забезпечуючи властивості захищеності. Цей підхід надає переваги SOA (повторне використання сервісів, слабке зв'язування) [16], посилені функціональністю Arrowhead (зокрема, виявлення сервісів, оркестрація та security). Стандартизація всіх взаємодій через формально визначені сервісні інтерфейси, подібно до контрактного підходу, спрощує застосування до розподілених систем.

Ця методологія підтримується спеціальним інструментарієм, який інтегрує різні компоненти та керує аналітичним двигуном.

3.2. Застосування контрактного підходу для гарантування безпеки систем автономних транспортних засобів

З метою зниження складності проектування систем та забезпечення структурованості, галузеві стандарти, такі як AUTOSAR [19], визначають

багаторівневу абстрактну архітектуру. Ця структура передбачає повне розділення програмних функцій транспортного засобу, починаючи від базових контролерів апаратного забезпечення.

Проте, для забезпечення виконання вимог безпеки (safety), гіпотеза про поширення відмов у таких архітектурах часто вимагає аналізу, який перетинає межі декількох рівнів. Цей необхідний міжрівневий аналіз порушує фундаментальний принцип розділення обов'язків (separation of concerns), закладений у цих архітектурах.

Для вирішення цієї дилеми ми застосовуємо контрактний підхід (Contract-Based Approach). Цей метод абстрагується від конкретної реалізації програмних та апаратних компонентів проектованої комунікаційної системи шляхом визначення припущень (assumptions) та гарантій (guarantees) безпеки для кожного компонента [20].

Заміна реальних реалізацій формальними контрактами дозволяє розробникам значно скоротити час проектування та імплементації системи. У даній роботі контракти використовуються для створення міжтранспортної специфікації, яка, попри це, зберігає структурне розділення ТЗ завдяки чіткому розмежуванню між припущеннями та гарантіями.

Ефективність застосування контрактного підходу посилюється використанням формалізованих мов, які підтримують моделювання та виконання моделей поширення відмов.

У цій роботі ми використовуємо мову на основі шаблонів BCL (Behavioural Contract Language).

BCL орієнтована на напівформальні симуляційні методи, які довели свою здатність ефективно обробляти реальні великомасштабні системи, на відміну від підходів, що ґрунтуються виключно на повноцінних формальних методах.

Крім того, BCL має можливість інтеграції з середовищем Matlab/Simulink і продемонструвала ефективність у оцінці надійності для

сервісно-орієнтованих специфікацій (SOA), що є архітектурною основою даної роботи.

Формальна семантика BCL визначена через відображення його шаблонів у лінійну темпоральну логіку (LTL - Linear Temporal Logic). Це відображення також слугує реалізацією контракту, що використовується для аналізу специфікації під час симуляції. Незважаючи на теоретичну можливість використання LTL безпосередньо для специфікації контракту, BCL пропонує більш простий та обмежений формалізм. Це сприяє мінімізації помилок та полегшує його освоєння проектувальниками, які не мають глибокого досвіду у сфері формальних тверджень.

3.3. Застосування фреймворку Arrowhead для забезпечення захищеності сервісно-орієнтованих архітектур

Для специфікації, аналізу та забезпечення властивостей захищеності (security) системи, зокрема аутентифікації та авторизації сигналів керування, ми використовуємо Фреймворк Arrowhead.

У структурі Arrowhead усі функціональні взаємодії реалізуються через сервіси (Services).

1. Системи (Systems). Сервіси виробляються (надаються) та споживаються системами, які виконуються на пристроях (Devices).

2. Пристрої. Пристрій може бути будь-яким постачальником обчислювальних можливостей: вбудованою системою, ноутбуком, віртуальною машиною у хмарі тощо.

3. Розподілений Додаток. Реалізується як система систем (System of Systems) — набір систем, що взаємодіють відповідно до парадигми Arrowhead.

4. Комунікація. Сервіси пропонуються як функції, доступні для дистанційного виклику. Комунікація є незалежною від базових протоколів, що дозволяє використовувати різні технології:

- Формат кодування повідомлень: (наприклад, JSON, XML, MQTT).

- Комунікаційний протокол: (наприклад, HTTP, HTTPS, XMPP).

- Комунікаційна парадигма: (наприклад, REST, publish/subscribe).

Arrowhead нормалізує всі взаємодії за допомогою Сервісно-Орієнтованої Архітектури (SOA). Цей підхід забезпечує взаємосумісність між системами, які можуть бути засновані на гетерогенних технологіях.

Ключові переваги SOA в контексті Arrowhead:

1. Спрощує розробку програмного забезпечення та діяльність з технічного обслуговування.

2. Значно скорочує час виведення продукту на ринок, підтримуючи розгортання та покращуючи підтримку взаємопов'язаних кооперативних додатків.

3. Будучи сервісно-орієнтованими, додатки отримують переваги слабкого зв'язування у просторі, часі та синхронізації.

Сервіси Arrowhead поділяються на дві основні категорії:

- Додаткові сервіси (Application Services) - Реалізують функціональні вимоги конкретних випадків використання системи (наприклад, алгоритм керування ТЗ).

- Базові сервіси (Core Services): Управляють нефункціональними вимогами (як-от захищеність) та забезпечують роботу самої платформи. Вони надають відчутну додаткову цінність для користувачів фреймворку, дозволяючи зосередитися на основній функціональності системи.

Базові сервіси поділяються на "обов'язкові" (mandatory), які є необхідними для будь-якої системи систем, та "підтримувальні" (supporting), що надають утиліти.

Обов'язкові базові сервіси (рисунок 3.1):

- ServiceRegistry (реєстр сервісів) - містить інформацію про доступні сервіси в системі.

- Orchestration (оркестрація) - керує послідовністю та вибором сервісів, які мають бути викликані.

- Authorization (авторизація) - забезпечує, щоб лише авторизовані системи могли споживати певні сервіси.

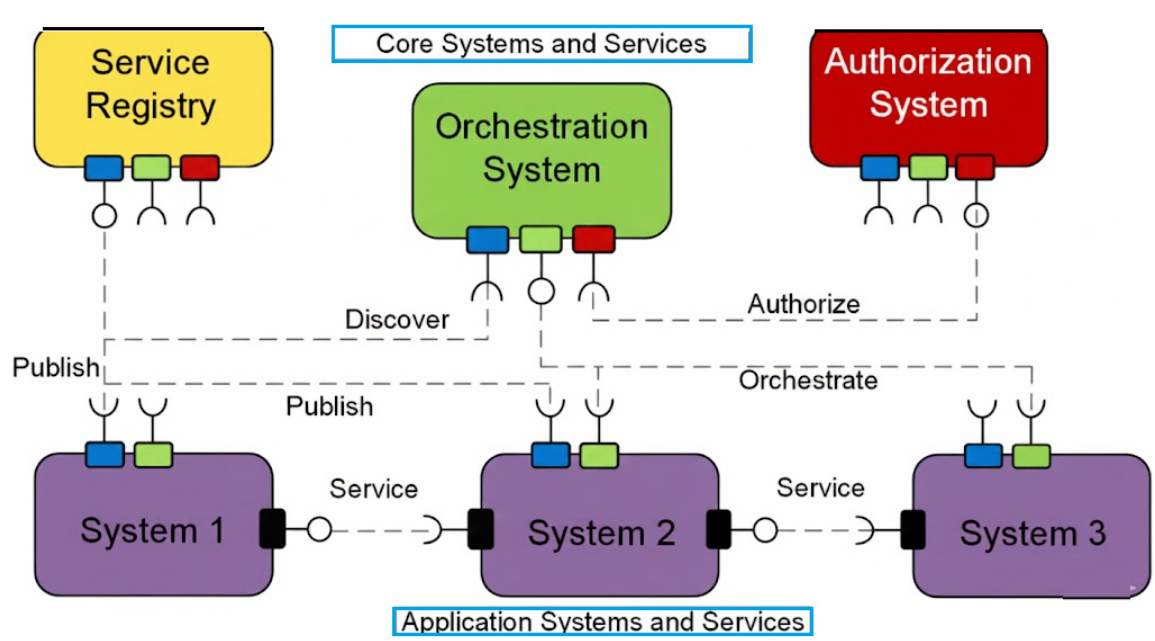


Рис. 3.1. Типова система систем засобами фреймворку Arrowhead

Для глибшого розуміння технічного підґрунтя нашого рішення, наступні розділи нададуть детальний огляд функціональності цих обов'язкових базових сервісів.

3.4. Методологія проектування безпечних автономних систем основі контрактного аналізу

У цьому розділі представлена методологія, що структуровано та систематично організовує процес аналізу для забезпечення його ефективної реалізації в упорядкованому та зручному для проектування процесі.

Оскільки предметне дослідження, особливо в контексті автономних транспортних засобів, має сильні залежності від фізичних факторів (таких як затримка комунікації та продуктивність сенсорів) та ступеня автономності, що визначається програмним керуванням, ми працюємо в галузі кібер-фізичних систем (Cyber-Physical Systems, CPS).

З цієї причини пропонується інклюзивний підхід, що ґрунтується на використанні гетерогенних мов та інструментів. Це дозволяє задіяти різні функціональні можливості та команди. Ключовою перевагою є відкритий та модульний потік проектування, який забезпечує гнучкість та адаптивність до внутрішніх мов, інструментів та процесів, прийнятих промисловістю.

Запропонований підхід, візуалізований на рисунку 3.2, складається з серії кроків, які підтримуються спеціалізованим інструментарієм CAT (Contract Analysis Tool) та Arrowhead Framework.

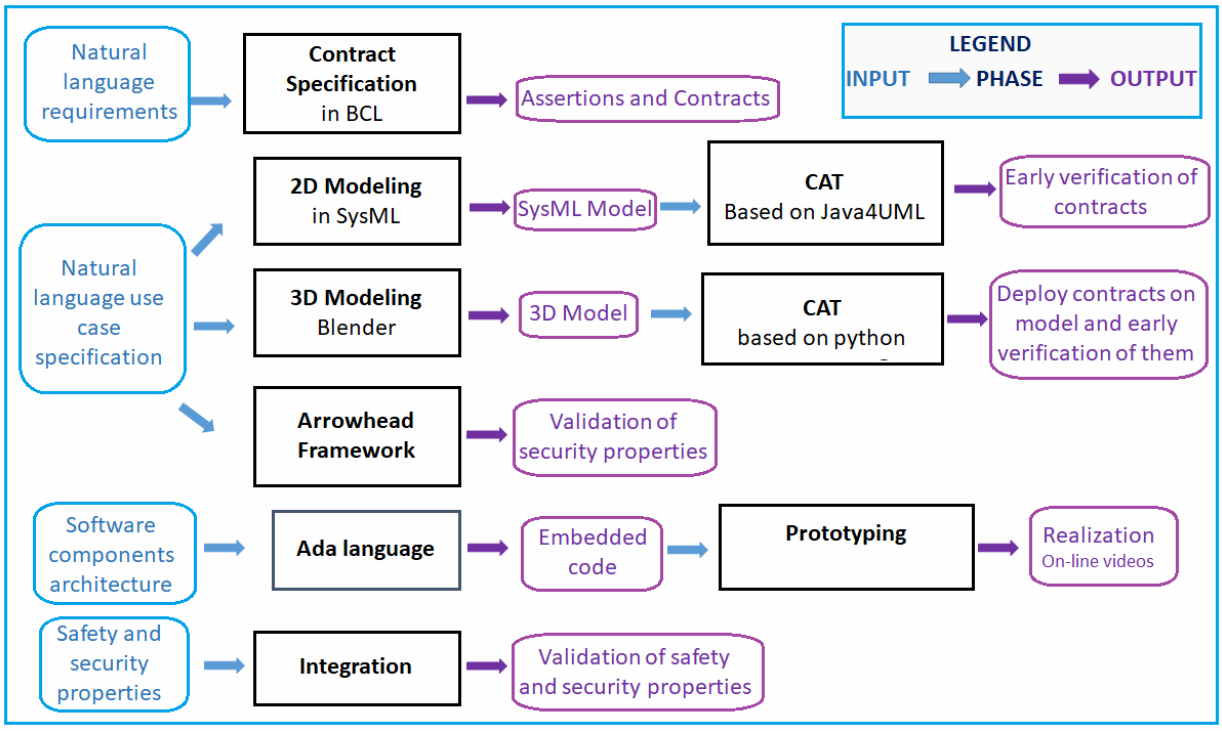


Рис. 3.2. Етапи потоку проектування, від специфікації до прототипування

На рисунку 3.2 основні кроки методології позначені блоками, вхідна інформація для кожного етапу показана ліворуч, вихідна інформація показана праворуч.

Як і в типових процесах проектування, вихід однієї фази часто стає входом для наступного етапу, особливо на стадіях валідації.

На рисунку 3.2 представлено багатоетапну методологію проектування та верифікації кібер-фізичних систем, розроблену для створення безпечної та

захищеної комунікації автономних транспортних засобів. Цей процес є інклюзивним, поєднуючи специфікацію вимог, моделювання, забезпечення захисту та інтеграцію.

Методологія, що подана на рис. 3.2 складається з низки послідовних фаз (блоків), які отримують вхідні дані (ліворуч) і генерують вихідні дані (праворуч).

I. Етапи специфікації та моделювання

Ці початкові фази даної методології фокусуються на перетворенні високоабстрактних вимог у формалізовані та модельні подання системи (таблиця 3.1).

Таблиця 3.1.

Початкові фази

ФАЗА (PHASE)	**ВХІДНІ ДАНІ (INPUT)**	**ВИХІДНІ ДАНІ (OUTPUT)**	**ОПИС**
Contract Specification in BCL (специфікація контрактів у BCL)	Natural language requirements (Вимоги природною мовою)	Assertions and Contracts	Вимоги безпеки (safety) формалізуються за допомогою мови поведінкових контрактів (BCL). Це створює набір формальних припущень і гарантій.
2D Modeling in SysML (2D моделювання в SysML)	Natural language use case specification (Специфікація випадків використання природною мовою)	модель SysML	Специфікація випадків використання перетворюється на двовимірну структурну та поведінкову модель за допомогою SysML (System Modeling Language).
3D Modeling Blender (3D моделювання Blender)	Natural language use case specification (Специфікація випадків використання природною мовою)	3D модель	Специфікація використовується для створення тривимірної візуальної моделі сценарію (наприклад, у Blender), що покращує розуміння кібер-фізичної системи.

II. Етапи верифікації та аналізу

Ці фази використовують інструментарій CAT (Contract Analysis Tool) для ранньої перевірки коректності та задіяння Arrowhead для забезпечення захисту (таблиця 3.2).

Таблиця 3.2.

Фази верифікації та аналізу

ФАЗА (PHASE)	**ВХІДНІ ДАНІ (INPUT)**	**ВИХІДНІ ДАНІ (OUTPUT)**	**ОПИС**
CAT (Based on Java4UML)	SysML Model	Рання верифікація контрактів	Інструмент CAT (на базі Java4UML) використовується для ранньої верифікації логічної коректності контрактів на основі структурної моделі SysML.
CAT (based on python)	3D Model	Розгортання контрактів на моделі та їх рання верифікація	CAT (на базі Python) розгортає формалізовані контракти на 3D-моделі для візуальної та симуляційної перевірки.
Arrowhead Framework	Natural language use case specification	Валідація властивостей захищеності	Використовується Arrowhead Framework для специфікації та валідації нефункціональних вимог захищеності (наприклад, аутентифікації та авторизації комунікації V2V).

III. Етапи реалізації та інтеграції

Фінальні фази зосереджені на перетворенні моделей у виконуваний код, створенні прототипу та фінальній валідації інтегрованої системи (таблиця 3.3).

Таблиця 3.3.

Фази реалізації та інтеграції

ФАЗА (PHASE)	**ВХІДНІ ДАНІ (INPUT)**	**ВИХІДНІ ДАНІ (OUTPUT)**	**ОПИС**
Ada language	Архітектура програмних компонентів	Вбудований код	Архітектура компонентів перетворюється на вбудований код з використанням мови Ada, що часто

ФАЗА (PHASE)	ВХІДНІ ДАНІ (INPUT)	ВИХІДНІ ДАНІ (OUTPUT)	ОПИС
			використовується для safety-критичних систем.
Прототипування	Embedded code	Реалізація онлайн-відео	Вбудований код розгортається на фізичному прототипі (наприклад, модельних автомобілях) для демонстрації та запису реальної поведінки.
Інтеграція	Властивості безпеки та захищеності	Валідація властивостей безпеки та захищеності	Фінальний етап, на якому проводиться комплексна валідація того, що інтегрована система відповідає як вимогам безпеки safety, так і захищеності security.

Методологія забезпечує простежуваність від вимог природною мовою до фізичної реалізації, використовуючи формальні контракти BCL для safety та Arrowhead Framework для security, підтримувані інструментом CAT.

Базова архітектура інструментарію, розроблена для підтримки логічних кроків методології, представлена на рисунку 3.3

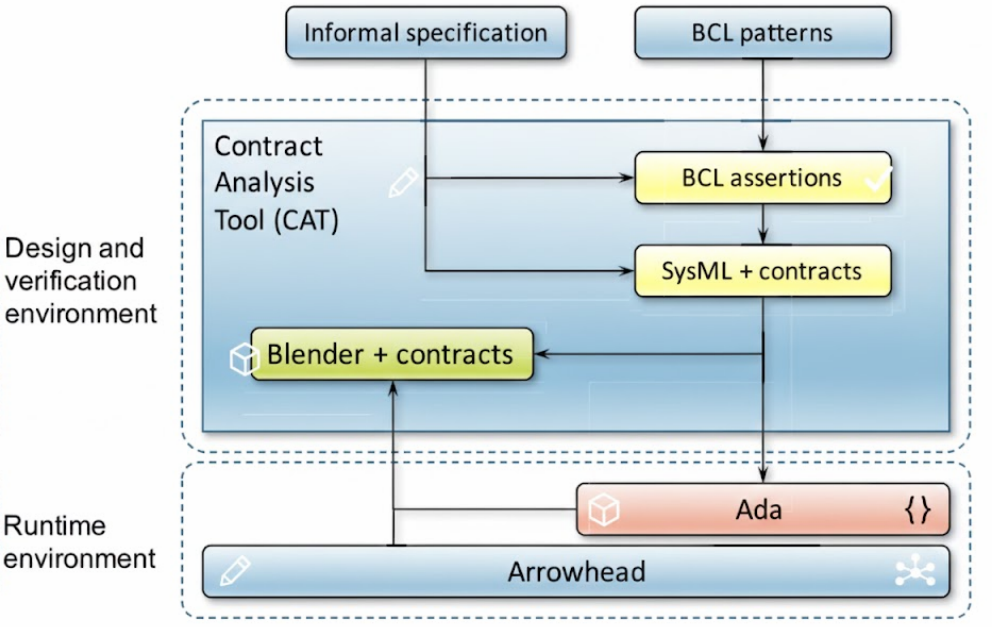


Рис. 3.3. Архітектура інструментарію, що відображає потік інформації між проектуванням та верифікацією та середовищем виконання

Вхідні дані проектувальника (дизайнера) обробляються та керуються інструментом CAT для цілей проектування та верифікації. CAT, у свою чергу, використовує Arrowhead Framework як середовище виконання (Runtime Environment), що забезпечує необхідні нефункціональні сервіси (наприклад, security).

Подальші підрозділи цього розділу містять детальний опис кожного окремого кроку пропонованої методології.

3.5. Детальний опис початкових етапів методології проектування

У цьому розділі детально розглянуто перші чотири етапи запропонованої методології, які охоплюють специфікацію вимог, моделювання та початкову верифікацію системи автономних транспортних засобів.

1. Специфікація контрактів у BCL

Діяльність починається з неформального визначення потреб користувачів, які первинно виражені природною мовою, з обов'язковим акцентом на критичних аспектах безпеки (safety) та захищеності (security).

Щоб зробити ці вимоги придатними для формального аналізу, вони повинні бути формалізовані разом з початковою специфікацією архітектури. Для цього ми використовуємо контрактний підхід, який чітко розмежовує відповідальність різних компонентів шляхом визначення припущень (Assumptions) та гарантій (Guarantees). Ці елементи виражаються як твердження (assertions) у мові поведінкових контрактів (BCL) [22], що базується на шаблонах.

BCL є зручною мовою, оскільки вимоги природною мовою логічно відображаються у її шаблонах, які додають структуру і допомагають уникнути поширених помилок специфікації.

Контракти, виражені твердженнями, підлягають відстеженню, вимірюванню та верифікації протягом усього процесу проектування, що

сприяє вирішенню міжрівневих проблем у складних архітектурах. Специфікація архітектури та функціональності проекту паралельно створюється у SysML (System Modeling Language), стандарті OMG, широко використовуваному в промисловості.

2. 2D моделювання

Специфікація випадку використання, яка зазвичай також первинно виражається природною мовою, моделюється за допомогою компонентно-орієнтованого формалізму.

У цій роботі ми обираємо SysML як основний інструмент моделювання, оскільки він є поширеним стандартом OMG, що підтримується низкою промислових (наприклад, IBM, PTC/Artisan Studio) та відкритих інструментів (Papyrus). На рисунку 3.2 модель SysML є безпосереднім результатом цього етапу моделювання та виступає вхідними даними для наступної фази, що використовує CAT.

3. Реалізація версії для Eclipse

На цьому етапі відбувається інтеграція контрактів та архітектури системи. Контракти та архітектура об'єднуються у комбіновану специфікацію SysML, де:

- окремі твердження (припущення та гарантії) пов'язуються з портами компонентів.

- самі контракти асоціюються з компонентами системи.

Це забезпечує повторне використання тверджень для різних сутностей у дизайні, що є критично важливою функцією для підвищення ефективності розробки та запобігання помилкам.

Інструмент CAT (Contract Analysis Tool):

- інтеграція та анотація специфікації SysML контрактами реалізується через CAT.

- CAT визначає мета-модель та предметно-орієнтовану мову (DSL), яка розширює SysML відповідними концепціями контракту, гарантії та припущення як обмеженнями UML Metaclass (рисунок 3.4).

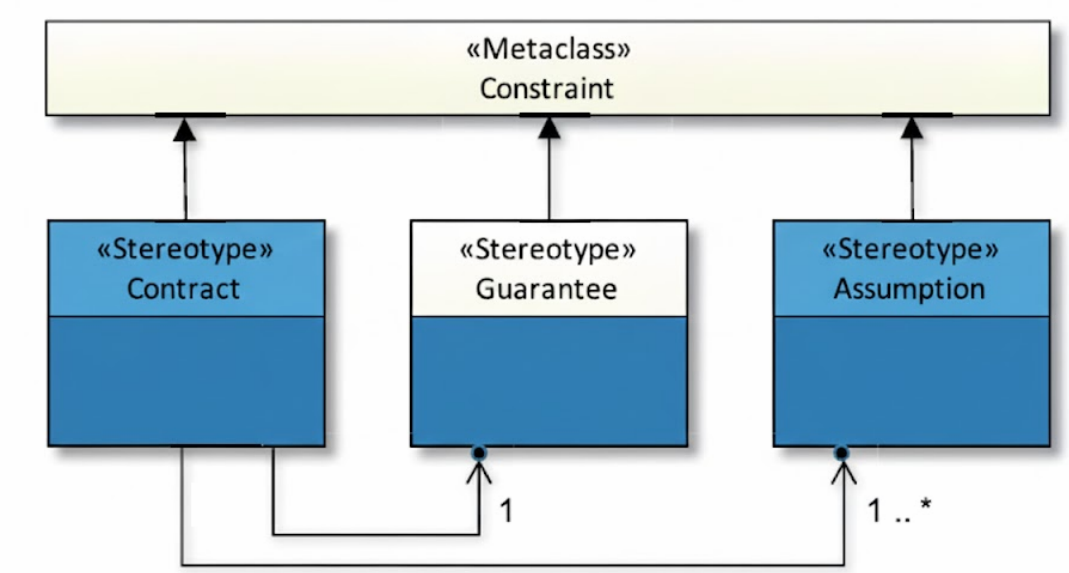


Рис. 3.4. Інтерфейс предметно-орієнтованої мови між BCL та SysML/UML

Це рішення базується на промислових вимогах, зокрема, на ранніх тестах реалізації, де властивості виражалися як обмеження OCL (Object Constraint Language). CAT функціонує як додаток до Eclipse та базується на Java4UML. Оскільки припущення та гарантії можуть належати до кількох різних контрактів, вони представлені як стереотипи.

4. 3D Моделювання

Відмінною особливістю методології, особливо цінною для динамічних та розподілених сценаріїв (таких як автономні транспортні засоби), є можливість симуляції системи за допомогою тривимірної анімаційної моделі.

Перевагою є те, що тривимірне представлення графічно спрощує складні взаємодії, надаючи краще розуміння аналізованих проблем. Для моделювання обрано Blender — безкоштовний та відкритий інструмент, що пропонує гнучкі інтерфейси для інтеграції зовнішнього інструментарію.

Найбільш важливим аспектом є функціональність скриптингу на Python, яка дозволяє автоматизувати завдання та створювати власні інструменти. Вбудовані консоль та текстовий редактор Python спрощують інтеграцію скриптів та налаштування програмного забезпечення.

Тривимірна модель є результатом цього етапу і виступає вхідними даними для наступної фази CAT.

3.6. Деталізація етапів аналізу та інтеграції захищеності

У цьому розділі описано фази, пов'язані з симуляційним аналізом контрактів за допомогою CAT та паралельним забезпеченням властивостей захищеності через Arrowhead Framework.

1. Реалізація на Python для 3D Симуляції

Інструмент CAT (Contract Analysis Tool) використовується через його Python-інтерфейс, інтегрований із середовищем Blender. Це дозволяє розширити 3D-модель контрактами, де твердження формалізовані в лінійній темпоральній логіці (LTL), яка є основною формальною семантикою мови BCL.

Процес верифікації:

1. Розширення моделі, тобто контракти (виражені в LTL) застосовуються до 3D-моделі.

2. Покрокова перевірка як полягає в тому, що CAT перевіряє дійсність контрактів для кожного часового кроку 3D-симуляції.

3. Результат аналізу:

- У разі позитивного результату, CAT надає докази коректної та безпечної поведінки системи.

- У разі негативного результату, інструмент виявляє порушення контрактів.

2. Фреймворк Arrowhead та специфікація захищеності

Специфікація аспектів захищеності (security) системи слідує паралельному шляху "знизу-вгору" та ґрунтується на використанні Arrowhead фреймворку. Ключова парадигма проектування полягає у розділенні протоколу додатку від базових інфраструктурних сервісів, які реалізують комунікаційні примітиви.

Комунікація між системами концептуально поділяється на три критичні етапи, кожен з яких повинен бути врахований комунікаційним примітивом:

- виявлення одержувача - знаходження цільової системи.

- авторизація - перевірка прав доступу.

- обмін даними - фактична передача інформації.

SOA rrowhead дозволяє використовувати спеціалізовані базові сервіси для всіх цих операцій (Discovery, Authorization). Фреймворк надає гнучкість у виборі різних реалізацій комунікаційних примітивів. Це дозволяє порівнювати різні підходи (наприклад, виконання етапів виявлення та аутентифікації заздалегідь для підвищення продуктивності) та використовувати різні протоколи для кожного етапу для забезпечення різних ступенів захищеності повідомлень.

Вимоги захищеності (наприклад, використання визначених процесів для обміну даними або налаштування комунікації) можуть накладати обмеження на поведінку системи, що потенційно може призвести до несподіваного порушення контракту (safety).

Комунікаційні примітиви, що розглядаються контрактами безпеки, інтерпретуються як абстрактні операції, а їхня часова інтерпретація як гарантій є максимально "лінивою" (lazy), щоб відкласти вирішення до фактичних затримок виконання. Аналіз часу, таким чином, поділяється на два етапи:

1. Протокол додатку: аналізується протокол додатку для побудови виразу, що відображає часові гарантії протоколу.

2. Реалізація примітивів: враховуються конкретні реалізації примітивів для пов'язування затримок з їхнім виконанням.

Значення, що використовуються для затримки, пов'язаної з кожною реалізацією примітиву, обчислюються або шляхом формальних доказів, або за допомогою бенчмаркінгу.

3.7. Інтеграція, реалізація та відповідність стандартам методології проектування

Цей розділ деталізує фінальні етапи запропонованої методології, включаючи інтегровану перевірку, розробку вбудованого програмного забезпечення та оцінку відповідності міжнародним стандартам безпеки.

1. Інтеграція та інтегрована перевірка

На етапі інтеграції 3D-модель (Blender) додатково анотується інформацією про продуктивність, яка розглядається для кожного комунікаційного примітиву. Ця абстрактна інформація надалі вирішується до фактичних часових затримок, зібраних безпосередньо з прототипу системи.

Інтегрована перевірка може бути досягнута шляхом підтвердження узгодженості (consistency) реалізації кожного компонента з його відповідним контрактом. Якщо ця вимога задоволена, і за умови відсутності порушень, виявлених у попередньому аналізі CAT, теорія контрактів гарантує загальну коректність системи через свої композиційні правила.

Хоча цей підхід може бути потенційно консервативним, його перевага полягає у можливості окремої перевірки реалізації контракту для кожного компонента. У разі виявлення порушень, аналіз CAT може бути повторений із уточненими контрактами та показниками продуктивності для забезпечення загальної інтегрованої перевірки.

У нашому конкретному випадку дослідження ми оцінюємо як абстрактну поведінку (контракти), так і виміряну продуктивність для вирішення критичної взаємодії між контрактами безпеки (safety) та захищеності (security).

2. Вибір мови реалізації

Наступним кроком є розробка фактичного вбудованого програмного забезпечення. Хоча на цьому етапі в поточному процесі проектування все ще вимагається ручна праця, досвід та результати, отримані завдяки ранній оцінці моделі, є вирішальними для реалізації добре структурованого коду, що

відповідає системним обмеженням. Ми орієнтуємося на мову Ada. Цей вибір обумовлений гарантіями, які вона може надати у сфері критичних систем, та її широким впровадженням у промисловості.

3. Відповідність стандарту ISO

Комбіноване використання Arrowhead фреймворку та контрактного підходу відповідає директивам міжнародних стандартів безпеки, зокрема ISO 26262 для автомобільної галузі. Відповідність стандартам визначається шляхом досягнення однієї або кількох цілей, визначених у нормативних документах, без втручання в існуючі промислові процеси. Наша методологія характеризується певною незалежністю від конкретних мов, що дозволяє компаніям:

- інтегрувати підхід у свої внутрішні процеси (включаючи політику управління безпекою).

- використовувати свої власні кваліфіковані інструменти.

- підтримувати аргументи, необхідні для представлення органу сертифікації.

Стандарт ISO 26262 вимагає розробити опис елемента щодо його функціональності, інтерфейсів, умов навколишнього середовища, правових вимог, відомих небезпек тощо. Межі елемента та його інтерфейси, а також припущення щодо інших елементів, систем та компонентів визначаються.

Ми пропонуємо просту стратегію досягнення цієї мети:

1. Контракти використовуються як засіб специфікації властивостей безпеки та захищеності, яких системи повинні дотримуватися на інтерфейсах системи та компонентів.

2. Система моделюється у 2D (SysML) для специфікації компонентів, функціональності та вимог, а потім симулюється у 3D (Blender) для візуалізації всієї системи та бажаних властивостей.

3. CAT (Contract Analysis Tool) використовується для специфікації та перевірки властивостей, пов'язаних з безпекою, як у 2D, так і в 3D моделюванні.

4. Властивості захищеності специфікуються контрактами, моделюються та тестуються з використанням експериментальних даних, отриманих через Arrowhead Framework.

Цей підхід допомагає промисловості надати чітке та структуроване визначення компонента та його залежностей/взаємодій з оточенням та іншими компонентами, тим самим відповідаючи ISO 26262.

Слід зазначити, що код та прийнята гетерогенна багатодатчикова архітектура, використані в даному випадку дослідження, представлені лише як приклад застосування методології і не можуть бути використані в промисловому застосуванні без додаткової повної валідації. Тим не менш, специфікація вимог safety та security та їх аналіз суттєво полегшують цей критичний крок, оскільки помилки виявляються на максимально ранніх стадіях, що зменшує загальні витрати на проектування та розробку.

3.8. Імплементація методології на прикладі колони автономних транспортних засобів

У цьому розділі ми детально ілюструємо послідовні етапи нашої методології, застосовуючи їх до повноцінного випадку дослідження, що охоплює взаємодію автономних транспортних засобів (АТЗ) у складі колони.

I. Специфікація випадку використання природною мовою

Концепція колони (Platooning). Випадок дослідження базується на концепції колони транспортних засобів (platooning). Система систем визначається як "колона", якщо її складові системи здатні функціонувати у тісній взаємодії, і, крім того, одна система може надсилати керуючі сигнали іншим. Концепція колони спрямована на збільшення пропускної здатності доріг, покращення плинності руху та може мати вирішальне значення в надзвичайних ситуаціях.

Ми розглядаємо систему колони, що складається з k підключених транспортних засобів V1…Vk. На рисунку 3.5 показана конфігурація для

k=3. У нашому випадку використання, транспортний засіб V1 є лідером (Leader) колони.

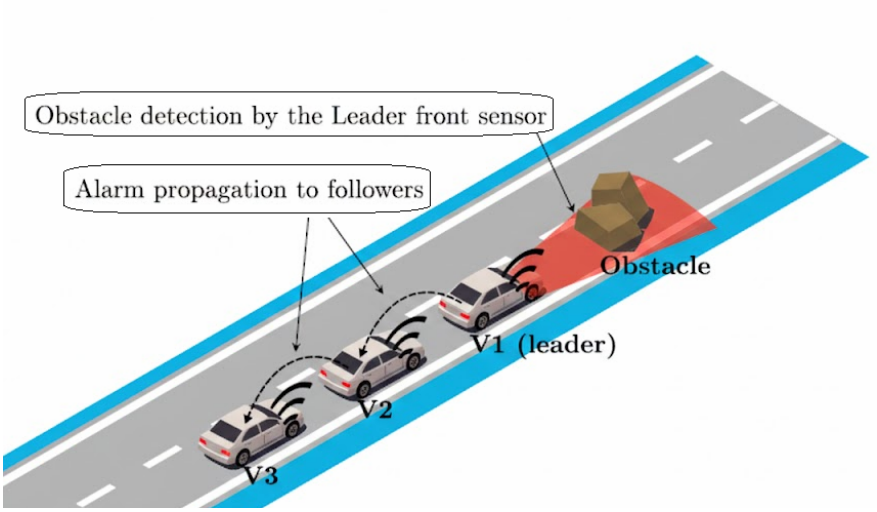


Рис. 3.5. Колона підключених автономних транспортних засобів

Вимоги природною мовою стосуються як дві критичні властивості:

1. Властивість безпеки (Safety). Комунікація V2V повинна бути безпечною. Якщо лідер (V1) виявляє перешкоду на своєму маршруті, він повинен попередити наступний транспортний засіб (V2) сигналом гальмування. Ця комунікація V2V має бути гарантована протягом заданого часового інтервалу.

2. Властивість захищеності (Security). Комунікація V2V повинна бути захищеною. Це вимагає гарантування фаз авторизації та аутентифікації.

II. Специфікація контрактів у BCL

Властивості безпеки формалізуються за допомогою контрактів. Контракт — це пара (припущення, гарантія), де гарантія є послугою, яку надає компонент, а припущення — вимогами, необхідними для виконання гарантії.

У таблиці 3.4 представлено підмножину тверджень, які використовуються для побудови контрактів, що стосуються керуючого сигналу гальмування.

Таблиця 3.4.

Підмножина тверджень для побудови контактів

Id	Твердження
r_1	Кожного разу [ПЕРЕШКОДА], тоді [КОМАНДА ГАЛЬМУВАННЯ] протягом [Y мс]
r_2	Кожного разу [ПЕРЕШКОДА], тоді [НАДІСЛАТИ Тривогу V_{i+1}] протягом [X мс]
r_3	[WiFi з'єднання] завжди
r_4	Кожного разу [ОТРИМАНО Тривогу], тоді [КОМАНДА ГАЛЬМУВАННЯ] протягом [Y мс]
r_5	Кожного разу [ОТРИМАНО Тривогу], тоді [НАДІСЛАТИ Тривогу V_{i+1}] протягом [X мс]
r_6	Кожного разу [КОМАНДА ГАЛЬМУВАННЯ], тоді [ЗУПИНКА] протягом [W мс]
r_7	Кожного разу [ПЕРЕШКОДА], тоді [ПЕРЕШКОДА УНИКНУТО] протягом [X. ($List_V$)]
r_8	[Гетерогенна багатодатчикова архітектура, здатна вимірювати відстань між двома транспортними засобами] завжди
r_9	Кожного разу [d(V_{i-1}, V_i) < 25 м], тоді [КОМАНДА ГАЛЬМУВАННЯ (V_i)] протягом [Y мс] ∩ [ПЕРЕДАТИ Тривогу Гальмування V_{i+1}] протягом [X мс] ∩ [НАДІСЛАТИ Повідомлення V_{i-1}] протягом [X мс]
r_10	Кожного разу [d(V_{i-1}, V_i) > 50 м], тоді [НАДІСЛАТИ Повідомлення Гальмування V_{i-1}] протягом [X мс]
r_11	[Підтримка Arrowhead] завжди
r_12	Кожного разу [Потрібна аутентифікація V_i - V_{i-1}], тоді [аутентифікація Arrowhead. Встановлено] протягом [Z мс]

У мові BCL контракти виражаються за допомогою шаблонів та виразів над змінними системи. Шаблони мають шарову структуру для вираження інваріантів у формі подій та їхніх часових і логічних відносин. Ключове слово "Кожного разу [E], тоді [C]" стверджує, що поява події E вимагає появи події C, а "протягом" обмежує часову область. Ключове слово "завжди" вказує на постійне виконання часових обмежень.

Позначимо ListV як скінченний список транспортних засобів, де V1 та Vk є першим та останнім відповідно. Кожен Vi∈ListV комунікує із Vi−1 та Vi+1. Кожне твердження асоційоване з компонентом як Припущення або Гарантія. Наприклад, r_5 стверджує, що якщо Vi отримує сигнал тривоги і не є останнім (Vk), то він передає сигнал тривоги далі.

Наведемо приклади контрактів.

Контракт Vi2Vi+1 затримка = (r_3, r_5): гарантує максимальний час передачі повідомлення V2V (Хмс у r_5) за умови, що WiFi з'єднання завжди доступне (припущення r_3). Це вимагає від інженерів забезпечити стійкість WiFi інфраструктури навіть у надзвичайних сценаріях.

Контракт зупинка = (r_1, r_2, r_3, r_4, r_5, r_6, r_11), r_7: стверджує максимальну затримку для зупинки всіх ТЗ у колоні. Гарантія r7 представляє WCET (Worst-Case Execution Time) для всього ланцюга керування. Параметр W залежить від змінних (швидкість, вага). Контракт базується на затримці передачі між ТЗ (X), часі внутрішньої команди гальмування (Y) та часі повної зупинки (W).

Контракт виявлення мінімальної відстані = (r_1, r_2, r_3, r_8), r_9 та контракт виявлення максимальної відстані = (r_1, r_2, r_3, r_8), r_10: визначають мінімальну та максимальну прийнятну відстань між ТЗ, підтримуючи безпечний інтервал.

Сценарій мінімальної відстані (r_9): у разі збою комунікації (наприклад, тварина перетинає дорогу між V1 та V2, але перешкода виявляється лише V2), V2 ініціює гальмування, стає лідером, передає сигнал гальмування наступному ТЗ та повідомляє попередній ТЗ (V1). Контракт виявлення мінімальної відстані слугує засобом зниження порогу тяжкості аварії. Він припускає r_8 — наявність гетерогенної багатодатчикової архітектури (датчики положення, швидкості, ультразвуковий датчик), здатної виявляти відстань.

Контракт захищеної аутентифікації = (r_11, r_12): стверджує максимальну затримку (Zмс), необхідну для аутентифікації сервісу.

Контракт безпечної та захищеної команди гальмування = (r_3, r_8, r_11, r_12), r_4: гарантує, що Vi виконує команду гальмування, отриману лише від аутентифікованого ТЗ. Припущеннями є стійка WiFi інфраструктура (r_3), гетерогенна багатодатчикова архітектура (r_8) та підтримка Arrowhead (r_11, r_12).

Для статичного аналізу на етапах проектування та розробки інженери специфікують змінні. У нашому випадку X=100мс та Y=120мс.

III. 2D Моделювання та CAT

На рисунку 3.6 представлена частина SysML моделі сценарію (рис. 3.5), включаючи Контракт Vi2Vi+1 Затримка.

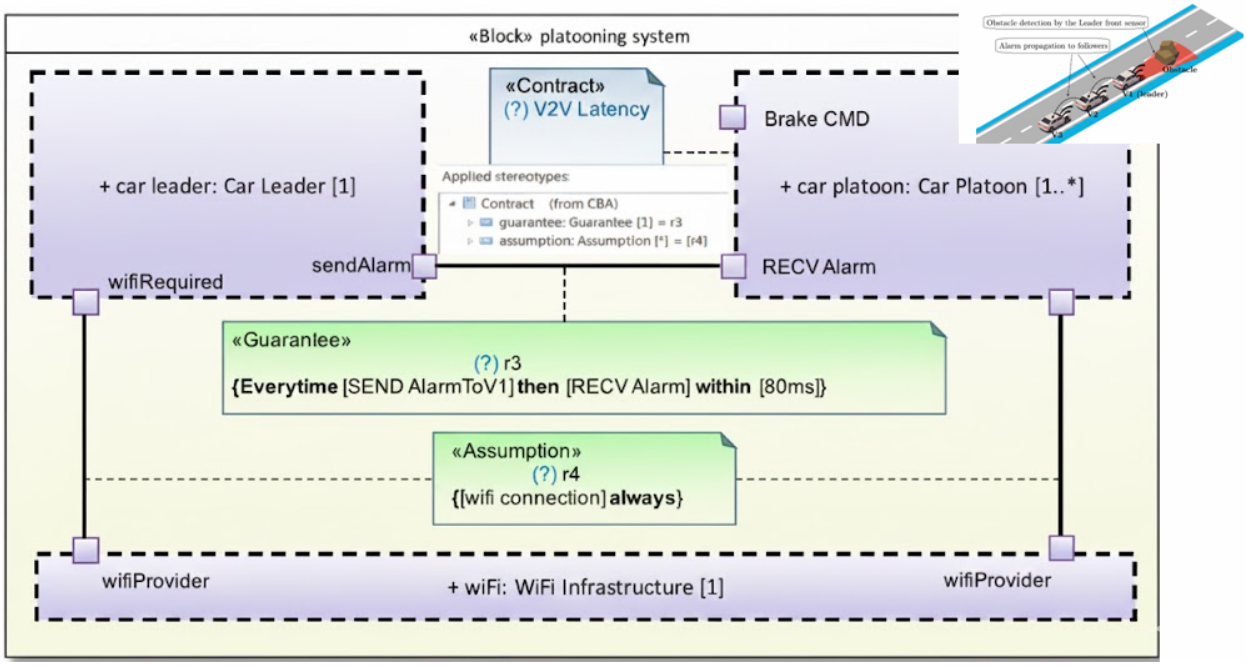


Рис. 3.6. 2D Модель системи колони з контрактами

На моделі три блоки представляють лідера (V1), ТЗ у колоні (V2) та WiFi інфраструктуру. WiFi (представлений вертикальним з'єднувачем) використовується обома ТЗ. V1 може надіслати тривогу V2 (горизонтальний з'єднувач).

Припущення r_3 стосується з'єднувача між WiFi та ТЗ (і підключених до нього портів). Гарантія r_2 стосується з'єднувача між двома ТЗ.

CAT надає гнучкість у розміщенні елементів контракту. У цьому випадку використання з'єднувачів як цільових елементів для специфікації припущення та гарантії підвищує читабельність моделі.

На етапі Arrowhead Framework та вимірювання захищеності для забезпечення довіри між учасниками колони використовуються лише обов'язкові базові сервіси Arrowhead Framework. Автомобільні одиниці з'єднані через багатоперехідний бездротовий ad-hoc протокол (IEEE 802.11g або IEEE 802.11p). Обов'язкові базові сервіси Arrowhead розміщуються в одній автомобільній одиниці (лідері).

Кожна одиниця, що пропонує додатковий сервіс (BrakeSignal), реєструє його в ServiceRegistry лідера. Одиниці завершують налаштування, шукаючи BrakeSignal через сервіс Orchestration та аутентифікуючись щодо нього. Після цього ТЗ починають рух, обмінюючись повідомленнями з сусідами.

Для тестування підходу було розглянуто дві реалізації комунікації:

- простий Веб-Сервіс (SWS): Одиниця-одержувач публікує сервіс, з яким контактує одиниця-відправник.

- веб-Сервіс з Постійним HTTPS (SSE - Server-Sent Events): Одиниця-відправник публікує сервіс, а одиниця-одержувач підключається до нього та підтримує сервіс у режимі очікування. Відправник надсилає повідомлення, коли настає час.

Обидва протоколи були протестовані як у відкритому тексті (HTTP), так і в захищеному вигляді (HTTPS) з використанням Arrowhead токенів. Аутентифікація для захищених випадків забезпечується ієрархією сертифікатів X.509 Arrowhead.

Отже, в розділі продемонстровано методологічний підхід, спрямований на вирішення потенційних конфліктів між властивостями безпеки (safety) та захищеності (security) у системах автономних транспортних засобів, зокрема у конфігурації колони.

Ми розглянули інтегровані властивості safety та security на всіх ключових етапах життєвого циклу проектування системи від текстового

опису вимог до формалізованого подання. Незважаючи на те, що допоміжний інструментарій, задіяний у процесі, ще потребує повної оптимізації, наше дослідження чітко підтверджує, що проблему інтеграції safety-security можна ефективно специфікувати та аналізувати на ранньому етапі проектування.

Висновки до розділу

У третьому розділі сформовано та описано методологію проектування безпечної взаємодії автономних транспортних сутностей, яка поєднує контрактний аналіз, формальні методи та підходи сервісно-орієнтованих архітектур. Доведено, що застосування контрактного підходу дозволяє забезпечити прозорість, формальну верифікованість і перевірюваність поведінкових вимог у складних системах, де взаємодіють автономні агенти з високим рівнем свободи дій.

Проаналізовано можливості використання фреймворку Arrowhead для побудови захищених сервісно-орієнтованих архітектур у транспортній галузі, зокрема для організації процесів аутентифікації, авторизації та управління взаємодією компонентів. Показано, що інтеграція Arrowhead із контрактним моделюванням створює передумови для забезпечення наскрізного контролю відповідності вимогам безпеки на всіх етапах проєктування.

Детально розроблена методологія проєктування безпечних автономних систем включає етапи аналізу вимог, побудови поведінкових моделей, опису контрактів, перевірки коректності взаємодії, інтеграції захищеності та відповідності галузевим стандартам. Реалізація методології на прикладі колони автономних транспортних засобів продемонструвала її прикладну ефективність та можливість створення масштабованої, керованої та перевірюваної системи платонування.

ВИСНОВКИ

Магістерська робота «Програмні моделі та методи безпечної взаємодії автономних транспортних сутностей» присвячена комплексному дослідженню теоретичних засад, технологічних рішень, алгоритмів та методологій, необхідних для створення надійних, функціонально стійких та безпечних систем взаємодії сучасних автономних і підключених транспортних засобів. Виконані наукові дослідження підтвердили актуальність проблематики, що обумовлена зростанням рівня автономності транспортних систем, підвищенням складності дорожнього середовища та потребою у забезпеченні високого рівня безпеки при роботі транспортних агентів у спільному просторі.

У роботі проведено всебічний аналіз предметної області, який дозволив визначити ключові поведінкові, технологічні та інфраструктурні аспекти, що впливають на безпечність автономних транспортних систем. Було встановлено, що технології платонування, архітектури IoT та методи кооперативного обміну даними формують фундамент для побудови сучасних підходів до координації транспортних сутностей та забезпечення їх узгодженої поведінки.

Суттєвим результатом дослідження є розроблення алгоритмів визначення відносного положення транспортних засобів, прогнозування їхніх траєкторій та виявлення потенційно небезпечних ситуацій. Створені V2V- і V2I-алгоритми попередження, зокрема щодо лобового зіткнення, наявності сліпої зони, повільного транспортного засобу, заборони обгону та порушення сигналів світлофора, демонструють практичну значущість відносного позиціонування та кооперативних повідомлень безпеки. Їх інтеграція у транспортні системи сприяє підвищенню рівня ситуаційної обізнаності та зниженню ризику критичних дорожніх подій.

Особливе значення в роботі приділено методології проєктування безпечної взаємодії автономних транспортних сутностей, яка охоплює

використання контрактного підходу, формальних методів перевірки та сервісно-орієнтованих архітектур. Застосування фреймворку Arrowhead та формалізації поведінкових контрактів створює можливість забезпечення узгодженості, передбачуваності та перевірюваності роботи автономних систем на всіх етапах їх життєвого циклу. Розроблена методологія, імплементована на прикладі колони автономних транспортних засобів, підтвердила власну практичну ефективність та здатність забезпечувати високу надійність колективної взаємодії транспортних агентів.

Узагальнюючи результати, слід відзначити, що виконана робота забезпечує суттєвий внесок у розвиток теоретичних і практичних підходів до побудови безпечних автономних транспортних систем.

Отже, результати магістерської роботи створюють методологічний, алгоритмічний та архітектурний базис для подальшого розвитку систем автономного транспорту та підвищення рівня їхньої безпеки у реальних умовах експлуатації. Розроблені підходи можуть бути адаптовані та розширені в рамках комплексних інтелектуальних транспортних систем наступного покоління, сприяючи підвищенню ефективності, надійності та стійкості транспортної інфраструктури загалом.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. A Novel Path Planning Algorithm for Truck Platooning Using V2V Communication. - https://www.mdpi.com/1424-8220/20/24/7022
2. Alam, M., Ferreira, J., & Fonseca, J. A. “Introduction to Intelligent Transportation Systems.” Journal of Advanced Transportation, Wiley, Hoboken, 2016, pp. 1–18.
3. Chen, L., Englund, C. “Cooperative Intersection Management: A Survey.” IEEE Transactions on Intelligent Transportation Systems, IEEE, New York, 2016, pp. 570–586.
4. Elshenawy, A., Elbery, A., Glaser, S., & Elshafie, S. “Autonomous Vehicle Collision Avoidance Using Cooperative V2V Communication.” Transportation Research Part C, Elsevier, Amsterdam, 2020, pp. 298–316.
5. Sepulcre, M., Gozalvez, J., Coll-Perales, B. “Why 6G Will Need Vehicle-to-Everything Communications.” IEEE Vehicular Technology Magazine, IEEE, 2021, pp. 56–64.
6. Nilsson, J., Brännström, M., Coelingh, E., & Fredriksson, J. “Longitudinal and Lateral Control for Cooperative Adaptive Cruise Control.” IEEE Intelligent Transportation Systems Magazine, IEEE, 2017, pp. 22–32.
7. Darbha, S., & Rajagopal, K. “Intelligent Cruise Control Systems and Traffic Flow Stability.” Transportation Research Part C, Elsevier, 1999, pp. 329–352.
8. Taleb, T., Kunz, A. “Machine Type Communications in 5G: System Design and Performance Evaluation.” IEEE Communications Magazine, IEEE, 2017, pp. 52–59.
9. Kyriakidis, M., de Winter, J., & Stanton, N. “Autonomous Vehicle Safety: Critical Review.” Safety Science, Elsevier, 2019, pp. 172–189.
10. Liu, C., & Hedrick, J. “Coordination of Platoon Vehicles at Highway Merges.” American Control Conference (ACC), IEEE, Portland, 2014, pp. 122–127.

11. Van Arem, B., van Driel, C., & Visser, R. “The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics.” IEEE Transactions on Intelligent Transportation Systems, 2006, pp. 429–436.
12. Shladover, S. “Cooperative Driving and Automated Vehicle Systems.” California PATH Research Report, University of California Berkeley, 2018, pp. 1–39.
13. Amanatiadis, A., & Andreadis, I. “A Survey on Vision-Based Vehicle Detection Systems.” Journal of Intelligent Transportation Systems, Taylor & Francis, 2015, pp. 1–14.
14. Singh, S., et al. “V2X-Based Collision Warning System for Autonomous Driving.” IEEE Access, IEEE, 2020, pp. 8750–8762.
15. Fernandes, P., & Nunes, U. “Platooning with IVC-Enabled Autonomous Vehicles.” IEEE Transactions on Intelligent Transportation Systems, 2012, pp. 829–843.
16. Amoozadeh, M., et al. “Security Vulnerabilities of Connected Vehicle Streams.” ACM Workshop on Automotive Cybersecurity, ACM, New York, 2015, pp. 1–12.
17. Lin, P., & Chen, Y. “Simulation-Based Evaluation of V2I Safety Applications.” Transportation Research Part C, Elsevier, 2018, pp. 287–302.
18. Ploeg, J., et al. “Design and Experimental Evaluation of Cooperative Adaptive Cruise Control.” IEEE International Conference on Intelligent Transportation Systems, IEEE, 2011, pp. 260–265.
19. Zhou, Q., Qu, X., & Song, Z. “Modeling Autonomous Vehicle Behavior under V2V.” Transportation Research Part B, Elsevier, 2020, pp. 1–24.
20. Kuutti, S., et al. “A Survey of Deep Learning for Autonomous Vehicle Control.” IEEE Transactions on Intelligent Transportation Systems, 2021, pp. 294–310.
21. Liu, R., & Gayah, V. “Traffic Stability Analysis of Mixed Autonomy Systems.” Transportation Research Part C, Elsevier, 2020, pp. 103–119.

22. Abbas, M., et al. “Emerging Trends in Cooperative ITS.” Journal of Advanced Transportation, Wiley, 2019, pp. 1–19.
23. Ge, J. I., & Orosz, G. “Connected Cruise Control with V2V.” IEEE Transactions on Control Systems Technology, 2014, pp. 113–120.
24. Behere, S., & Törngren, M. “Architecture Framework for Autonomous Systems.” SAE Technical Paper Series, SAE International, Detroit, 2016, pp. 1–12.
25. Hossain, M., & Chowdhury, M. “Framework for V2X Safety Applications.” Accident Analysis & Prevention, Elsevier, 2017, pp. 507–524.
26. Lee, J., et al. “Cooperative Merging Algorithms for Autonomous Vehicles.” IEEE Robotics and Automation Letters, IEEE, 2018, pp. 202–209.
27. Wang, Z., & Chan, C. “Autonomous Vehicle Emergency Handling Modeling.” Transportation Research Part C, 2019, pp. 424–441.
28. Talebpour, A., & Mahmassani, H. “Influence of Autonomous Vehicles on Traffic Flow.” Transportation Research Part C, 2016, pp. 162–180.
29. Kim, K., & Kumar, P. “V2X-Based Lane Change Strategies.” IEEE Transactions on Vehicular Technology, 2020, pp. 553–568.
30. Saeed, A., et al. “Traffic Prediction for Smart Transportation Systems.” IEEE Access, 2019, pp. 1638–1652.
31. Lopes, A., et al. “Formal Verification of Autonomous Driving Functions.” International Conference on Computer Safety, Reliability and Security, Springer, Berlin, 2020, pp. 155–170.
32. El Ghzaoui, M., et al. “Security Challenges in V2X.” Vehicular Communications, Elsevier, 2021, pp. 1–18.
33. Raza, S., et al. “IoT Security Applied to ITS.” IEEE Communications Surveys & Tutorials, 2018, pp. 302–317.
34. Janson, T., Schwung, A., et al. “Simulation Methods for Autonomous Vehicles.” IEEE International Conference on Intelligent Transportation Systems, 2019, pp. 144–151.

35. Gurney, J. “Liability for Autonomous Vehicle Decisions.” Minnesota Journal of Law, Science & Technology, Univ. of Minnesota Press, 2016, pp. 1–56.
36. Kesting, A., Treiber, M., & Helbing, D. “Enhanced Intelligent Driver Model.” Philosophical Transactions of the Royal Society A, Royal Society Publishing, London, 2010, pp. 4541–4573.
37. Shalev-Shwartz, S., et al. “On a Formal Model of Safe and Scalable Self-Driving Cars.” arXiv.org, Cornell University, New York, 2017, pp. 1–45.
38. Rajamani, R. “Vehicle Dynamics and Control.” Springer Tracts in Mechanical Engineering, Springer, Berlin, 2012, pp. 1–272.
39. Bianchi, V., et al. “Sensor Fusion for Autonomous Driving.” Sensors, MDPI, Basel, 2019, pp. 1–19.
40. Lee, E., et al. “V2X-Based Cooperative Perception.” IEEE Transactions on Vehicular Technology, 2021, pp. 34–47.
41. Göhring, D., Wang, M., Schnürmacher, M., & Ganjineh, T. “Autonomous Driving in Urban Environments: Vision-Based Control.” IEEE International Conference on Intelligent Robots and Systems, IEEE, 2011, pp. 94–99.