Довідка
про відсутність плагіату у випускній кваліфікаційній роботі

Міністерство освіти і науки України
Український державний університет науки та технологій

Кафедра «Комп'ютерні інформаційні технології»

ДОВІДКА

За результатами перевірки випускної кваліфікаційної роботи здобувача вищої освіти *Мерзлого Олексія Дмитровича*
(прізвище, ім'я, по батькові)
на тему: *Дослідження та оптимізація автомобільних транспортних потоків засобами імітаційного моделювання*
в роботі не виявлено порушень академічної доброчесності.

Керівник ВКР _____ Олександра ГОРБОВА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Український державний університет науки і технологій

Кафедра Комп'ютерні інформаційні технології

«ДО ЗАХИСТУ»

Завідувач кафедри

_____ /Вадим ГОРЯЧКІН/

«20» 12 2021 р.

ДИПЛОМНА РОБОТА

на здобуття освітнього ступеня «магістр»

Галузь знань **12 Інформаційні технології**

Спеціальність **121 Інженерія програмного забезпечення**

Тема **Дослідження та оптимізація автомобільних транспортних потоків засобами імітаційного моделювання**

Theme **Research of traffic flows by means of simulation modelling**

Керівник дипломної роботи доц. _____ Олександра ГОРБОВА

Нормоконтролер доц. _____ Олена КУРОП'ЯТНИК

Студент групи ПЗ2021 _____ Олексій МЕРЗЛИЙ

Student Oleksii MERZLYI

Дніпро – 2021

Дніпровський національний університет залізничного транспорту
імені академіка В. Лазаряна

Факультет Комп'ютерних технологій і систем
Кафедра Комп'ютерні інформаційні технології
Спеціальність Інженерія програмного забезпечення

«ЗАТВЕРДЖУЮ»
Завідувач кафедри
_____ проф. В.І. Шинкаренко
(підпис)
«25» листопада 2021р.

ЗАВДАННЯ

до дипломної роботи на здобуття ОС магістр
(освітній ступінь)

студента групи ПЗ2021 О.Д. Мерзлого
(номер групи) (ПІБ)

1 Тема дипломної роботи: Дослідження транспортних потоків засобами імітаційного моделювання
затверджена наказом по університету від «18» листопада 2021 р. №690.

2 Термін подання студентом закінченої роботи 3 грудня 2021р

3 Вихідні дані до дипломної роботи час роботи світлофорів, початкова інтенсивність транспорту і пішоходів

4 Зміст пояснювальної записки (перелік питань до розробки) аналіз та дослідження сучасного стану проблеми автомобільних доріг, огляд існуючих рішень вирішення проблеми авто заторів, обгрунтування створення імітаційної моделі, проектування і створення інструментального забезпечення, проведення експериментів і дослідження автомобільних потоків, охорона праці та безпеки в надзвичайних ситуаціях

5 Перелік демонстраційного матеріалу доповідь, презентація, демонстраційне відео

6. Консультанти (з назвами розділів):

Розділ	Консультант	Підпис, дата	
		завдання видав	завдання прийняв
Техніко-економічні розрахунки	Микола ГНЕННИЙ	18.10.21	25.10.21
Охорона праці	Олег САБЛІН	26.10.21	24.11.21

КАЛЕНДАРНИЙ ПЛАН

№ пор.	Назва розділів дипломної роботи	Термін виконання розділів роботи	Примітка
1	Вступ	01.09.21	
2	Аналіз сучасного стану дослідження проблеми за науковими літературними джерелами	02.09.21 – 15.09.21	від 70 джерел
3	Аналіз сучасного стану програмно-апаратного забезпечення, яке потребує вдосконалення для вирішення проблем дослідження	16.09.21 – 10.10.21	
4	Постановка задачі, технічне завдання	11.10.21 – 17.10.21	30%
5	Техніко-економічні показники	18.10.21 – 19.10.21	
6	Розробка інструментальних засобів дослідження	20.10.21 – 07.11.21	
7	Виконання досліджень	08.11.21 – 14.11.21	60%
8	Оформлення тез доповідей	15.11.21 – 18.11.21	
9	Оформлення статті у фаховий журнал	19.11.21 – 22.11.21	
10	Оформлення пояснювальної записки	23.11.21 – 28.11.21	
11	Розробка демонстраційних матеріалів	29.11.21 – 05.12.21	100%

Дата видачі завдання « 18 » листопада 2020 р.

Керівник дипломної роботи _____ О.В. Горбова
(підпис) (ПІБ)

Завдання прийняв до виконання _____ О.Д. Мерзлий
(підпис) (ПІБ)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту містить 84 сторінки, 35 рис., 11 таблиць, 6 додатків, 27 джерел.

Об'єкт дослідження – імітаційна модель транспортних потоків.

Мета дослідження імітаційного моделювання полягає у відтворенні поведінки досліджуваної системи на основі результатів аналізу найбільш суттєвих взаємозв'язків між її елементами або іншими словами – розробці симулятора досліджуваної предметної області для проведення різних експериментів.

Методи розв’язання та аналізу результатів здійснені з використанням методів системного аналізу, імітаційного моделювання, теорії графів та об’єктно-орієнтованого аналізу, математичної статистики, кореляційного аналізу, теорії ймовірностей, дискретно – подієвого моделювання, агентного моделювання та системна динаміка для використання в розробці імітаційної моделі.

Наукова новизна. Всі результати, отримані в роботі, є новими і актуальними. Зокрема, запропонована модель агента-учасника дорожнього руху, що відображає основні аспекти поведінки водіїв. Також виконана програмна реалізація моделі агента, придатна для дослідження транспортних систем. Усі дослідження виконуються на основі реальної транспортної системи.

Наукові результати, які отримані в дипломній роботі, а також методи можуть бути використані при розробленні та/або удосконаленні транспортної системи України. Математична модель та розрахункові дані можуть бути застосовані при розрахунках пропускної спроможності системи, виставлення оптимального часу роботи світлофорів, розвантаження транспортної системи, уникнення переповнених маршруток шляхом коректування інтервалу між ними.

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ

ІМ – імітаційна модель;

ПП – програмний продукт;

ЕОМ – електрона обчислювальна машина;

ПК – персональний комп’ютер;

НПАОП – нормативно-правові акти з охорони праці;

ДСанПіН – державні санітарні правила та норми.

ВСТУП

Актуальність теми.

Основний розвиток будь-якої країни напряму залежить від стану транспортної системи. Розвиток цієї системи веде до позитивних економічних і соціальних наслідків. Транспорт покращує умови проживання, сприяє переміщенню людини, ефективній роботі підприємств, організацій. Автомобільні перевезення стали дуже важливою частиною процесу перевезення і взаємодії з усіма видами транспорту, виконуючи перевезення вантажів і пасажирів.

Предмет дослідження.

Через збільшення автотранспортних потоків в транспортній мережі актуальною є проблема їх раціональної організації. Однак з урахуванням впливу різних чинників, таких як завантаженість ділянки дороги, стану дороги, дана задача не може бути вирішена за допомогою аналітичних моделей, заснованих на графових моделях, або експериментами на реальній транспортній мережі. Предметом дослідження проблеми виступає імітаційна модель транспортних потоків.

Мета і задачі дослідження.

Мета дослідження імітаційного моделювання полягає у відтворенні поведінки досліджуваної системи на основі результатів аналізу найбільш суттєвих взаємозв'язків між її елементами або іншими словами – розробці симулятора досліджуваної предметної області для проведення різних експериментів.

Метою вирішення поставленої задачі є розробка безпечної моделі дорожнього руху методом імітаційного моделювання координованих транспортних потоків в міський дорожньої мережі та розробка необхідної для досягнення поставленої мети системи комп'ютерного моделювання.

Математичний опис впливу різних характеристик середовища, в якому рухаються транспортні потоки, взаємний вплив транспортних потоків між собою в конфліктних ситуаціях, пошук характеристик і методики її обчислення, як показника стійкості параметрів управління координованого транспортного потоку до випадкових коливань інтенсивності є основними інструментами у досягненні поставленої задачі.

Методи дослідження.

Важливим засобом для розв'язання багатьох економічних завдань, проведення аналітичного дослідження є створення моделі, і сам процес моделювання. Модель – це образне відображення реального об'єкта, умовний об'єкт дослідження. Іншими словами це матеріальне чи образне відображення реального об'єкта або процесу його функціонування в якомусь конкретному середовищі. Метод моделювання – це створення та конструювання моделі на основі попереднього вивчення об'єкта. Визначення його найбільш суттєвих характеристик, експериментальний і теоретичний аналіз створеної моделі, а також необхідне коригування властивостей на підставі одержаної інформації.

Імітаційне моделювання дозволяє імітувати поведінку системи в часі. Причому перевагою є те, що часом в моделі можна управляти: уповільнювати у випадку з швидкоплинними процесами і прискорювати для моделювання систем з повільною несталістю. Імітаційне моделювання дає можливість розглянути поведінку тих об'єктів, реальні експерименти з якими є дуже витратними, неможливими або небезпечними. З популяризацією використання комп'ютерів, створення складних і унікальних процесів супроводжується комп'ютерним тривимірним імітаційним моделюванням. Ця точна і відносно швидка технологія дозволяє накопичити всі необхідні знання, обладнання та напівфабрикати для майбутньої системи до початку впровадження у технологічний процес. Комп'ютерне 3D моделювання тепер не рідкість навіть для невеликих компаній [1].

Для пошуку ефективних стратегій управління транспортними потоками в мегаполісі, оптимальних рішень з проектування вулично-дорожньої мережі та організації дорожнього руху необхідно враховувати широкий спектр характеристик транспортного потоку, закономірності впливу зовнішніх і внутрішніх факторів на динамічні характеристики змішаного транспортного потоку. Застосування моделювання і створення адекватної моделі транспортного потоку є актуальним завданням в процесі організації та управління дорожнім рухом.

Методика досліджень дозволить створити комплексний підхід до вирішення задач наведеного типу та буде міститиме симбіоз теоретичних та експериментальних досліджень.

В моделюванні моделі буде використана системна динаміка, дискретно-подієвого моделювання (процесно-орієнтоване) та агентне моделювання.

Таким чином, розв’язок задачі полягає в створенні методу імітаційної моделі координованих транспортних потоків і реалізації її в вигляді як програмного модуля, так і реалізації моделі у вигляді спеціалізованого програмного забезпечення для розрахунку параметрів управління дорожнім рухом.

Наукова новизна.

Всі результати, отримані в роботі, є новими і актуальними. Зокрема, запропонована модель агента-учасника дорожнього руху, що відображає основні аспекти поведінки водіїв. Також виконана програмна реалізація моделі агента, придатна для дослідження транспортних систем.

Наукова новизна полягає у:

– у створенні загальної методології при моделюванні процесів за допомогою методу поетапного моделювання;

– удосконаленні формалізації методу агентного моделювання;

– удосконалення транспортної системи на основі проведених експериментів.

Практичне значення

Результати роботи покладені в основу системи імітаційного моделювання транспортних потоків, що дозволяє аналізувати властивості існуючих і проектованих транспортних вузлів. Система реалізована у вигляді програмного комплексу, який може бути використаний в установах державного управління, проектних організаціях і консалтингових компаніях, що займаються проектуванням і реорганізацією схем дорожнього руху. Запропонована модель агента може бути використана в складі більш складних імітаційних моделей організаційно-технічних систем.

Апробація результатів дослідження та публікації.

Апробація результатів дослідження та публікації. Основні положення магістерської роботи доповідалися та були схвалені на XIV міжнародній науково-практичній конференції «Сучасні інформаційні та комунікаційні технології на транспорті, в промисловості та освіті», яка відбулася в Дніпровському національному університеті залізничного транспорту імені академіка В. Лазаряна (ДНУЗТ) 15 – 16 грудня 2020 року, див додаток В.

На основні положень магістерської роботи було створено тези доповіді, котрі були схвалені на 81 міжнародній науково-практичній конференції "Проблеми та перспективи розвитку залізничного транспорту", яка відбулася в Дніпровському національному університеті залізничного транспорту імені академіка В. Лазаряна (ДНУЗТ) 22 – 23 квітня 2021 року, та на 81 всеукраїнській науково-технічній конференції молодих учених, магістрантів та студентів «Наука і сталий розвиток транспорту», яка відбулася в Дніпровському національному університеті залізничного транспорту імені академіка В. Лазаряна (ДНУЗТ) 28 жовтня 2021 року, див додаток Г, Д.

Опубліковано наукову статтю «ДОСЛІДЖЕННЯ ТРАНСПОРТНИХ ПОТОКІВ ЗАСОБАМИ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ», «Вісник Дніпропетровського національного університету залізничного транспорту» (подана до друку), див додаток Е.

1 АНАЛІЗ ТА ДОСЛІДЖЕННЯ СУЧАСНОГО СТАНУ ПРОБЛЕМИ АВТОМОБІЛЬНИХ ДОРІГ

1.1 Проблема заторів автодоріг України та шляхи її вирішення

Україна має досить розвинуту мережу автодоріг загального користування протяжністю 172,4 тис. км, в тому числі 164,1 тис. км з твердим покриттям. За оцінками Міністерства інфраструктури, з числа міжнародних автотрас (близько 8200 км) в хорошому стані знаходиться 24%, в задовільному – ще 65%. Для доріг національного значення (4800 км) ці цифри складають 21% і 68% відповідно, а ось для регіональних (9800 км) – лише 10% і 50%. Очевидно, вітчизняні автомагістралі потребують серйозної модернізації [19].

Провівши аналіз, можна побачити швидкий темп зростання автотранспорту. На момент 1900 року в усьому світі нараховувалось близько 12 тисяч автомобілів. 1920 році – 10922 тисяч, 1950 року – 70388, 1957 року – 102827 тисяч. На даний час більше 1,5 млрд авто.

На момент 2014 року в усьому світі нараховувалось на кожних 1000 чоловік у світі припадало 118 легкових авто. У деяких країнах показник склав 2–10 автомобілів на 1000 жителів, але в деяких країнах цей показник перевищує 500 автомобілів на 1000 мешканців. Це нормальний показник для багатьох європейських країн. Рівень автомобілізації (250 – 300 автомобілів) дані країни досягли ще в 60–70-х роках минулого століття. Україна в порівнянні з іншими країнами має показник у чотири рази менший. Треба відмітити, що на 1000 мешканців в Україні припадає 191 автомобіль, а всього нараховується 8 млн. 700 тисяч автомобілів, що ставить Україну на 69 місце по рівню автомобілізації серед 145 країн, показано на табл. 1.1 [2].

Таблиця 1.1 – Рівень автомобілізації в деяких країнах світу [2]

Країна	Населення, тис. чол.	Кількість легкових автомобілів, тис. шт.	Кількість комерційних автомобілів, тис. шт.	Загальна кількість автомобілів, тис. шт.	Забезпечення авто на 1000 мешканців
У світі	7043106	833342	309888	1143231	162
Бруней	412	120,0	240,0	360,0	873
Пуерто-Ріко	3652	2700,0	290,0	2990,0	819
США	313874	120901,6	130595,5	251497,1	801
Ісландія	321	210,0	33,0	243,0	758
Люксембург	531	350,0	39,0	389,0	733
Австралія	22724	13000,0	3 436,0	16 436,0	723
Кувейт	3250	850,0	1 500,0	2350,0	723
Мальта	419	250,0	47,0	297,0	708
Італія	59540	37078,0	4 922,0	42000 ,0	705
Фінляндія	5414	3037,0	530,0	3567,0	659
Нова Зеландія	4433	2425,0	459,0	2884,0	651
Литва	2988	1753,0	152,0	1905,0	638
Канада	34754	20750,0	995,0	21745,0	626
Норвегія	5019	2433,0	571,0	3004,0	599

Процес автомобілізації найбільших міст Західної Європи, що почався в 50-ті роки минулого сторіччя, проходив практично за однією закономірністю для усіх країн: лінійний ріст кількості автомобілів до рівня 300-350 авт./1000 жителів, потім уповільнення зростання та стабілізація при 550±50 авт./1000 жителів [2].

У цілому в розвинутих країнах проходить зниження темпів росту автомобільного парку, який в останні роки складає 1 – 2 %. У Північній Америці вже достатньо довго спостерігається ріст чисельності сімей, які мають два автомобілі і більше. У США їх доля вже досягла 58 %. У Західній Європі ріст чисельності середнього класу і його благополуччя супроводжується аналогічними процесами. Так, наприклад, в Іль-де-Франс (агломерація Парижу) у 1991 році 75 % домогосподарств мали один автомобіль, а 20 % – два і більше. Такі тенденції спостерігалися і у Великобританії, наприклад, за період 1992–2000 роки відбулася зміна: кількість домогосподарств без автомобілів знизилась з 32 до 26 %; доля господарств з одним автомобілем залишилася сталою – 45 %, а доля домогосподарств з двома автомобілями і більше збільшилась із 24 % до 27 %; середня кількість автомобілів, яка приходиться на одне домогосподарство збільшилась з 0,86 до 1,04. Особливо цікавою для нас є динаміка зростання автомобільного парку в містах країн Східної Європи та прогнозовані показники, що приймаються фахівцями цих країн. Темпи зростання автомобільного парку в містах Східної Європи, звичайно, вищий, ніж у містах Західної Європи. Так, зростання парку Парижа становило 1 %, Варшави – 7 %, а зростання парку автомобільного транспорту для України в середньому складає 4,8 % , хоча у 2008 р. цей показник складав – 6,9 %(рис. 1.1). Слід зазначити, що у країн Східної Європи збільшення частки поїздок, що здійснюються з використанням легкового автомобіля, виявилося менше, ніж зростання автомобільного парку. Україна проходить той самий етап розвитку і формування автомобільного парку, який 30 – 50 років тому пройшли розвинуті країни світу [2].

Рисунок 1.1 – Зростання автомобільного парку України в період 2000–2014 рр., у %

В Україні інтенсивний ріст парку транспортних засобів розпочався в 1990-х роках і стабільно продовжується. Цей процес прямий і безпосередньо пов'язаний із набуттям економічної свободи громадянами України, свободи щодо вибору місця проживання і прикладення праці. Це в свою чергу призвело до швидкого росту автомобілізації, що підтверджує бажання українців до підвищення мобільності та якості життя. Автомобіль став не лише засобом переміщення, а й підтвердженням статусу життя, символом благополуччя і успішності в житті [2].

Але збільшення кількості транспорту у світі залежить не тільки від індивідуального транспорту. Через збільшення населення і потреб, модернізації транспорту і підвищення рівня життя, з'явилася потреба в комерційних перевезеннях.

Основні фактори, що впливають на ціну перевезення, тим саме на ціну самого товару – обсяг і вага вантажу (питома вартість перевезення великих партій завжди нижче, як і в будь-яких інших випадках з оптом). Деякі товари

потребують особливих умов перевезення і зберігання. Чим довше товар буде в дорозі тим гірше для перевізника, і в купі ці всі фактори будуть впливати на кінцеву ціну товару.

Має значення і вид використовуваного авто, а також регіон: в великих логістичних та промислових вузлах перевезення для вантажовідправників традиційно дорожче. Це в першу чергу Київ, Дніпро, Одеса, Львів і Рівне.

Лише у 2020 році обсяг перевезення вантажів автомобільним транспортом склав близько 242,7 мільйонів тон. Так, як Україна територіально розташована майже у самому центрі автомобільних перевезень і основні дороги між державами проходять через Україну, доцільно організувати максимально швидкі шляхи для подолання відстані між точкою А і Б. А через збільшення транспорту, з'явилася дужа важлива проблема – це затори.

Чим більше місто і його населення, з тим більшою кількістю проблем йому доводиться зустрічатися щодня. А бажання кожного жителя мати особистий транспорт створює одну величезну проблему – трафік, з яким не кожні дороги можуть впоратися.

Відсутність достатньої кількості паркувальних місць, збільшення часу, проведеного в дорозі, висока ймовірність ДТП, зниження безпеки для пішоходів і, звичайно ж, пробки, які збільшують витрату палива і рівень забруднення навколишнього середовища.

Основні проблеми функціонування транспортних систем у містах:

– зростання рівня автомобілізації населення;
– збільшення інтенсивності використання індивідуально транспорту;
– зниження ефективності міського пасажирського транспорту;
– збільшення потреби жителів міста в переміщеннях;
– диспропорція між рівнем автомобілізації і темпами дорожнього будівництва;
– містобудівні проблеми розвитку міської території.

Особливо проблематичним є високі прямі і непрямі економічні витрати, в тому числі високі витрати, пов'язані з логістикою, аваріями, заторами та забрудненням повітря. Від цього страждають конкурентоспроможність України та її привабливість для інвесторів. Тому головним завданням є модернізація концепцій мобільності в українських містах.

Затори не тільки порушують наші плани. Вони – основне джерело забруднення повітря, що негативно відбивається на здоров'ї кожного. За даними Центру аналізу ризиків Гарвардського університету, пробки в 83 найбільших містах Сполучених Штатів стали причиною понад 2200 випадків передчасної смерті в 2010 році і збільшили витрати на практику охорони здоров'я на 18 млрд доларів.

Але ж є ще економічні витрати, пов'язані з втраченими в пробках годинами (робочими і неробочими) і з затримкою доставок. Водії в 10 американських містах з самим перевантаженим рухом щорічно проводять в пробках близько 42 годин, несучи збиток на більш ніж 121 млрд доларів.

У період автомобільного буму 1960-х містобудівники бачили тільки одне, здавалося б, очевидне рішення: будувати більше доріг і робити їх більш широкими. Але це не спрацювало. Чим більше доріг будувалося, тим більше автомобілів на них з'являлося. Наприклад, дослідження, проведене в Каліфорнії в 1997 році, показало, що як би не збільшувалася пропускна здатність доріг, знадобиться всього п'ять років, щоб рівень їх завантаження досяг 90%.

Проблема заторів не вирішується шляхом будівництва нових доріг і тунелів. Необхідно визначити пріоритети. Це означає, що громадський транспорт повинен отримати пріоритет завдяки окремим рейсовими шляхами і виділеним автобусним смугах, а також на світлофорах.

1.2 Огляд існуючих рішень для вирішення проблеми авто заторів

1.2.1 Психологічній аспект вирішення проблеми

Шляхів вирішення проблем заторів дуже багато, від соціальних, психологічних, до розробки моделей поведінки і розуміння основної причини появи заторів.

При першому аналізі проблеми, можливо зробити перші висновки, що є головною причиною заторів. Це, здавалося б, очевидно – вузькі місця на дорогах і скупчення автомобілів. Тим не менш, деякі пробки, на перший погляд виникають спонтанно, можуть бути викликані так званим ефектом «метелика», коли всього один водій раптом робить нічим не виправданий маневр – наприклад, різко перебудовується в інший ряд. Автомобілям, які їдуть за ним, доводиться різко гальмувати, що викликає ланцюгову реакцію, миттєво приводить до утворення пробки на трасі.

Дослідження, проведене Технологічному університеті штату Джорджія, бачить проблему в поєднанні агресивних водіїв (їдуть занадто швидко і занадто близько до машини попереду) і боязких водіїв, які дотримуються занадто велику дистанцію. І перші, і другі змушують інших водіїв гальмувати, в результаті чого рух сповільнюється ще більше, аж до повної зупинки.

Вчені намагалися змоделювати транспортні потоки, шукаючи аналогії в тому, як течуть рідина або газ, переміщаються птиці або лижники. Однак, вважає Габор Орос з Університету штату Мічиган, «хоча подібні аналогії допомагають зрозуміти деякі закономірності, стає все більш і більш очевидним, що транспортні потоки не схожі ні на які інші потоки в ньютонівському всесвіті».

1.2.2 Вирішення проблеми за допомогою смарт технологій

На даний час покладаються великі надії на смарт технології і те, як саме вони можуть вирішити і вже вирішують проблеми завантаженого трафіку.

Оскільки велика частина машин на дорозі – це ті, чиї водії шукають місця для паркування, в деяких містах намагаються управляти дорожнім рухом, використовуючи датчики для визначення того, зайнято конкретне місце на вулиці або на автостоянці або вільно.

Якщо зв'язати ці розумні датчики з системою, яка швидко і ефективно направляє водіїв на вільні паркувальні місця, є надія, що це знизить рівень завантаженості доріг. Першим такі датчики випробував Сан-Франциско. Не відстає від сусіда і Лос-Анджелес. Сьогодні обидва міста обслуговує ACS, дочірня компанія корпорації Xerox.

Є й інші ідеї. Ізраїльський стартап Waze (див. рис 1.2), в минулому році куплений компанією Google, для більш раціонального використання перевантажених доріг застосовує краудсорсінг, допомога онлайнового співтовариства.

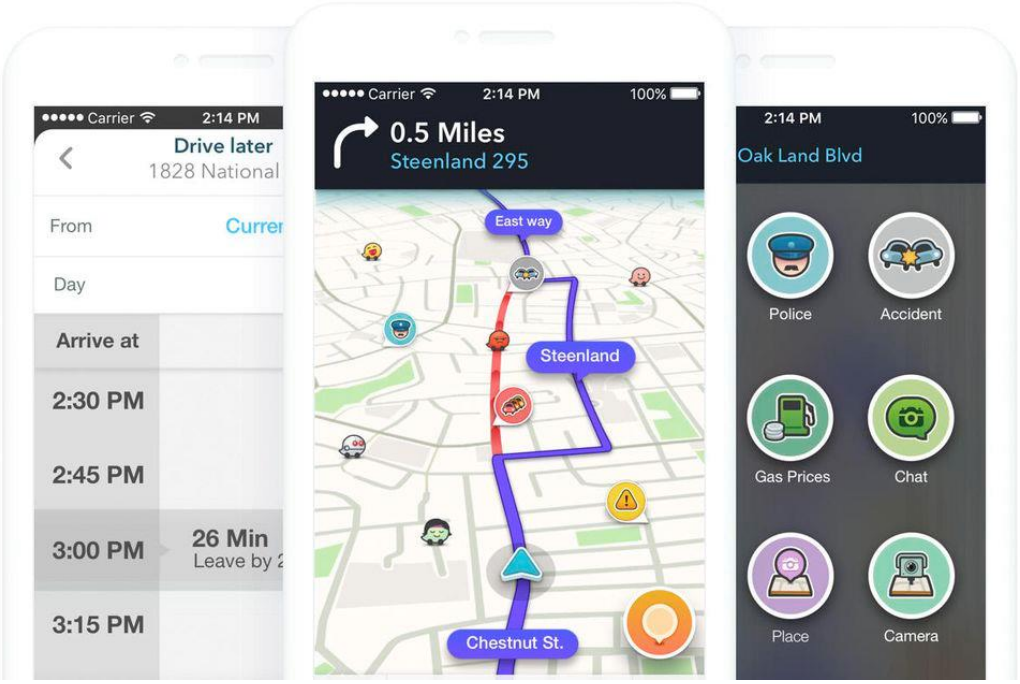


Рисунок 1.2 – Навігаційний додаток Waze

Навігаційний додаток Waze для смартфонів працює як соціальна мережа для водіїв. Воно пропонує своїм користувачам не тільки мобільний карту, але і

отримується від інших водіїв в режимі реального часу оновлену інформацію про пробки, дорожні роботи і аваріях.

Waze використовує провулки, польові дороги і порожні «кишені», прокладаючи маршрут через найменш перевантажені ділянки дороги і, таким чином, мінімізуючи трафік на головній магістралі.

Ще одним способом позбутися від пробок, як вважають деякі, може стати автоматизація самого процесу водіння. Нещодавно компанія Google представила свій власний прототип повністю автономного самоврядного автомобіля, якому не потрібен водій.

1.2.3 Імітаційне моделювання, як метод вирішення проблеми заторів

Рішень питання заторів багато, але кожне з них вимагає часу і моделювання того чи іншого процесу. Наприклад вирішення питання від компанії Ford. Імітаційне моделювання дозволяє розробникам тестувати безпілотні автомобілі в контрольованому середовищі.

Дослідження системної продуктивності безпілотних автомобілів в реальних умовах – дуже дорогий спосіб з точки зору часу і ресурсів або навіть практично неможливий. Тому імітаційне моделювання в даній ситуації є ефективною альтернативою.

За допомогою добре розроблених імітаційних моделей можна розміщувати віртуальні транспортні засоби в віртуальних середовищах, підключати віртуальні датчики, додавати віртуальні вимірювальні прилади і мільйони разів виконувати широкий спектр тестів. Такі моделі дозволяють вивчити, як безпілотні автомобілі можуть працювати при самих різних сценаріях. Дані цих випробувань дають уявлення про те, як працюють безпілотні автомобілі Ford, перш ніж їх відправлять в реальний світ [3].

Також імітаційні моделі можуть бути використані для імітації досвіду пасажирів. У той час як дочірня компанія Ford Argo AI очолює розробку систем безпілотного водіння, Quantum Signal тісно співпрацює з Ford, щоб вивчити, як

ця система буде взаємодіяти з платформою автомобіля і пасажирами. Quantum Signal допомагає Ford створювати імітаційні моделі, за допомогою яких можна аналізувати якість їзди в безпілотному автомобілі. Мета – забезпечити пасажирам плавне і комфортне водіння [3].

Одне з можливих рішень – показати пасажирам на екрані, що робить транспортний засіб. Використовуючи імітаційне моделювання, можна швидко досліджувати численні способи візуалізації інформації, що надходить від датчиків автомобіля, і допомогти команді Ford розробити інтерфейси, які будуть легко і доступно повідомляти пасажиру інформацію про те, що відбувається в будь-якій точці під час поїздки. Можна показати клієнтам поточну траєкторію руху автомобіля по вулиці, а також наявність світлофорів, пішоходів та інших транспортних засобів на дорозі. Зрештою, якщо ви знаєте, що безпілотне транспортний засіб бачить попереду пішохода, що переходить дорогу, або що воно сповільнюється перед поворотом направо, то ви, швидше за все, будете почувати себе комфортніше [3].

Імітаційне моделювання може бути використано для підвищення якості поїздки, і це одна з основних причин, чому воно буде грати ключову роль в розробці майбутніх безпілотних автомобілів [3].

Повертаючись до питання появи і раціонального рішення питання появи транспортних заторів засобами імітаційного моделювання. На сьогоднішній день вулично-дорожні мережі являють собою найскладніші споруди. Вони повинні забезпечувати якомога більш безпечний рух потоків автомобілів високою швидкістю. Вулично-дорожні мережі є дуже дорогим елементом міської інфраструктури, їх проектування відносять до числа найбільш складних питань теорії транспортної планування міст. Багато факторів можуть вливати на функціонування транспортної системи.

Вирішення проблем заторів засобами імітаційного моделювання на основі реальної автотранспортної системи допоможе зрозуміти суть появи заторів, допоможе зібрати необхідну статистику, для подальшої оптимізації

системи, та для обґрунтування щодо доцільності використання такої системи в реальному житті.

1.3 Огляд аналогів на програмних комплексів

За останні два роки стрімко набрало оберти імітаційне моделювання. З’явилось багато програмних комплексів. Нижче наведено основні і більш популярні.

Середовище моделювання Aimsun призначена для моделювання транспортних потоків.

Розробник: компанія TSS – Transport Simulation Systems, S. L, Іспанія.

Сайт: www.aimsun.com.

Система Actor Pilgrim – система імітаційного моделювання тимчасової, просторової і фінансової динаміки економічних процесів. Система дозволяє працювати з багатошаровими імітаційними моделями. Підтримувані види (технології) моделювання: дискретне і дискретно-безперервне, одночасна реалізація тимчасової, просторової і фінансової динаміки [4].

Розробники: А. А. Ємельянов, Н. З. Ємельянова (Москва).

AGNES (AGent NEtwork Simulator) – система імітаційного моделювання великих систем з дискретними подіями. Система AGNES є кросплатформовою, можливий розподілений запуск (на локальній мережі або обчислювальному кластері). Область застосування: обчислення на супер ЕОМ, локальні комп'ютерні мережі, бездротові сенсорні мережі. AGNES поширюється безкоштовно за ліцензією LGPL.

Розробник системи: Д. І. Подкоритов, Інститут обчислювальної математики і математичної геофізики СО РАН.

Система моделювання AnyLogic підтримує три підходи до створення імітаційних моделей: процесно-орієнтований (дискретноподієвий), системно-динамічний і агентний, а також будь-яку їх комбінацію. Графічний інтерфейс AnyLogic, інструменти та бібліотеки дозволяють швидко створювати моделі

для широкого спектра задач – від моделювання виробництва, логістики, бізнес-процесів до стратегічних моделей розвитку компанії і ринків. AnyLogic став корпоративним стандартом на бізнес-моделювання в багатьох транснаціональних компаніях, широко використовується в освіті [4]. Численні приклади імітаційних моделей, побудованих засобами AnyLogic, наведені на сайті www.runthemodel.com [5].

Розробник: The AnyLogic Company. Сайт: www.anylogic.ru.

Arena – система дискретного моделювання. Сфера основних додатків системи – імітаційне моделювання виробничих технологічних процесів і операцій, складський облік, банківська діяльність, оптимізація обслуговування клієнтів в сфері послуг, транспортні завдання. Arena забезпечена зручним об'єктно-орієнтованим інтерфейсом і володіє можливостями адаптації до різних предметних областях. Система не вимагає написання програмного коду і виключно проста у використанні, але для її освоєння потрібні значний час і досить глибокі знання теорії ймовірностей, математичної статистики, теорії систем масового обслуговування, мереж Петрі [6].

Основа технологій Arena – мова моделювання SIMAN і система Cinema Animation. Для відображення результатів моделювання використовується анімаційна система Cinema animation. Інтерфейс Arena включає в себе всілякі засоби для роботи з даними [6].

Розробник: Rockwell Automation Inc., Wexford, PA, США. Сайт: www. arenasimulation.com.

Система AutoMod призначена для моделювання систем логістики та виробництва. Програмне забезпечення розроблене для детального аналізу операцій і потоків. Гнучка архітектура AutoMod дозволяє використовувати її в широкому діапазоні прикладних областей, від аеропортів до промисловості напівпровідників [7].

Розробник: Brooks Automation, США. Сайг: www.automod.se/eng/home. html.

AweSim – універсальна система імітаційного моделювання для мережі з дискретної або безперервної інтерпретацією. Можливі області застосування: бізнес, промисловість, охорона здоров'я, військова справа. Продукт включає побудову інтерактивної моделі, одночасну і подальшу анімацію, статистичну інформацію в текстовому і графічному видах, інтерактивне уявлення і вибір сценаріїв. Мережеві моделі будуються графічно і можуть бути ієрархічними. Вони можуть бути розширені за заданим користувачем правилами, написаним на мові C / C ++ або Visual Basic. Одночасно можуть відображатися кілька анімованих зображень. Сценарії порівнюються статистично, після чого з них вибирають набір альтернатив з кращими показниками [8].

Розробник: Symix Systems Inc., США. Сайт: https://awesim.org.

Система Boson NetSim – комерційний симулятор мережевих пристроїв компанії Cisco. Дана система дозволяє отримати практичні знання по роботі з мережевими пристроями. В поставку включається утиліта для моделювання мережі. Сертифікація фахівців Cisco CCNP (Cisco Certified Network Professional) проходить саме в цій програмі. Система унаслідок низької вартості при достатньому функціонал використовується в більш ніж 250 університетах світу [9].

Розробник: Boson, США. Сайт: www.boson.com.

1.4 Постановка задачі

Засобами імітаційного моделювання на основі транспортного графа, вирішити проблему заторів на реальній транспортній мережі. Зібрати статистику, виділити основні фактори які впливають на стан транспортної системи. Розробити імітаційну модель транспортної мережі. Знайти причини появи заторів і шляхи вирішення. Провести реорганізацію транспортних потоків. Провести експерименти і розробити кращі умови для транспортної мережі шляхом оптимізації руху и параметрів системи.

1.5 Висновки

У ході аналізу існуючих шляхів вирішення проблеми, було виявлено, що шляхи вирішення заторів є, але всі вони досить дорогі і займають багато часу. Основним чинником появи заторів – це людський фактор, застаріла автотранспортна система і, як наслідок не правильна організація такої системи при збільшення напливу транспорту. Через це страждає населення, комерційний транспорт і екологія. Вирішення заторів є дуже актуальною темою. Вирішення цієї проблеми на реальних дорогах з використанням імітаційних моделей дає змогу швидко усунути причину без впровадження нових технологій. Використання імітаційних моделей дасть змогу протестувати транспортну систему на різних параметрах і з тестуванням на різних умовах. Цей підхід дає змогу правильно виділити пріоритети і дасть змогу впровадити правильні зміни, з збереженням коштів міста.

2 ОБГРУНТУВАННЯ СТВОРЕННЯ ІМІТАЦІЙНОЇ МОДЕЛІ

Завдання оптимізації, створення нових транспортних розв’язок і в цілому покрашення життя людства ніколи не втрачало своєї актуальності. З огляду на те, що експериментування з реальним об'єктом може привести до негативних наслідків, найбільш придатним інструментом для оцінки альтернативних схем організації дорожнього руху є моделювання.

Моделювання – це проведення експерименту по заміщення одного об’єкта іншим, з метою досягти покращення властивостей об’єкта–оригіналу за допомогою об’єкта-моделі. До того, як почати розробку моделі, необхідно зрозуміти, що собою представляють структурні елементи, з яких така модель будується. Математична або фізична структура моделі може бути дуже складною, але основи її побудови досить прості [10].

У загальному вигляді структуру моделі математично можна уявити так:

$$W = f(x_i, y_i), \tag{2.1}$$

де W – результат роботи транспортної моделі;

x_i – це набір змінних і параметрів, якими є можливість керувати;

y_i – це набір змінних і параметрів, якими неможливо керувати з об’єктивних причин.

Таке спрощення моделі показує – є необхідність довести, що характер протікання процесу залежить від контрольованих і неконтрольованих змінних.

Фактично, кожна модель являє собою комбінацію таких складових, як: змінні, компоненти, параметри, функціональні залежності, обмеження, цільова функція.

Компоненти – це складові частини моделі, які при правильному об’єднанні можуть утворювати систему.

Параметр – це така величина, яка характеризує будь-яку властивість або характеристику об’єкта моделі, за якою можна розділяти такі об’єкти у групи.

А змінні, у свою чергу, є атрибутом такого об’єкту. Змінні виступають як деякі параметри об’єкта, які можна змінювати і з якими можна експериментувати.

При аналізі складних (великих) систем отримав розвиток системний підхід, який відрізняється від класичного підходу. Класичний підхід розглядає систему шляхом переходу від часткового до загального і конструює систему шляхом злиття її компонентів, що розробляються окремо. На відміну від цього системний підхід передбачає послідовний перехід від загального до конкретного, коли в основі розгляду лежить мета, причому досліджуваний об’єкт виділяється з навколишнього середовища.

Системний підхід полягає в тому, що будь-який об’єкт розглядається як самостійна система зі своїми особливостями, функціонування та розвитком.

Важливим для цього підходу є конкретно визначена система, а саме чітко визначені зв’язки, які відображають їх взаємодію.

При системному підході до створення систем моделювання необхідно чітко визначити мету моделювання. Неможливо повністю змоделювати реально функціонуючу систему, створюється модель, під поставлену проблему. Таким чином, стосовно питань моделювання мета виникає з необхідних завдань моделювання, що дозволяє підійти до вибору критерію і оцінити, які елементи увійдуть в створювану модель. Тому необхідно мати критерій відбору окремих елементів в створювану модель.

2.1 Підходи, при побудові імітаційної моделі

Розглядаючи імітаційне моделювання як засіб вирішення проблем бізнесу, можна виділити три основні підходи:

- системна динаміка;
- дискретно-подієвого моделювання (процесно-орієнтований);
- агентне моделювання.

Дискретно-подійне моделювання – різновид імітаційного моделювання. В дискретно-подійному моделюванні функціонування системи представляється

як хронологічна дискретна послідовність подій. Подія відбувається у визначений момент часу та призводить до зміни стану системи. Між цими подіями система перебуває у незмінному стані [11].

Агентне моделювання можна визначити як метод імітаційного моделювання, який досліджує поведінку децентралізованих агентів і то, як ця поведінка визначає поведінку всієї системи в цілому. При розробці агентної моделі, інженер вводить параметри агентів (це можуть бути люди, компанії, активи, проекти, транспортні засоби, міста, тварини, тощо.), визначає їх поведінку, поміщає їх в навколишнє середовище, встановлює можливі зв'язки, після чого запускає моделювання. Індивідуальна поведінка кожного агента утворює глобальну поведінку моделі [12].

Системна динаміка – це підхід імітаційного моделювання, своїми методами і інструментами дозволяє зрозуміти структуру та динаміку складних систем. Також системна динаміка – це метод моделювання, який використовується для створення точних комп'ютерних моделей складних систем для подальшого використання з метою проектування більш ефективної організації і політики взаємин з даною системою. Разом, ці інструменти дозволяють створювати мікросвіти-симулятори, де простір і час можуть бути стиснуті і уповільнені так, щоб можна було вивчити наслідки рішень, швидко освоїти методи і зрозуміти структуру складних систем, спроектувати тактики і стратегії для більшого успіху [13].

Перші два підходи є «традиційними» методами імітаційного моделювання, що з'явилися в 50–60х роках. Агентне моделювання – відносно новий метод, що отримало широке практичне розповсюдження тільки після 2000 року. Системна динаміка і дискретно-подієве моделювання розглядають систему зверху вниз, працюючи на так званому системному рівні. Агентне моделювання – це підхід знизу-вгору [14].

Системна динаміка передбачає високий рівень абстракції і використовується в основному для завдань стратегічного рівня.

Процесно-орієнтований (дискретно-подієвий) підхід використовується в основному на операційному і тактичному рівні. Агентне моделювання включає завдання будь-якого рівня абстракції: агент може представляти компанію на ринку, покупець, проект, ідея, транспортний засіб, пішохід, робот, тощо [14].

2.2 Транспортна система – як об’єкт моделювання

Транспортна система – це сукупність транспортних засобів, доріг (шляхів), засобів інформування, засобів управління, а також різних технічних пристроїв, механізмів що забезпечують роботу.

Інфраструктура – це такий фізичний компонент системи, який створює транспортну мережу, що включає зв’язки і перетини.

Переміщення транспортних засобів по мережі утворює транспортні потоки. При моделюванні і налаштуванні транспортних засобів у мережі, слід враховувати діапазон характеристик. Залежно від транспорту, будь то велосипед, чи автобус, трамвай, чи залізничний рухомий склад, слід враховувати, що будуть змінюватися характеристики геометричні і технічні.

З цієї системи не слід виділяти присутність водія, а саме людський фактор. Система управління транспортними потоками виконує необхідні дії щодо впорядкування руху транспортних засобів і виключенню конфліктів між ними. Ця система оперує знаками, дорожньою розміткою та сигналами у відповідно до правил дорожнього руху.

Ефективність транспортної системи не може розглядатися тільки в рамках досягнення оптимальності виконання відповідних процесів усередині системи. Основними завданнями транспортної системи є задоволення потреби економіки у перевезенні вантажів та забезпечення мобільності населення. У зв’язку з цим ефективність транспортної системи завжди буде визначатися таким собі балансом між суперечливими вимогами економіки та суспільства. Яскравим прикладом є бажання пасажира, щоб транспорт під’їхав до зупинки, як тільки пасажир підійшов до неї, і бажання перевізника встановити такий

інтервал руху, щоб транспортні засоби завжди були заповнені повністю і приносили, максимальний дохід. Таким чином, для побудови ефективної транспортної системи необхідно знання в галузі транспорту поєднувати з економікою, містобудуванням, географією, екологією, соціологією і психологією.

2.3 Транспорт як об’єкт моделювання

Вивчення сутності проблем транспортної перевантаженості сучасних великих міст і мегаполісів спиралося на два методологічні принципи: принцип історизму та принцип системності. Згідно з принципом історизму основна увага в аналізі було приділено формуванню, розвитку і динаміці досліджуваних об’єктів. Аналіз спирається на принцип історизму як методологічний вираз саморозвитку дійсності, включаючи вивчення сучасного стану, минулого і процесів генезису. Принцип системності дозволяє розглядати міський транспорт як складну систему. Причому тут можливі різні підходи. Наприклад, можна розглядати міський транспорт як складну систему з точки зору [14]:

– інтенсивності руху по міських магістралях;

– пропускної здатності перехресть;

– оптимального регулювання і розподілу транспортних потоків за допомогою заборонних та обмежують знаків.

В цьому випадку елементами складної системи будуть:

– міські магістралі та перехрестя;

– засоби сигналізації та управління;

– транспортні засоби.

Можливо вивчати міський пасажирський транспорт як складну систему з транспортних засобів (тролейбусів, автобусів, трамваїв, метрополітену, таксі та ін.), маршрутів руху, перехресть і світлофорів з урахуванням їх завантаження іншими видами транспорту, пасажиропотоками, що формуються в різних пунктах міста в залежності від часу доби, диспетчерських пунктів, засобів

зв’язку і збору інформації, органів планування і управління, засобів ремонту і заправки автомобілів і т.д. При такому підході міський пасажирський транспорт як складна система розглядається з точки зору якості обслуговування пасажирів, планування маршрутів, розподілу рухомого складу на маршрутах і визначення оптимальних режимів руху (розкладів), планування поточного та капітального ремонту транспортних засобів [14]. Аналогічно як складну систему можна розглядати міський вантажний транспорт. Замість пасажиропотоків розглядаються вантажопотоки від постійних і тимчасових джерел, що вимагають доставки в постійні і тимчасові пункти призначення. При дослідженні такої складної системи виникають питання попереднього і оперативного планування перевезень, що забезпечує своєчасну і економічну доставку вантажів, а також проблеми, пов’язані з нормальною експлуатацією транспортних засобів [14].

2.4 Імітаційне моделювання, як спосіб вирішення проблеми

Імітаційне моделювання – це різновид математичного моделювання, в якому опис моделі задається у вигляді алгоритмів поведінки та взаємозв’язку елементів модельованої системи. Використовувані алгоритми дозволяють імітувати як поведінку елементів системи, так і всієї системи в цілому, а також визначати необхідні параметри функціонування системи [15].

Такі імітаційні моделі відтворюють події, що відбуваються в реальній системі. При імітаційному моделюванні транспортної мережі не потрібно купувати дороге обладнання – його роботи імітується програмами, досить точно відтворюють всі основні особливості і параметри такого устаткування.

Перевагою таких систем є можливість швидкої зміни подій, прискорення процесу роботи моделі. Можна в результаті декількох хвилин отримати статистику роботи моделі на основі деяких годин, днів, що надає змогу оцінити роботу мережі в широкому діапазоні зміни параметрів.

Як вже вказувалось, результатом роботи імітаційної моделі є зібрані в ході спостереження за подіями статистичні дані про найбільш важливі характеристики мережі.

2.5 Висновки

Вирішення питання оптимізації дорожньої мережі можливе за допомогою імітаційного моделювання. Основною перевагою використання сукупності такого підходу – це, по перше, вартість створення такої моделі. А по друге – можливість швидкого реагування на зміни в системі, швидку зміну параметрів, проведення великої кількості експериментів і виявлення вразливостей у модельованій системі, усунути, і тим саме оптимізувати рух.

3 ПРОЕКТУВАННЯ Й РОЗРОБКА ІНСТРУМЕНТАЛЬНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис середовища, подходів і мови програмування

AnyLogic – єдиний інструмент імітаційного моделювання (ІМ), який підтримує всі підходи до створення імітаційних моделей: процесно-орієнтований (дискретно-подієвий), системно динамічний і агентний, а також будь-яку їх комбінацію [5].

Унікальність, гнучкість і потужність мови моделювання AnyLogic дозволяє врахувати будь-який аспект моделювання з будь–яким рівнем деталізації. Графічний Інтерфейс AnyLogic, інструменти та бібліотеки дозволяють швидко створювати моделі для широкого спектру завдань від моделювання виробництва, логістики, бізнес-процесів до стратегічних моделей розвитку компанії і ринків. AnyLogic – це імітаційна платформа для повного бізнес-циклу [5].

AnyLogic був обраний тому, що:

– графічне середовище розробки моделей AnyLogic значно прискорює процес створення моделей [5];

– створення бібліотек дозволяє розробнику багаторазово використовувати вже написані модулі [5];

– об'єктно-орієнтований підхід піднімає процес розробки моделей на новому рівні [5];

– інтуїтивний графічний інтерфейс спрощує перехід з інших інструментів імітаційного моделювання на AnyLogic [5];

– завдяки вбудованим бібліотекам, написаними професіоналами, можна створювати складні моделі всього в «два кліка» мишею [5];

– проектувати агентні, системно-динамічні, дискретно-подієві і багато підхідні моделі, використовуючи тільки один інструмент [5];

– AnyLogic підтримує як дискретний, так і безперервний підхід в межах однієї моделі [5];

– java платформа інструменту AnyLogic надає безмежну розширюваність моделей за рахунок програмування на Java, створення призначених для користувача бібліотек і роботи з базами даних [5];

– сильна експериментальна база, вбудована підтримка моделювань Монте Карло і передових форм оптимізації дає велику різноманітність підходів моделювання [5];

– AnyLogic написаний на мові програмування Java, тому він є мультиплатформенним програмним продуктом. Середовище розробки і моделі працюють на Windows, Mac OS і Linux [5];

– AnyLogic включає в себе можливість створення інтерактивної анімації для поліпшення наочності моделей [5].

Агент – елемент моделі, який може мати поведінку, пам'ять (історія), контакти тощо. Агенти можна моделювати у вигляді людей, компаній, проектів, автомобілей, міст, тварин, кораблів, товарів тощо.

Можна створювати всередині об'єкта змінні, діаграми станів, задавати події, потокові діаграми системної динаміки, а також додавати всередину агента об'єкти бібліотек AnyLogic. Можна створити в одній моделі стільки типів агентів, скільки агентів потрібно промоделювати [5].

Створення агента зазвичай починається з визначення його інтерфейсу для зв'язку із зовнішнім світом. У разі систем з великою кількістю агентів з динамічними зв'язками (наприклад, в моделях соціальних мереж) агенти можуть взаємодіяти один з одним шляхом виклику методів друг у друга [5].

Початковий стан і поведінка агента може бути реалізована різними способами. Стан (накопичена історія) агента може бути представлена за допомогою змінної, або стану діаграми станів. Поведінка може бути або пасивною (агенти реагують тільки на прибуття повідомлень або на виклик методів і не мають власних подій, запланованих на майбутнє) або активною,

коли внутрішня динаміка агента(події, заплановані через заданий таймаут або процеси системної динаміки) є причиною дій, що здійснюються агентом. В останньому випадку всередині агентів швидше за все повинні бути задані події і/або діаграми станів [13].

AnyLogic включає в себе графічну мову моделювання, а також дозволяє користувачеві розширювати створені моделі за допомогою мови Java. Інтеграція компілятора Java в AnyLogic надає більш широкі можливості при створенні моделей, а також створення Java аплетів, які можуть бути відкриті будь-яким браузером. Ці аплети дозволяють легко розміщувати моделі AnyLogic на веб-сайтах. На додаток до Java-аплетів, AnyLogic Professional підтримує створення Java-додатків, в цьому випадку користувач може запустити модель без інсталяції AnyLogic [15].

Графічне середовище моделювання AnyLogic включає в себе наступні елементи:

– Stock & Flow Diagrams (діаграма потоків і накопичувачів) застосовується при розробці моделей, використовуючи метод системної динаміки [15];

– Statecharts (карти станів) в основному використовуються в агентних моделях для визначення поведінки агентів. Вони також часто використовуються в дискретно-подієвому моделюванні, наприклад для симуляції машинних збоїв [15];

– Action charts (блок-схеми) використовуються для побудови алгоритмів. Застосовується в дискретно-подієвому моделюванні (маршрутизація дзвінків) і агентно моделюванні (для логіки рішень агента) [15];

– Process flowcharts (процесні діаграми) – основна конструкція, яка використовується для визначення процесів в дискретно-подієвому моделюванні [15].

Середовище моделювання також включає в себе: низькорівневі конструкції моделювання (змінні, рівняння, параметри, події тощо), форми подання (лінії, квадрати, овали і т.п), елементи аналізу (бази даних, гістограми, графіки), стандартні картинки і форми експериментів [15].

Середовище моделювання AnyLogic підтримує проектування, розробку, документування моделі, виконання комп'ютерних експериментів з моделлю, включаючи різні види аналізу – від аналізу чутливості до оптимізації параметрів моделі щодо деякого критерію [15].

3.2 Опис використаних бібліотек

При розробці моделі було використано елементи з таких бібліотек, як:

– бібліотека дорожнього руху (елементи бібліотеки показані на рис 3.1);

– бібліотека моделювання процесів (елементи бібліотеки показані на рис 3.2);

– пішохідна бібліотека (елементи бібліотеки показані на рис 3.3).

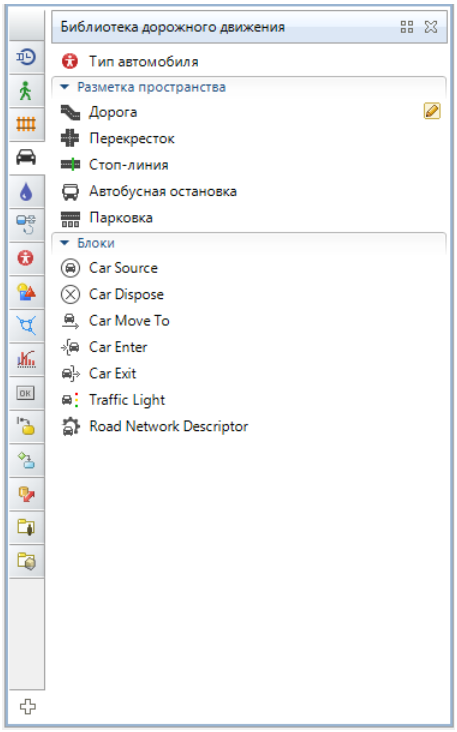


Рисунок 3.1 – Дорожня бібліотека

Бібліотека дорожнього руху дозволяє детально імітувати фізичне переміщення машин по дорожній мережі. Крім того, вона дає можливість моделювати:

– рух з урахуванням правил дорожнього руху;

– світлофори і пріоритети проїзду на перехрестях;

– місця для паркування;

– рух і зупинки громадського транспорту.

Можна легко моделювати вулиці, дороги, перехрестя, розв'язки, під'їзди до громадських будівель, технологічні проїзди на виробництві та будь-які інші системи, де є машини і дороги. У бібліотеці також є інструмент, що дозволяє візуалізувати щільність трафіку в мережі [1].

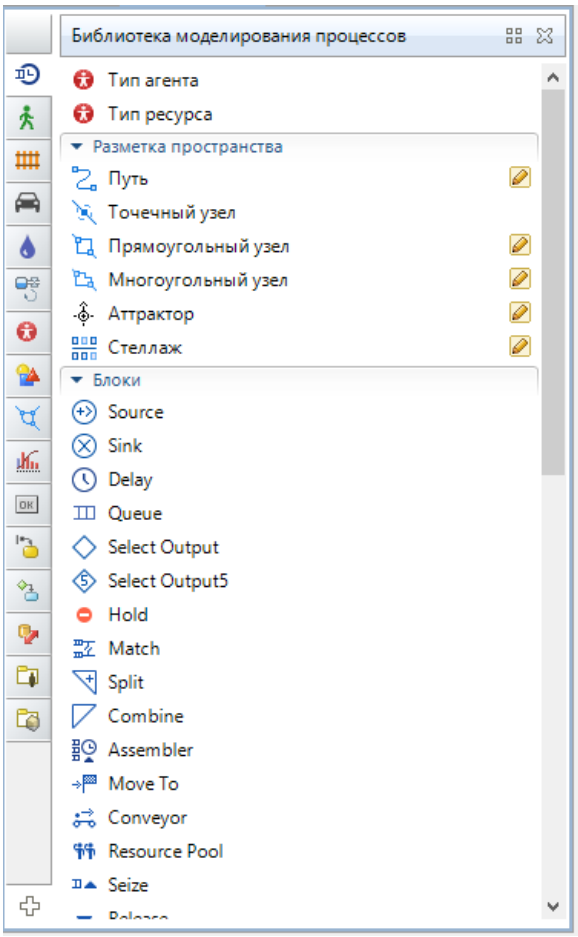


Рисунок 3.2 – Бібліотека моделювання процесів

Бібліотека моделювання процесів AnyLogic підтримує дискретно-подієвий, або, якщо бути більш точним, «процесний» підхід моделювання. За допомогою об'єктів бібліотеки моделювання процесів можна моделювати системи реального світу, динаміка яких представляється як послідовність операцій (прибуття, затримка, захоплення ресурсу, поділ) над агентами, що представляють клієнтів, документи, дзвінки, пакети даних, транспортні засоби, тощо. Ці агенти можуть володіти певними атрибутами, що впливають на процес їх обробки (наприклад, тип дзвінка, складність роботи) або накопичують статистику (загальний час очікування, вартість).

Процес задається у формі потокових діаграм (блок-схем) – графічне представлення, прийняті в багатьох областях: виробництво, бізнес-процеси, центри обробки дзвінків, логістика, охороні здоров'я, тощо. Потокові діаграми AnyLogic ієрархічні, масштабуються, розширювані об'єктно-орієнтовані, що дозволяє користувачеві моделювати складні системи будь-якого рівня детальності. Інша важлива особливість бібліотеки моделювання процесів є можливість створення досить складних анімацій процесних моделей [1].

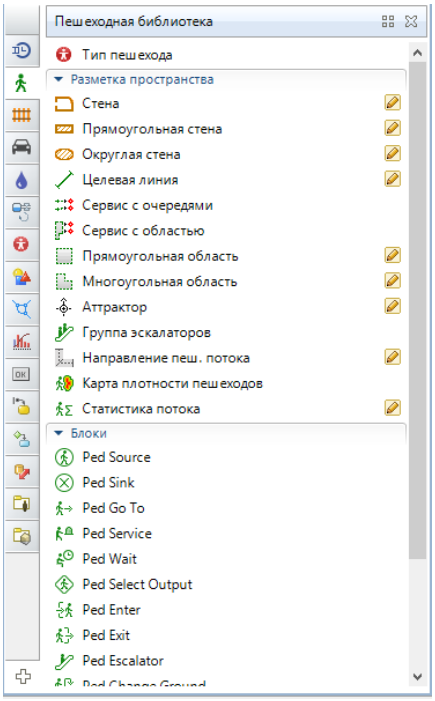


Рисунок 3.3 – Пішохідна бібліотека

Пішохідна бібліотека AnyLogic є високорівневою бібліотекою моделювання руху пішоходів у фізичному просторі. Вона дозволяє моделювати будівлі, в яких рухаються пішоходи (станції метро, стадіони, музеї), а також вулиці і інші місця великого скупчення людей. З нею можна збирати статистику. Можна збирати статистику щільності пішоходів в різних областях моделі для того, щоб переконатися, що сервіси зможуть впоратися з потенційним зростанням навантаження, обчислити час перебування пішоходів в якихось певних ділянках моделі, виявити можливі проблеми, які можуть виникнути при перепланування інтер'єру будівлі, тощо. У моделях, створених за допомогою об'єктів пішохідної бібліотеки, пішоходи рухаються в безперервному просторі, реагуючи на різні види перешкод у вигляді стін, різних областей і інших пішоходів [1].

Моделі руху пішоходів складаються з двох складових – середовища і поведінки. Під середовищем мається на увазі об'єкти фізичного середовища – стіни, різні області, сервіси, черги, тощо.

Основний об'єкт бібліотеки є пішохід. Пішохід задається за допомогою об'єкта типу Ped. Пішохід «живе» в заданому фізичному просторі (середовищі) і пересувається згідно із заданими правилами. З іншого боку, тип пішохода успадкований від типу агента Agent, тому пішоходи пересуваються по блок-схемі так само, як агенти.

Пішохідна бібліотека сумісна з бібліотекою моделювання процесів AnyLogic. Це дозволяє використовувати в пішохідних моделях будь-які об'єкти бібліотеки моделювання процесів, роблячи можливим створення складних моделей, що складаються з блок-схем бібліотеки моделювання процесів і середовища пішохідних бібліотеки. Така сумісність можливо завдяки наявності в пішохідній бібліотеці об'єктів, що перетворює агентів в пішоходів і навпаки.

Блок-схема пішохідної моделі будується за допомогою об'єктів, що міститься в пішохідній бібліотеці. Тип агента Ped є базовим типом для

моделювання пішоходів. В бібліотеці є об'єкти для створення пішоходів і управління потоком пішоходів [1].

Правила завдання потоку пішоходів аналогічні правилам завдання потоку агентів в бібліотеці моделювання процесів. Різниця полягає в тому, що пішоходи рухаються згідно з правилами руху в фізичному просторі і вибирають свій шлях, аналізуючи поточний стан в просторі.

3.3 Опис агенів

Main – головний агент моделі, який об'єднує всі інші агенти. Цей агент описує повну роботу всієї моделі

Bus113A – агент, який описує властивості автобуса, що перевозить людей по системі. Він має свій параметр «ВремяПоявления113», який фіксує час появи агента в системі. Агент має свою модель з 2D/3D текстурою. Цей агент показаний на рис 3.4.

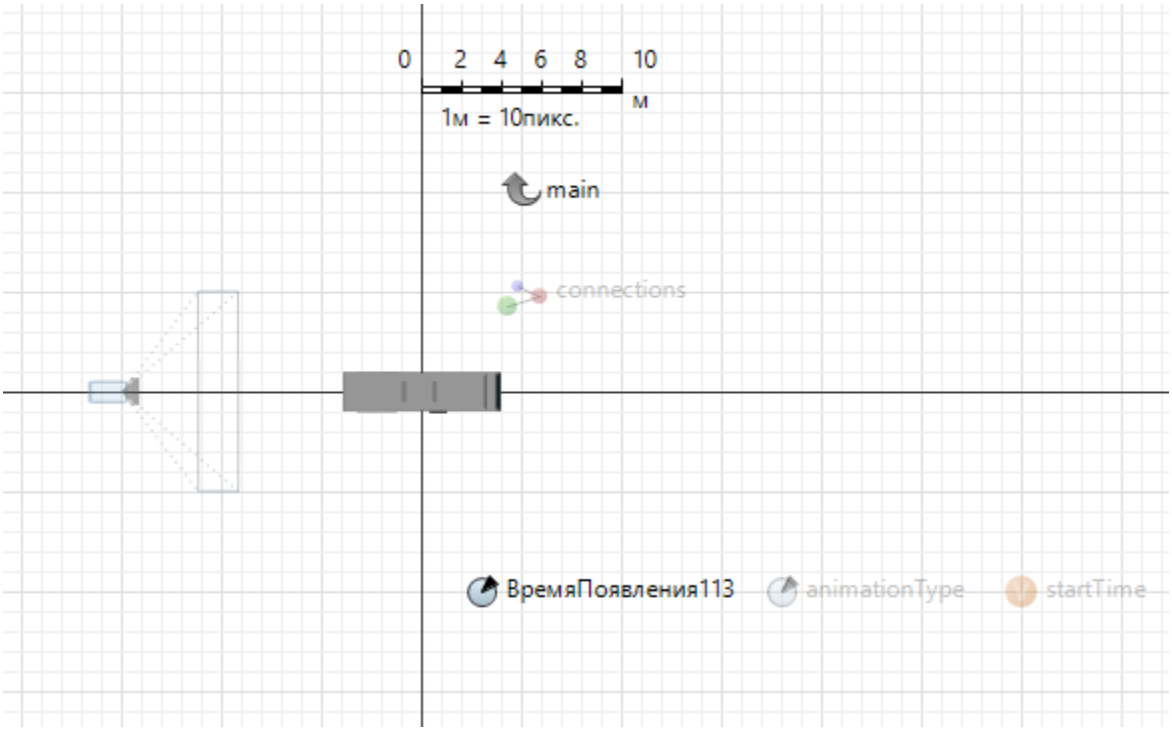


Рисунок 3.4 – Модель агента Bus113A

При знищенні агента виконується код, показаний на лістингу 3.1.

Лістинг 3.1 – Знищення агента Bus113A.

main.ВремяПроезда113.add (time() – ВремяПоявления113);

При появі агента у системі, виконується код, який показано у лістингу 3.2.

Лістинг 3.2 – Знищення агента Bus113A.

main.ВсегоАвто++;

Початкова швидкість агента 10м/с.

Bus146A – агент, який описує властивості автобуса, що перевозить людей по системі. Він схожий с агентом Bus113A і має такий же параметр – «ВремяПоявления146».

Car – агент, який описує властивості авто, що рухаються по системі. Цей агент схожий з агентами Bus146A та Bus113A. Має властивість «ВремяПоявления». Цей агент має п'ять текстур різного кольору. Агент представлений на рис 3.5.

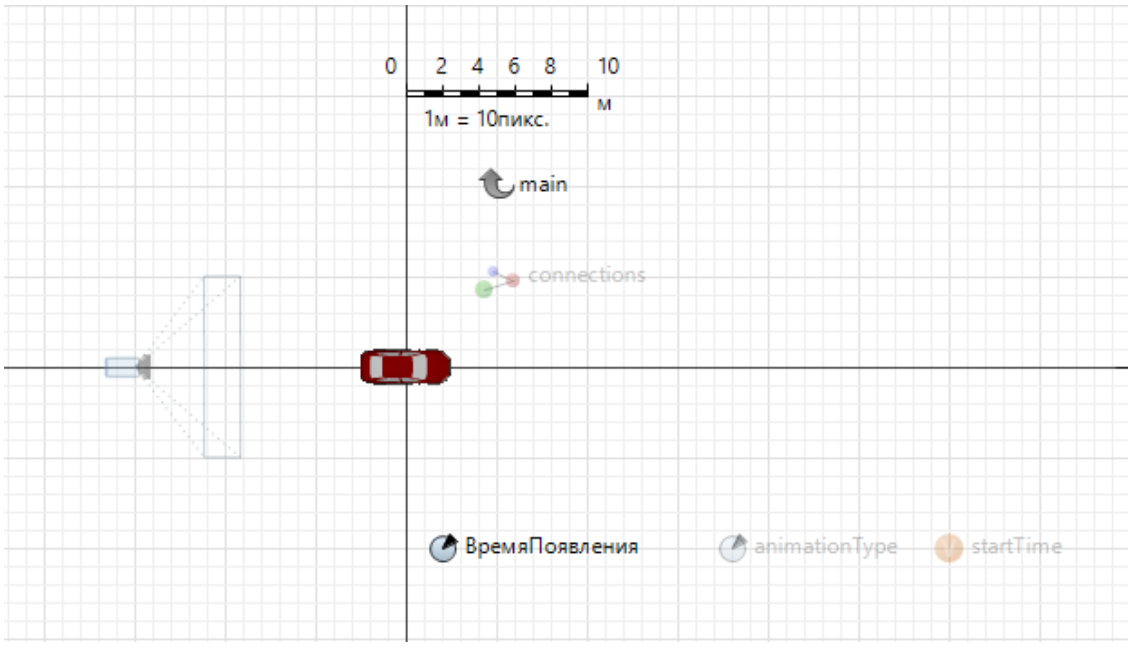


Рисунок 3.5 – Модель агента Car

При знищенні агента виконується код, показаний на лістингу 3.3.

Лістинг 3.3 – Знищення агента Car.

main.ВремяПроезда.add (time() – ВремяПоявления);

При появі агента у системі, виконується код, який показано у лістингу 3.4.

Лістинг 3.4 – Поява агента Car.

main.ВсегоАвто++;

main.СтатистикаАвто.add(main.Авто.size());

Vehicle – це агент без текстури, який слугує для того, щоб доповнювати агенти Car, Bus146A та Bus113A та розширювати їх функціонал. Цей агент має камеру camera, параметр animationType та змінну startTime.

Пешеход – агент, який описує властивості пішохода. Він схожий з агентом Car крім того, що він не доповнюється агентом Vehicle і має свою текстуру.

При знищенні агента виконується код, показаний на лістингу 3.5.

Лістинг 3.5 – Знищення агента Пешеход.

main.ВремяНахождения.add (time() – ВремяПоявленияЧеловека);

При появі агента у системі, виконується код, який показано у лістингу 3.6.

Лістинг 3.6 – Знищення агента Пешеход.

main.ВсегоЛюдей++;

main.СтатистикаПешеходов.add (main.ПешеходВСистеме.size());

3.4 Опис елементів навігації

Навігація по працюючій моделі здійснюється за допомогою миші або панелі, що показано на рис 3.6.

Рисунок 3.6 – Панель навігації

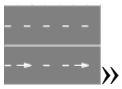
Ця панель була зроблена з таких елементів як прямокутник і текст. Кожен прямокутник має свій колір, в залежності від обраного пункту. При натисканні лівою кнопкою миші на відповідний прямокутник, виконується операція, відповідно до тексту. Приклад показано в лістингу 3.7.

Лістинг 3.7 – Операція, при натисканні на прямокутник

```
viewAreaStatistics.navigateTo();
```

3.5 Опис елементів та блоків в моделі

В імітаційній моделі використовуються такі елементи:

Елемент «Дорога  » є графічним елементом розмітки простору, це безперервна дорога (тобто такої дороги, яка не містять перехрестя). Використовуючи дороги, перехрестя і інші елементи розмітки простору, можна створювати дорожні мережі для моделей бібліотеки дорожнього руху.

За допомогою спеціального елемента «  Масштаб» можна задати масштаб анімації моделі, тобто, відношення розміру об'єкта на анімації моделі до натурального розміру об'єкта, що моделюється. Фактично, масштаб задається як співвідношення пікселів анімації і фізичної одиниці довжини.

За допомогою елемента розмітки простору «перехрестя» можна поєднувати дві або більше дороги. Використовуючи дороги, перехрестя і інші елементи розмітки простору, є можливість створювати дорожні мережі для моделей бібліотеки дорожнього руху. Відображені на перехресті білі точкові лінії – з'єднувачі смуг – задають дозволені напрямки руху транспорту на перехресті. Приклад перехрестя показано на рис 3.7.

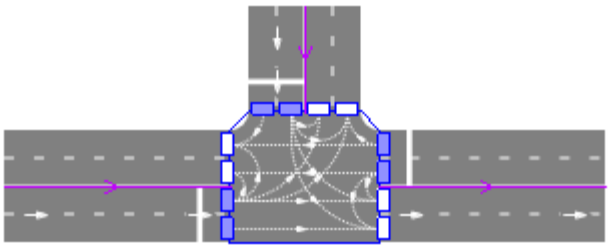


Рисунок 3.7 – Перехрестя

За допомогою елемента розмітки простору «автобусна зупинка» можна намалювати автобусну зупинку на узбіччі дороги у напрямку руху. Приклад автобусної зупинки показано на рис 3.8.

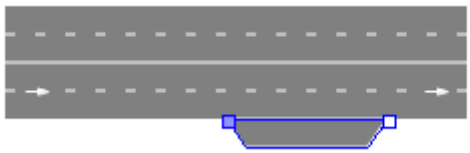


Рисунок 3.8 – Автобусна зупинка

За допомогою елемента розмітки простору «парковка» можна малювати однорядні паркувальні місця, розташовані на узбіччях дороги.

Парковка може бути паралельна (машини паркуються в одну лінію з іншими припаркованими машинами), або перпендикулярна – це задається у властивості «тип парковки». Приклад парковки приведено на рис 3.9.

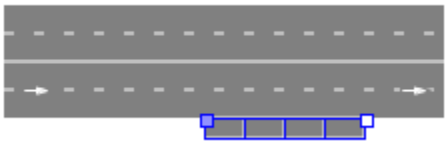


Рисунок 3.9 – Парковка

CarSourse – створює автомобілі і намагається помістити їх в зазначеному місці дорожньої мережі (на обраній дорозі або парковці).

 CarDispose – видаляє машини з моделі.

 CarMoveTo – блок, який керує рухом автомобіля. Автомобіль може їхати, тільки коли він знаходиться в блоці CarMoveTo. Автомобіль намагається розрахувати шлях від свого поточного місця до зазначеного місця призначення, коли надходить до блоку CarMoveTo. В якості місця призначення можуть виступати: дорога, парковка, автобусна зупинка або стоп-лінія.

 TrafficLight – моделює світлофор, керуючий рухом машин на перехресті або на будь-якій стоп-лінії.

 RoadNetworkDescriptor – опціональний блок. За допомогою блоку RoadNetworkDescriptor розробники отримують доступ до управління всіма транспортними засобами, що перебувають в одній дорожньої мережі. Блок дозволяє задавати дії, які будуть виконуватися при додаванні автомобіля в дорожню мережу, в'їзді на дорогу, зупинки автомобіля, зміні смуги, тощо. Крім того, за допомогою даного блоку можна включити відображення пробок на дорогах

 PedSource – створює пішоходів. Зазвичай використовується в якості початкової точки діаграми пішохідного процесу.

 PedSink – видаляє пішоходів. Зазвичай використовується в якості кінцевої точки діаграми пішохідного процесу.

 PedGoTo – змушує пішоходів йти до заданої точки простору.

 PedEnter – поміщає вже створених раніше пішоходів в модельовану середу.

 PedExit – видаляє пішоходів з моделюючого середовища.

 Delay – затримує агентів на заданий період часу.

 Queue – зберігає агентів в певному порядку. Моделює чергу агентів, які чекають прийом об'єктами, заданими в потоковій діаграмі.

 SelectOutput – направляє агентів в один з двох вихідних портів в залежності від виконання заданої умови.

SelectOutput5 – направляє агентів в один з п'яти вихідних портів в залежності від виконання заданих умов.

Pickup – додає агентів до вмісту агента-контейнера.

DropOff – видаляє обраних агентів з агента-контейнера і посилає їх далі.

3D Вікно – це елемент, що задає на діаграмі агента область, в якій під час запуску моделі буде відображатися тривимірна анімація цього об'єкта.

Область перегляду – За допомогою цього елемента можна виділити на діаграмі типу агента, деякі області, що містять логічно відокремлені групи елементів або ділянки діаграми. Задавши такі області, можна легко перемикатися між ними під час виконання моделі за допомогою спеціальних засобів навігації. Це дозволить швидко переходити до тієї або іншої ділянки діаграми агента.

Гістограма – це елемент, що відображає дані, зібрані об'єктом «дані гістограми» (на одній гістограмі можуть одночасно відображатися дані відразу декількох таких об'єктів). Ось X завжди масштабується таким чином, щоб вмістити всі дані. Масштаб по осі Y також вибирається автоматично, таким чином, щоб висота найвищої стовпця дорівнювала висоті області діаграми.

3.6 Опис логічної структури

Початок руху авто починається в одному із блоків CarSourse. Приклад властивостей блоку приведено на рис 3.10.

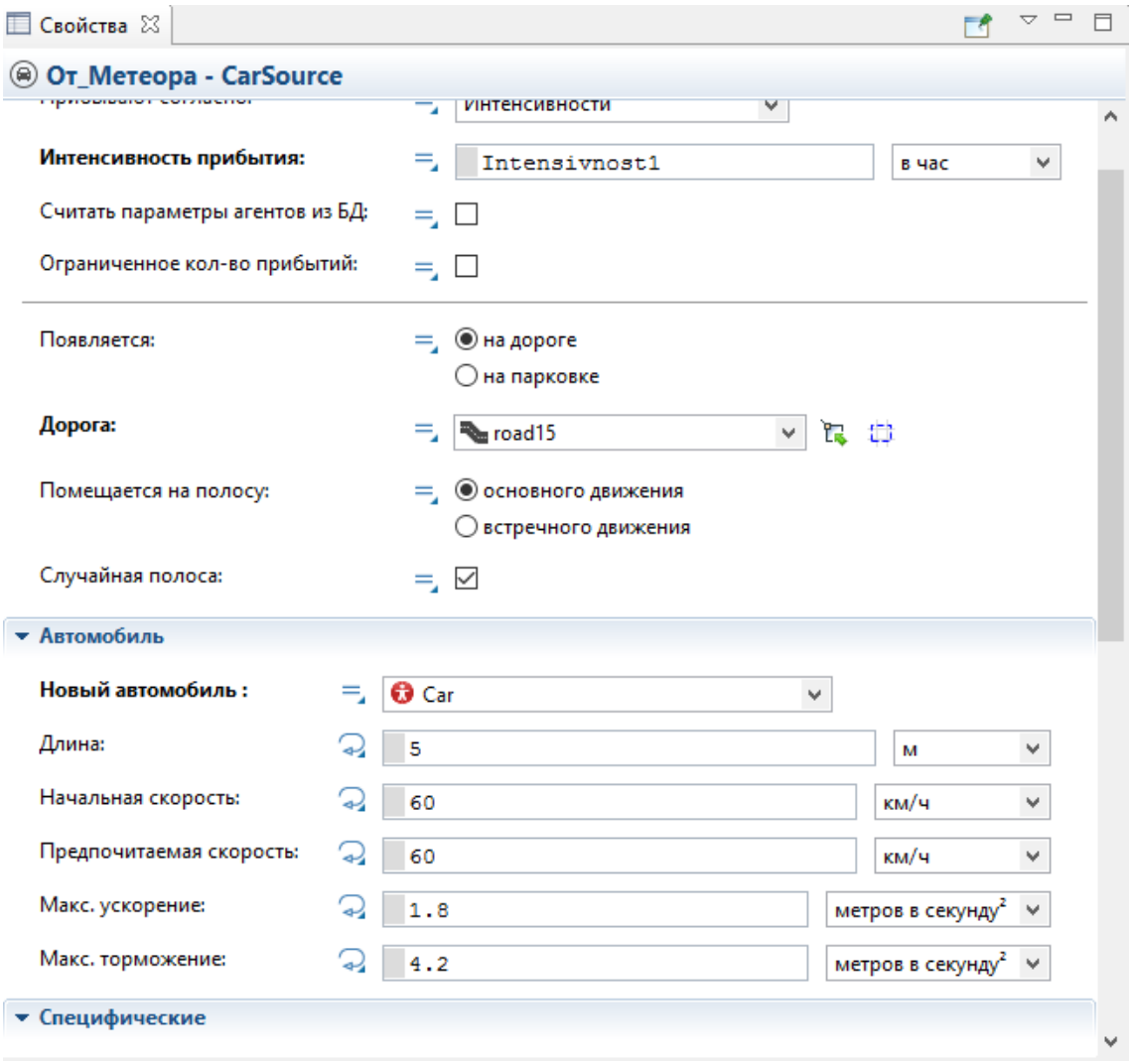


Рисунок 3.10 – Властивості блоку CarSourse

В кожному з блоків CarSourse можна вказати початкову дорогу або парковку, на якій будуть з'являтися автомобілі. Обрати тип автомобіля (агента, який відповідає за автомобіль), вказати початкові дані транспорту (початкову швидкість, середню швидкість, тощо.), задати інтенсивність потоку.

Після цього потік потрапляє в блок selectOutput5, який розподіляє, куди буде рухатися те чи інше авто. В блоці selectOutput5 є п'ять виходів із своїми ймовірностями. Кожен із виходів має свою вірогідність використання, це показано на рис 3.11.

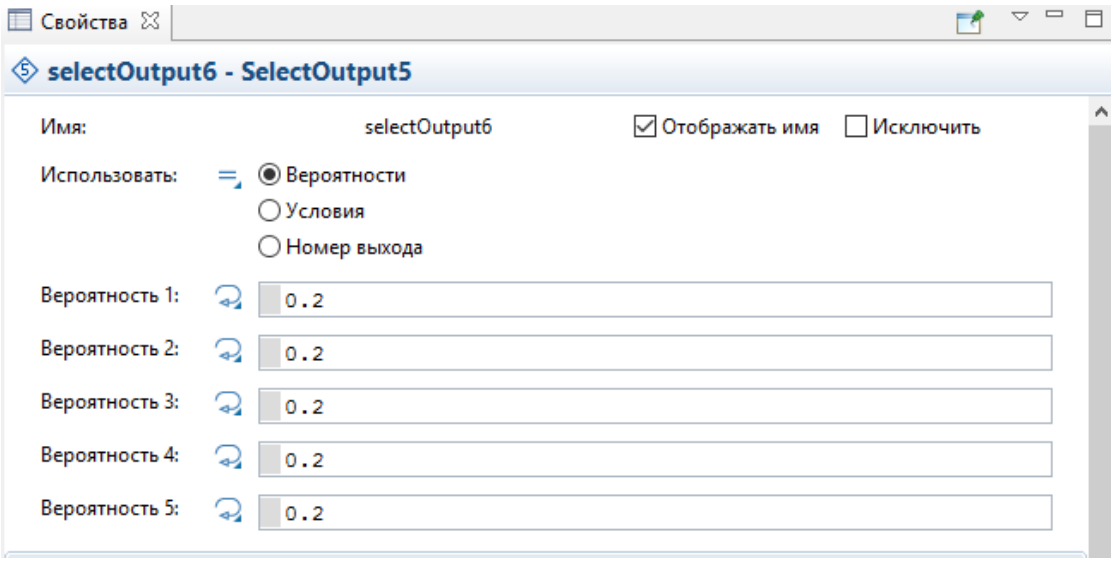


Рисунок 3.11 – Властивості блоку selectOutput5

Після обрання шляху авто потрапляє в блок selectOutput, який по властивостям схожий з блоком selectOutput5, але має лише 2 ймовірності з виходами:

По першій ймовірності, авто потрапляє в блок CarMoveTo, у властивостях якого вказано, до якої кінцевої дороги потрібно їхати автомобілю. Після приїзду до кінця обраної дороги, авто потрапляє до блоку carDispose, який видаляє авто з системи.

По другій ймовірності, авто потрапляє до блоку selectOutput5. Після цього блоку авто потрапляє в блок CarMoveTo, в якому вказано місце знаходження парковки. Якщо на парковці не має місця, то авто їде далі по логіці. Якщо місце вільне то авто паркується і потрапляє в блок Delay, в якому вказано скільки автомобілю потрібно очікувати часу. Після потрібного очікування авто їде далі по логіці моделі

Після пункту 2, авто знов потрапляє в блок вибору шляху, де по ймовірності обирається блок CarMoveTo. Авто їде до потрібної дороги, після чого видаляється з системи.

Логіка проїзду автомобілів в системі показано на рис 3.12.

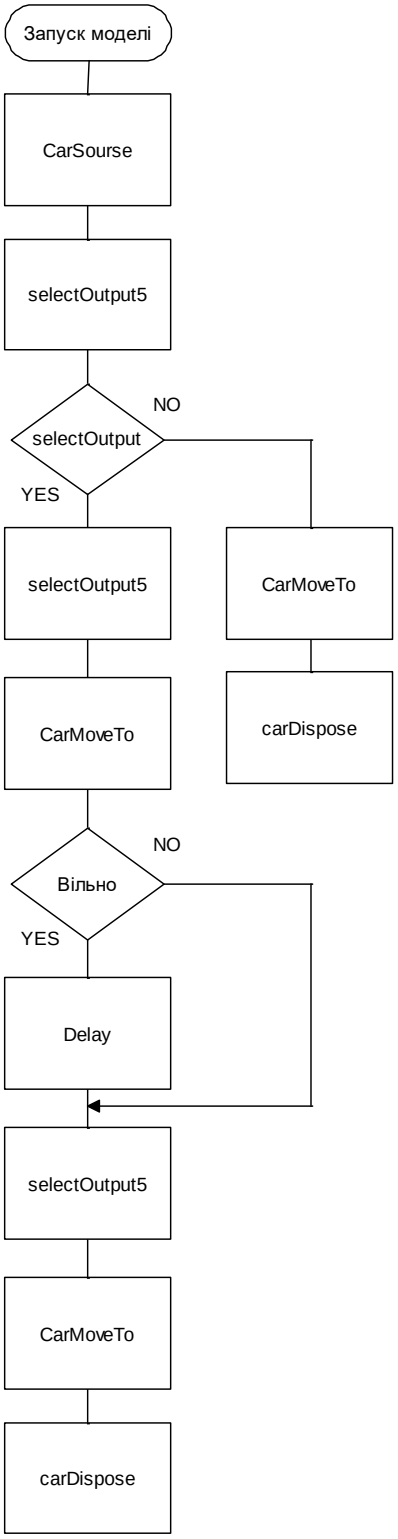


Рисунок 3.12 – Схема логіки автомобілів

Початок руху автобуса починається з блоку CarSourse. Властивості його такі, як показано на рис 3.10 окрім того, що блок має інший тип автомобілю,

інші параметри та інтенсивність появи. Паралельно з ним працює блок pedSourse, який відповідає за появу пішоходів в системі. Властивості цього блоку представлено на рис 3.13.

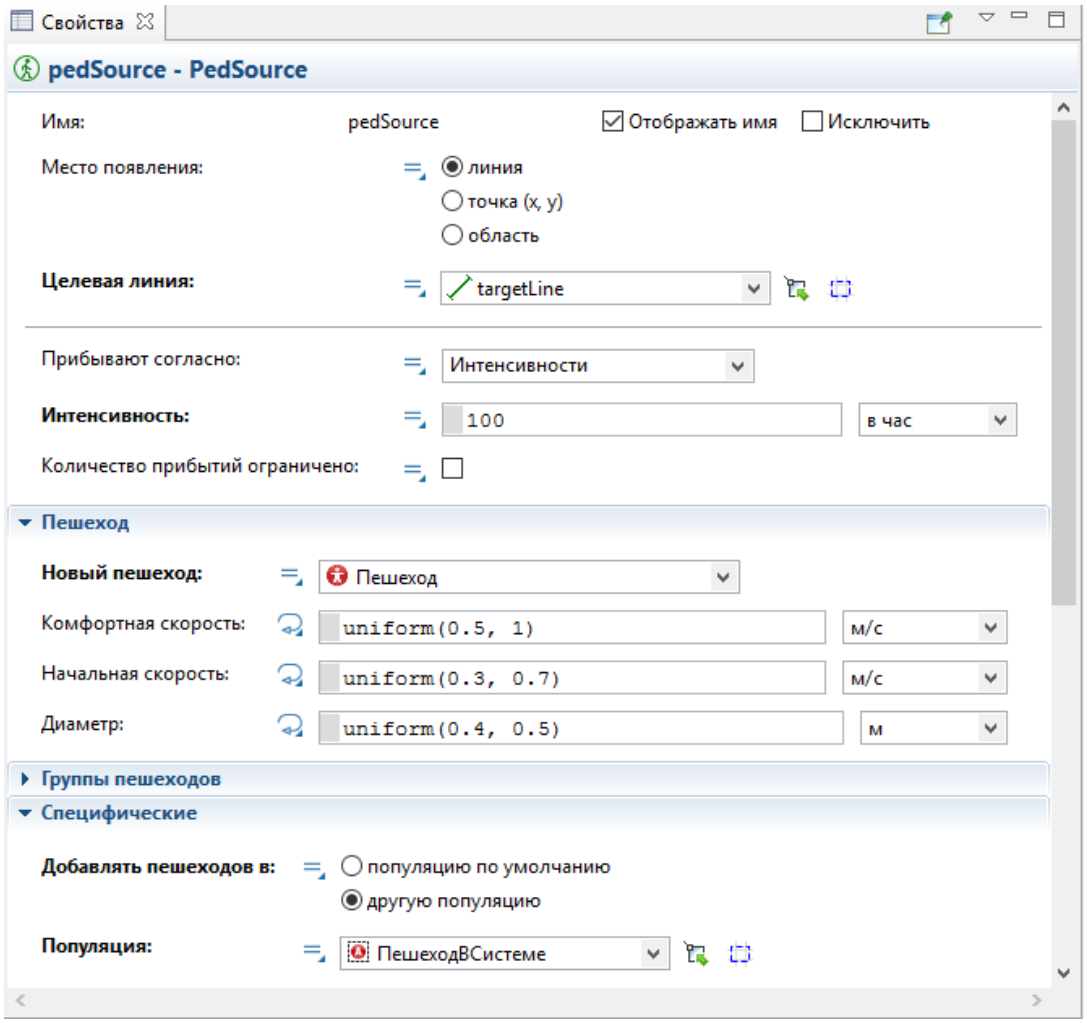


Рисунок 3.13 – Властивості блоку pedSourse

В властивостях блоку указано лінію, звідки з'являються пішоходи. Вказана їх інтенсивність появи, агент, популяція та швидкість.

Логіка автобуса наступна: автобуси з'являються в блоці CarSourse на дорозі, яка вказана в властивостях блоку. Після цього потрапляють в блок CarMoveTo, в якому вказано місце знаходження першої зупинки. Їдуть до неї після чого потрапляють в блок queue. Це блок черги, тобто якщо на зупинці стоять інші автобуси, автобус під'їжджає до зупинки і чекає поки звільниться

місце, після чого заїжджає на зупинку і зупиняється, та потрапляє в блок pickup. Автобус підбирає пішоходів, як вказано в блоці pickup (максимальна місткість автобуса – 45 пішоходів), після чого потрапляє в блок delay, в якому чекає відповідно до вказаного часу в блоці delay, і після чого їде до наступної зупинки. Ця логіка працює до поки автобус не під'їде до останньої зупинки, де потрапляє до блоку dropoff. На цьому етапі з автобуси виходять всі пішоходи і автобус від'їжджає від зупинки, їде до кінця дороги і потрапляє в блок CarDispose.

Логіка руху автобусів показано на рис 3.14.

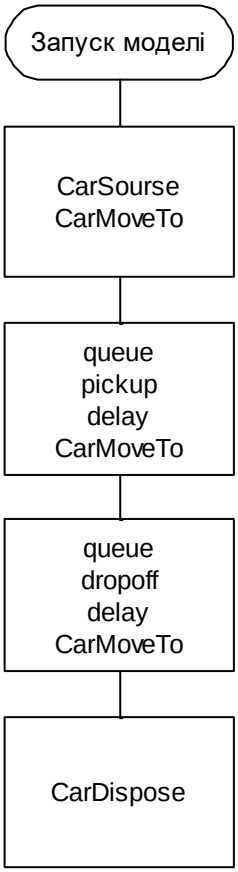


Рисунок 3.14 – Схема логіки автобусів

Логіка пішоходів така: пішоходи з'являються в області, яка вказана в блоці pedSourse. Після цього потрапляють в блок pedGoTo, в властивостях якого вказано, куди повинні йти пішоходи. Якщо через зупинку проїжджає

декілька різних автобусів, то пішохід обирає, в який автобус йому сісти. І так на всіх зупинках, окрім останньої. На останній зупинці всі пішоходи виходять з автобуса і йдуть до місця, яке вказано в блоці pedGoTo, після чого потрапляють в блок pedSink, який видаляє пішохода з системи.

Логіка руху пішоходів показано на рис 3.15.

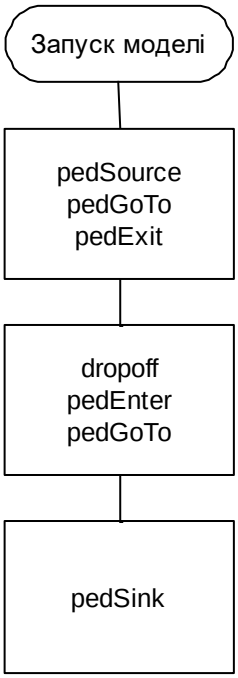


Рисунок 3.15 – Схема логіки пішоходів

Всі автомобілі рухаються за правилами дорожнього руху.

Для зручності проведення експериментів було додано елементи editbox. Один з таких елементів показано на рис 3.16 – 3.17.

Рисунок 3.16 – Вигляд елемента editbox

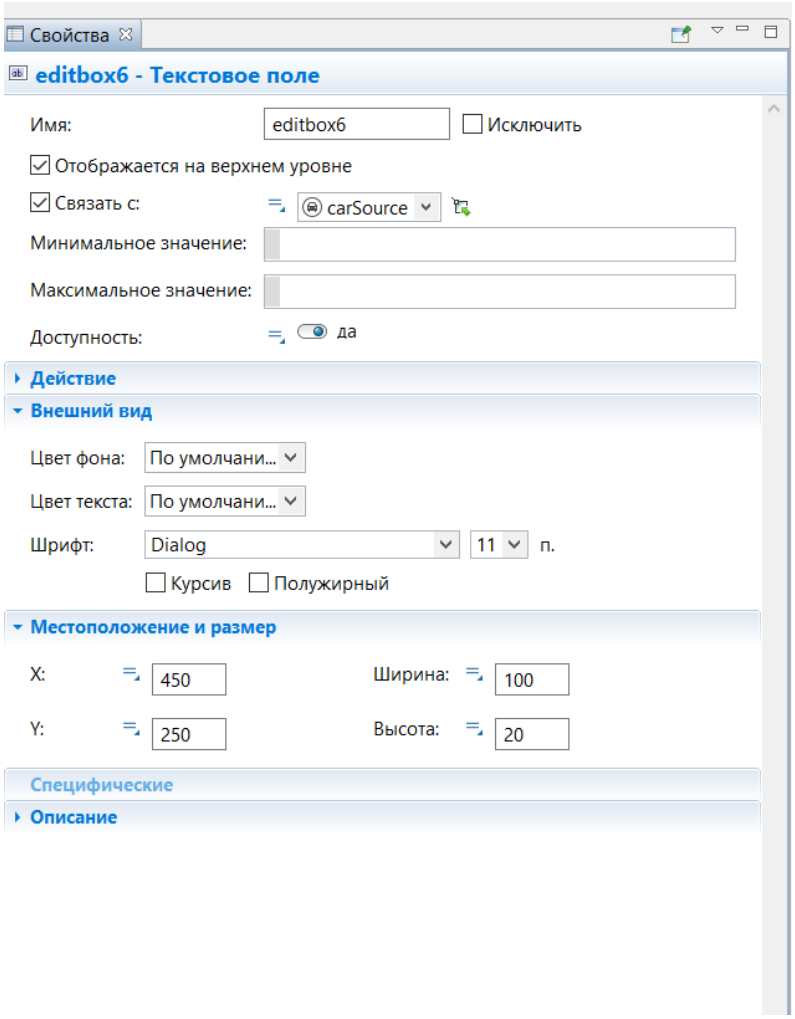


Рисунок 3.17 – Властивості блоку editbox

Такі блоки потрібні для того, щоб у процесі експерименту була можливість змінювати параметри (наприклад параметр появи кількості автомобілів у системі).

У системі присутні елементи Traffic Light, які вже були описані. Налаштування таких блоків було повністю взято з реального життя для чистоти експерименту. Приклад налаштування показано на рис 3.18.

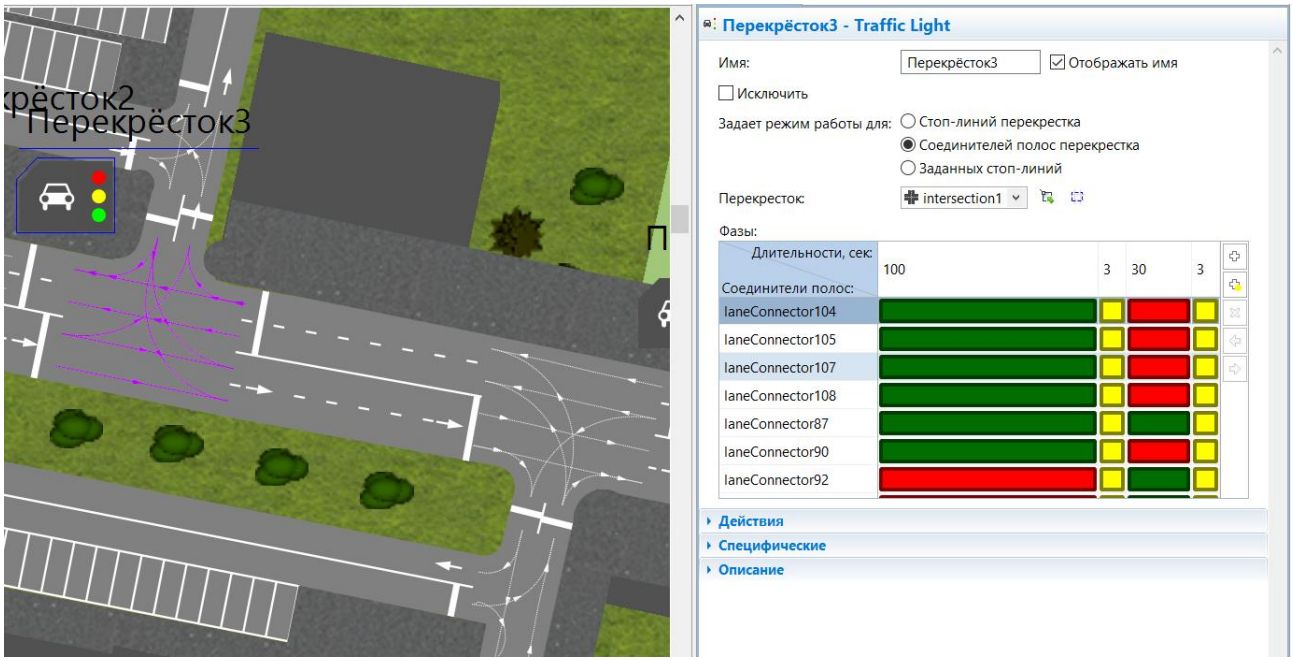


Рисунок 3.18 – Властивості блоку Traffic Light

Основна карта моделі і логіки моделі показано на рис 3.19.

Рисунок 3.19 – Дорожня карта моделі

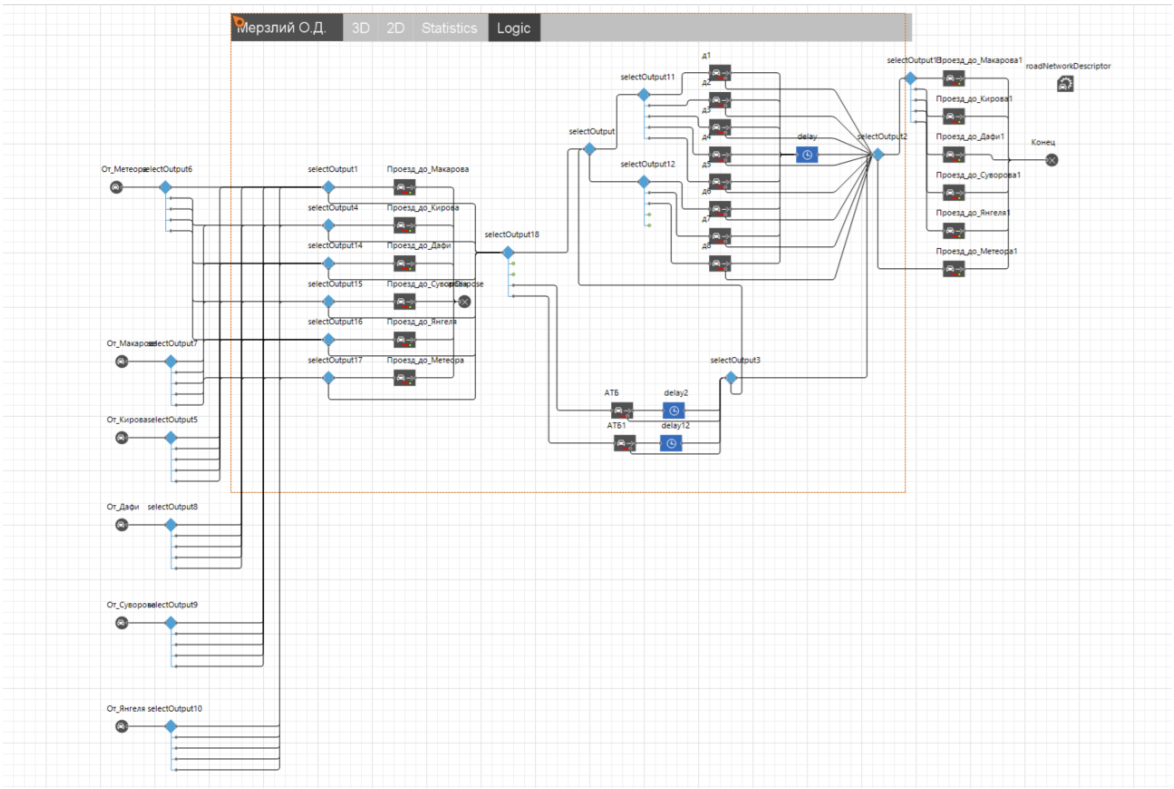


Рисунок 3.20 – Логіка проїзду по моделі

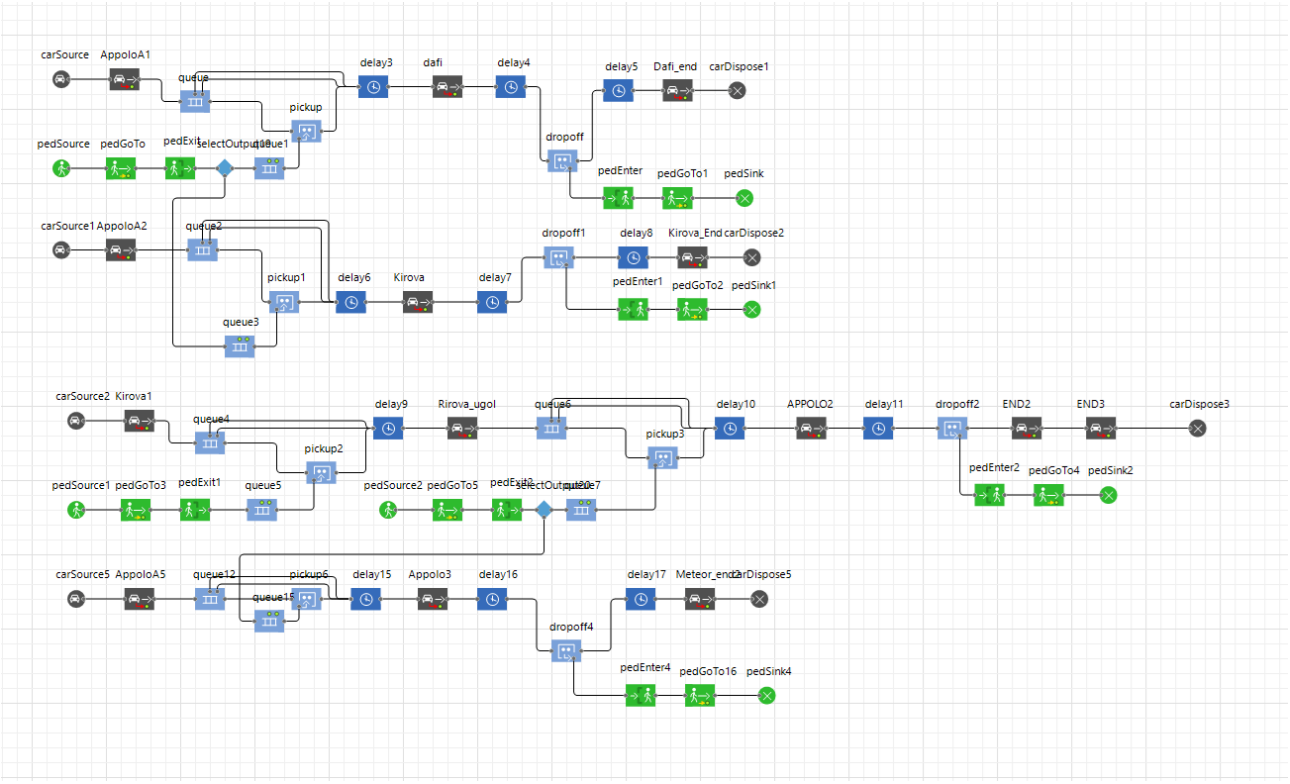


Рисунок 3.21 – Логіка поведінки агентів у моделі

3.7 Висновки

У ході проектування та розроблення програмного продукту «Імітаційна модель транспортних потоків» було обрано середовище AnyLogic. Розроблена логіка руху транспорту і пішоходів. Налаштовані світлофори. Розроблені діаграми та гістограми для збору статистики. Розроблений програмний продукт відповідає усім вимогам.

4 ДОСЛІДЖЕННЯ АВТОМОБІЛЬНИХ ПОТОКІВ

4.1 Підготовка до експерименту

Перед початком експерименту потрібно перевірити логіку роботи моделі. Перевірити усі зв’язки, правильність налаштування світлофорів і об’єктів в цілому. Перевірити правильність налаштування часових характеристик. Перевірити кількість вільної оперативної пам’яті (потрібно не менше 4 гігабайт). Перевірити навантаження процесору (не більше 20%).

Експеримент повинен показати, які фактори більше за все впливають на появу пробок і збільшення навантаження на транспортну мережу. Для більш точних показників, кожен експеримент буде проводитись 5 разів. Час тестування – 1 година.

4.2 Проведення експерименту

4.2.1 Експеримент 1

Мета: провести експеримент у звичному режимі роботи транспортної мережі, без чинників, які впливають на систему.

Основні вхідні данні приведено у табл 4.1.

Таблиця 4.1 – Кількість авто у системі

Напрямок	Кількість появи легкових авто
От_Метеора От_Макарова	200
От_Кирова От_Дафи	300
От_Суворова	30
От_Янгеля	70

Кількість пішоходів у системі – 300 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Результати експериментів показано на рис 4.1 – 4.2.

Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	210,09	364,31	449,94	601,10	50,79	1043	81,79
2	206,42	357,69	496,32	558,75	45,20	1012	71,88
3	204,65	428,41	497,30	597,55	48,99	977	75,44
4	220,92	354,39	467,93	674,08	55,88	973	73,19
5	201,57	414,41	458,30	652,60	54,79	1010	79,80
Середнє значення:	208,73	383,84	473,96	616,82	51,13	1003,00	76,42

Рисунок 4.1 – Результати експериментів

Маємо статистику роботи моделі, для порівняння з іншими експериментами.

По динаміці проїзду авто можна побачити, що модель працює стабільно, без просадок, стрімкого росту навантаження. Результати одного з експериментів показано на рис 4.2.

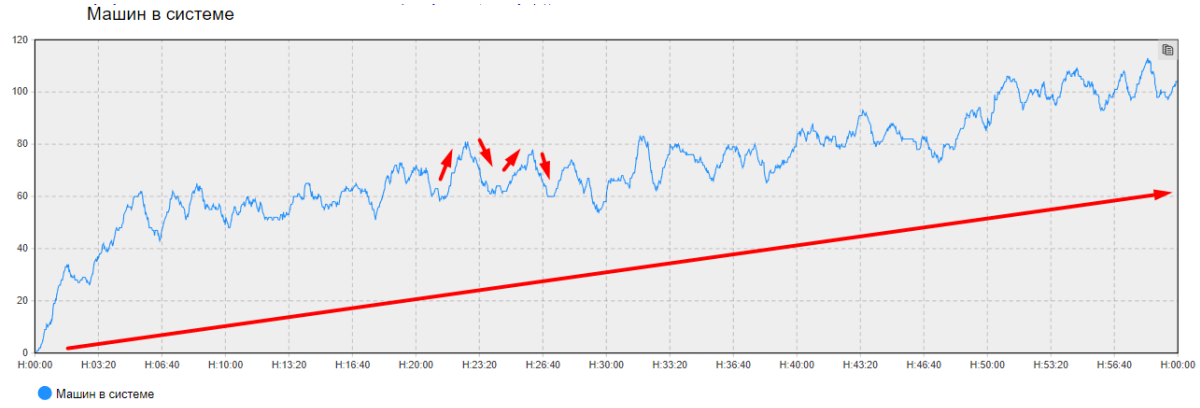


Рисунок 4.2 – Динаміка проїзду авто

Висновок: при розборі графіку проїзду можна побачити, що є зростання і падіння тренду, що є коректним. Основне зростання зумовлено тим, що у системі є місця для паркування авто. Такі авто теж рахуються, як «Авто у системі».

Ці результати будемо вважати за результати правильної роботи моделі.

4.2.2 Експеримент 2

Мета: збільшити кількість авто по кожному основному напрямку на 100 авто, імітуючи «Година пік».

Основні вхідні данні представлено у табл 4.2.

Таблиця 4.2 – Кількість авто у системі

Напрямок	Кількість появи легкових авто
От_Метеора От_Макарова	300
От_Кирова От_Дафи	400
От_Суворова	30
От_Янгеля	70

Кількість пішоходів у системі – 300 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Результати експериментів показано на рис 4.3 – 4.4.

A	B	C	D	E	F	G	H
Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	393,16	630,12	624,25	869,98	42,95	991	148,34
2	391,12	611,41	666,03	909,63	65,56	1228	143,22
3	440,48	668,03	681,88	912,61	64,99	1199	145,18
4	441,55	663,61	713,30	903,59	73,78	1194	154,58
5	520,41	867,42	778,30	1009,31	86,39	1035	148,86
Середнє значення:	437,34	688,12	692,75	921,02	66,734	1129,40	148,04

Рисунок 4.3 – Результати експериментів

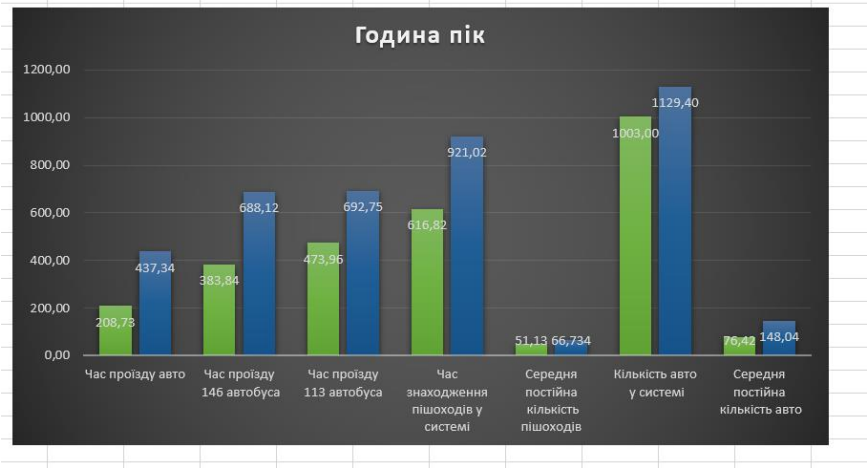


Рисунок 4.4 – Порівняння експериментів

Згідно з рис. 4.4 можна побачити стрімкий ріст часу кожного параметру, у деяких ріст понад 45%.

Висновок: зі збільшенням кількості автомобілів у системі, збільшується і час проходження системи транспортом. Збільшується час очікування і проходження системи для пішоходів так, як автобуси стоять у заторах і не можуть вчасно приїхати до зупинки.

4.2.3 Експеримент 3

Мета: збільшити кількість пішоходів у системі до 1500 чоловік. Кількість авто залишити незмінним.

Основні вхідні данні представлено у табл 4.3.

Таблиця 4.3 – Кількість авто у системі

Напрямок	Кількість появи легкових авто
От_Метеора От_Макарова	200
От_Кирова От_Дафи	300
От_Суворова	30
От_Янгеля	70

Кількість пішоходів у системі – 1500 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Результати експериментів показано на рис 4.5 – 4.6.

Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	274,37	478,47	531,08	804,38	152,79	1021	152,79
2	233,06	361,95	404,95	745,18	155,61	982	69,88
3	289,10	588,18	670,21	862,75	157,99	1002	94,44
4	292,23	548,28	648,82	820,35	154,88	1005	92,19
5	293,19	360,92	544,92	841,61	156,79	1055	97,80
Середнє значення:	276,39	467,56	560,00	814,85	155,61	1013,00	101,42

Рисунок 4.5 – Результати експериментів

Рисунок 4.6 – Порівняння експериментів

Через збільшення потоку пішоходів, час очікування збільшився на 20%, а постійна середня кількість пішоходів у системі > 150%.

Висновок: Через збільшення кількості пішоходів, незмінності інтервалу руху автобусів, час очікування пішоходів збільшується.

4.2.4 Експеримент 4

Мета: Збільшити кількість пішоходів у системі до 1500 чоловік. Кількість авто залишити незмінним. Зменшити максимальну місткість автобусів до 25 місць (імітувати застарілу автотранспортну систему).

Основні вхідні данні представлено у табл 4.4.

Кількість пішоходів у системі – 1500 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 25.

Результати експериментів показано на рис 4.7 – 4.8.

Таблиця 4.4 – Кількість авто у системі

Напрямок	Кількість появи легкових авто
От_Метеора От_Макарова	200
От_Кирова От_Дафи	300
От_Суворова	30
От_Янгеля	70

Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	196,99	310,08	455,25	1273,44	567,73	992	66,03
2	196,97	274,54	486,02	1235,59	505,51	961	62,02
3	220,02	315,02	450,17	1268,65	529,69	972	69,89
4	186,23	353,64	377,48	1267,08	513,88	1027	77,72
5	173,25	278,46	354,46	1215,54	511,36	999	61,71
Середнє значення:	194,69	306,35	424,67	1252,06	525,63	990,20	67,47

Рисунок 4.7 – Результати експериментів

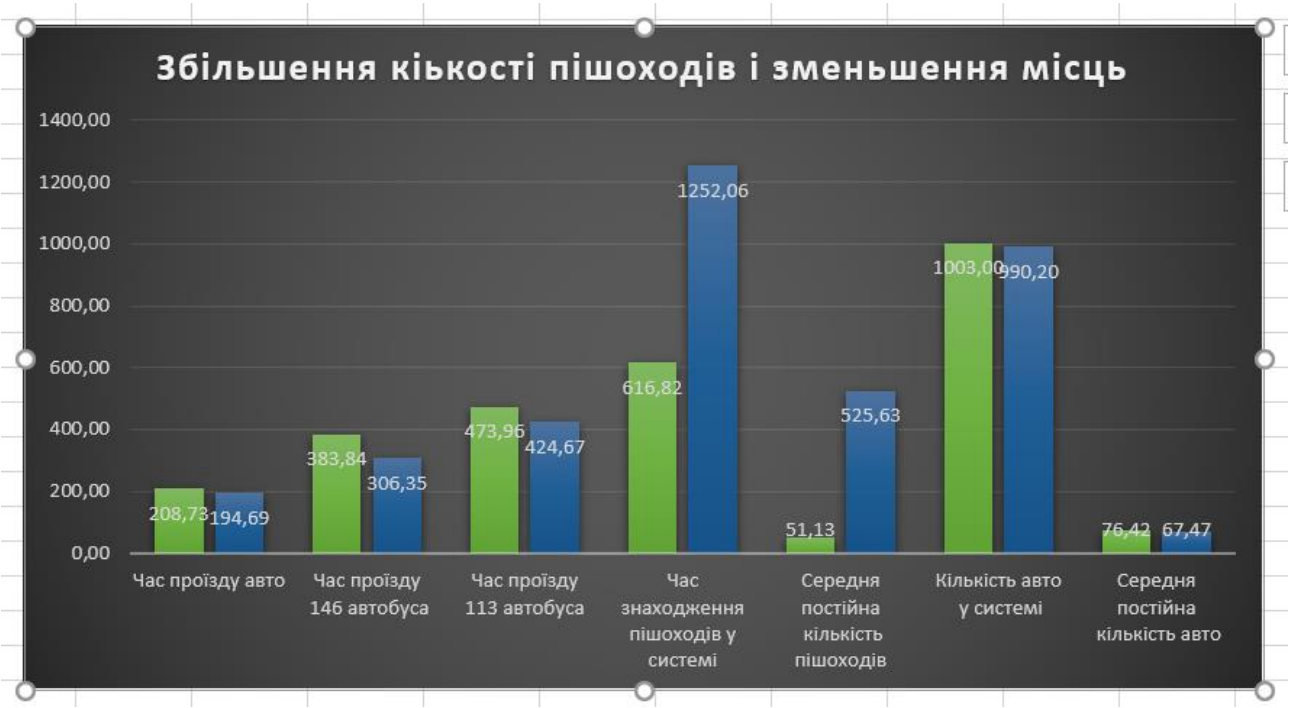


Рисунок 4.8 – Порівняння експериментів

Через збільшення потоку пішоходів і зменшення місткості автобусів, час очікування збільшився більше ніж на 100%, а постійна середня кількість пішоходів у системі збільшилось у 10 разів.

Висновок: Через збільшення кількості пішоходів, незмінності інтервалу руху автобусів і «застарілій» автобусній системі, час очікування пішоходів критично збільшується.

4.2.5 Експеримент 5

Мета: Збільшити кількість пішоходів у системі до 1500 чоловік. Кількість авто збільшити, імітуючи «Годину пік» Додати нову полосу тільки для міського транспорту.

Основні вхідні данні представлено у табл 4.5.

Таблиця 4.5 – Кількість авто у системі

Напрямок	Кількість появи легкових авто
От_Метеора От_Макарова	300
От_Кирова От_Дафи	400
От_Суворова	30
От_Янгеля	70

Кількість пішоходів у системі – 1500 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Результати експериментів показано на рис 4.9 – 4.10.

Результати цього експерименту будемо порівнювати з експериментом №2.

Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	379,23	166,61	192,83	603,61	354,95	1003	142,44
2	406,12	181,51	194,81	638,83	362,13	1128	147,35
3	432,31	178,28	193,14	669,22	375,87	1139	141,23
4	421,15	177,97	207,16	611,56	334,19	1194	151,14
5	513,23	165,84	201,19	610,31	340,21	1135	149,31
Середнє значення:	430,41	174,04	197,83	626,71	353,4704	1119,80	146,29

Рисунок 4.9 – Результати експериментів

Рисунок 4.10 – Порівняння експериментів

Як бачимо, з виділенням окремої смуги для міського транспорту навіть при збільшенні кількості автомобілів, проїзд по системі пішоходів набагато швидше, і порівнюючи з 4 експериментом можна побачити значні зміни, направлені у сторону покращення роботи перевезення пасажирів і роботи міського транспорту.

Висновок: через збільшення кількості пішоходів, незмінності інтервалу руху автобусів, але з побудовою окремої смуги для міського транспорту, спостерігається покращення у роботі міського транспорту. Якщо зменшити

інтервал між автобусами у «Годину пік», збільшити кількість місць у транспорті, то можна досягти швидкого проїзду по місту.

4.2.6 Експеримент 6

Мета: Усі вхідні данні залишити за замовчуванням. Зімітувати повне відключення світлофорів

Основні вхідні данні представлено у табл 4.7.

Таблиця 4.6 – Кількість авто у системі

Напрямок	Кількість появи легкових авто
От_Метеора От_Макарова	200
От_Кирова От_Дафи	300
От_Суворова	30
От_Янгеля	70

Кількість пішоходів у системі – 300 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Результати експериментів показано на рис 4.11 – 4.12.

Результати цього експерименту будемо порівнювати з експериментом №1.

Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	350,19	714,76	474,95	711,90	60,84	1200	250,79
2	361,42	449,65	477,31	526,76	57,32	1114	156,15
3	396,52	574,41	636,13	673,21	56,79	1183	162,44
4	428,52	542,39	426,18	630,54	51,38	1229	147,34
5	449,13	623,11	513,10	652,10	54,79	1010	199,20
Середнє значення:	397,15	580,86	505,53	638,90	56,23	1147,20	183,18

Рисунок 4.11 – Результати експериментів

Рисунок 4.12 – Порівняння експериментів

Через те, що перехрестя не регулювались світлофорами, на деяких ділянках дороги, проїзд авто став швидше, а на деяких навпаки збільшився час очікування у пробках.

Висновок: Якщо не регулювати деякі перехрестя, то є велика вірогідність появи в таких місцях заторів і підвищеного шансу появи ДТП.

4.3 Результати експерименту

Під час проведення експериментів, було виявлено, що при звичайній роботі транспортної мережі, швидкість проїзду усього виду транспорту задовільний. При появі чинників, які створюють велике навантаження, збільшується час проїзду усього виду транспорту. Було виявлено і виділено такі фактори, які можуть вплинути на оптимальну роботу мережі навіть під великим навантаженням, а саме:

1. Для уникнення великого скупчення людей на зупинках:

– у час, коли трафік починає збільшуватися (у ранці і у ввечері) зменшити інтервал між автобусами. Тим саме час очікування на зупинках зменшиться, а бюджет автопарку збільшиться;

– збільшити кількість місць у автобусі (оновлення автопарку);

– надавати можливість пішоходам використовувати альтернативний транспорт, наприклад електросамокати;

– організація окремих полос для руху міського транспорту;

– побудова метро.

2. Для уникнення пробок на дорогах:

– у час, коли трафік починає збільшуватися (у ранці і у ввечері) змінювати фази перемикання світлофорів таким чином, щоб навантаженні ділянки дороги проїжджалися швидше;

– заохочувати автомобілістів пересаджуватися на міський транспорт у годину пік;

– розширювати основні навантажені дороги/перехрестя.

4.4 Висновки

Під час проведення експериментів, було виявлено, що при звичайній роботі транспортної мережі, швидкість проїзду усього виду транспорту задовільний. При появі чинників, які створюють велике навантаження, збільшується час проїзду усього виду транспорту. Було виявлено і виділено фактори, які можуть вплинути на оптимальну роботу мережі навіть під великим навантаженням.

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів і засобів, спрямованих на збереження здоров'я та працездатності людини в процесі праці [16].

Відповідно до статті 18 Закону України "Про охорону праці", працівник зобов'язаний «знати і виконувати вимоги нормативних актів про охорону праці, правила поводження з машинами, механізмами, устаткуванням та іншими засобами виробництва, користуватися засобами колективного та індивідуального захисту, проходити у встановленому порядку попередні та періодичні медичні огляди» [16].

5.1 Вимоги безпеки

5.1.1 Вимоги під час експлуатації ПК

Обладнання й організація робочих місць з моніторами, ПК і ноутбуками мають забезпечувати відповідність усіх елементів робочого місця та їх розміщення ергономічним вимогам наказу від 14.02.2018, №207 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», характеру й особливостям діяльності. Конструкція робочого місця має забезпечити підтримання оптимальної робочої пози й оптимальне розміщення на робочій поверхні використовуваного обладнання (екрану, клавіатури, принтера) та документів.

5.1.2 Режим праці та відпочинку.

При організації праці, пов'язаної з використанням ПК, для збереження здоров'я працівника передбачаються:

– перерви для відпочинку і вживання їжі;

– перерви для відпочинку й особистих потреб;

– додаткові перерви.

5.1.3 Аналіз шкідливих та небезпечних виробничих факторів

Поява і використання електронних обчислюваних машин (ЕОМ) у різні галузі виробництва дуже позитивно вплинуло на якість, умови і продуктивність праці. Робота за монітором, ЕОМ, персональним комп’ютером (ПК) або ноутбуком має низку чинників, які у разі порушення правил експлуатації можуть негативно впливати на стан здоров'я користувачів.

Основним впливом таких пристроїв є зоровий дискомфорт. Він проявляється як біль в очах, почервоніння очних яблук і повік, головний біль, швидка втома, відчуття піску в очах. Кістково-м'язовий дискомфорт проявляється у вигляді болю різної сили у м'язах та суглобах, оніміння тощо. Частіше кістково-м'язовий дискомфорт проявляється у людей старшої вікової категорії. Не слід забувати, що ЕОМ та монітори можуть бути чинником захворювання шкіри. Вони проявляються у вигляді появи лущення, висипу, деяких видів дерматитів. Стресові чинники, до яких належать несприятливі умови та режим праці, здібності працівника, зміст праці, , умови життя, звички, можуть стати причиною виникнення фізіологічних і психологічних змін, погіршення здоров'я та змін у поведінці людини.

Користувачі ЕОМ повинні знати і розуміти потенційно шкідливі та небезпечні чинники під час роботи за монітором, ПК, ноутбуком та виконувати вимоги НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин» і ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами ЕОМ» та інших нормативних документі» щодо виключення або зменшення їх негативного виливу.

Аналіз шкідливих та небезпечних виробничих факторів проводять згідно з ДСТУ 2293:2014. Небезпечні та шкідливі фактори – це такі фактори, які призводять до раптового погіршення здоров’я людини, або навіть смерті, під час її трудової діяльності.

Згідно з класифікацією ДСТУ 2293:2014 до числа небезпечних та шкідливих факторів, що виникають при роботі з ПК слід віднести:

– нервово-психічні перевантаження (розумове перевантаження, монотонність праці, емоційні перевантаження);

– невідповідність параметрів мікроклімату робочої зони санітарним нормам;

– несприятливе забарвлення стін та підлоги, віддзеркалення;

– фізичне перевантаження;

– недостатня або надмірна освітленість робочої зони;

– робота з комп’ютером.

5.1.4 Вимоги до приміщень, розміщення в них моніторів, ПК, ноутбуків та організації лінії робочих місць

Планувальні рішення будівель і приміщень для роботи з моніторами, ПК, ноутбуками мають відповідати вимогам чинних нормативних актів, зокрема ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою», наказом МНС України від 15.06.2016 і мати ступінь вогнестійкості не нижчу II.

Заборонено розміщувати робочі місця з моніторами, ПК та ноутбуками у підвальних приміщеннях, на цокольних поверхах, поряд з приміщеннями, в яких рівні шуму та вібрації перевищують допустимі значення (поряд з механічними цехами, майстернями тощо), з мокрими виробництвами, з вибухопожежонебезпечними приміщеннями категорій А і Б, а також над такими приміщеннями або під ними.

Площу приміщень визначають із розрахунку, що на одне робоче місце вона має становити не менше ніж 6 м2, а об'єм не менше ніж 20 м2 з урахуванням максимальної кількості осіб, які одночасно працюють у зміні [17].

Приміщення мають бути оснащені природним і штучним освітленням відповідно до ДБН В.2.5-28-2006. Приміщення мають бути обладнані

системами водяного опалення, кондиціонування або припливно-витяжною вентиляцією відповідно до ДСТУ EN 13053:2013 «Системи вентиляції та кондиціювання повітря». Заземлені конструкції приміщення (батареї опалення, водопровідні труби, кабелі з заземленим відкритим екраном тощо) надійно захищені діелектричними щитками або сітками від випадкового дотику [17].

Для всіх споруд і приміщень, у яких експлуатують монітори, ПК і ноутбуки визначають категорію з вибухопожежної та пожежної безпеки відповідно до НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будівель та зовнішніх установок по вибухопожежній та пожежної небезпеки» та клас вибухонебезпепечних зон відповідно до НПАОП 0.00-1.32-01 «Правила будови електроустановок. Електрообладнання спеціальних установок». Відповідні позначення наносять на вхідні двері приміщення. Крім того, приміщення з моніторами, ПК і ноутбуками оснащують системою пожежної сигналізації з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками з розрахунку 2 од. на кожні 20 м2 площі приміщення. Підходи до засобів пожежогасіння мають бути вільними [17].

Робочі місця з моніторами, щодо світлових прорізів розміщують так, щоб природне світло падало збоку, переважно зліва [17].

Екран монітора і клавіатура мають розміщуватися на оптимальний відстані від очей користувача, але не ближче 600 мм з урахуванням розміру алфавітно-цифрових знаків і символів. Відстань від екрана до ока працівника залежить від діагоналі екрана та наведена у табл 5.1.

Таблиця 5.1 – Відстань від екрана до ока працівника

Діагональ екрана	Відстань від екрана
від 35 до 38 см (14"/15")	від 600 до 700 мм
43 см (17")	від 700 до 800 мм
48 см (19")	від 800 до 900 мм
53 см (21")	від 900 до 1000 мм

Розміщення екрана монітору має забезпечувати зручність зорового спостереження у вертикальній площині під кутом ±30° від лінії зору працівника.

Розміщення принтера або інших пристроїв введення-виведення інформації на робочому місці має забезпечувати добру видимість екрану, зручність ручного керування пристроєм введення-виведення інформації в зоні досяжності моторного поля: по висоті від 900 до 1300 мм, по глибині від 400 до 500 мм. Під матричні принтери потрібно підкладати вібраційні килимки для гасіння вібрації та шуму.

Площа для одного робочого місця за ПК повинна складати не менше 6 м2, а об'єм – не менше 20 м3. Робочі місця з моніторами відносно світлових прорізів повинні розміщуватися так, щоб природне світло падало збоку.

Робочий стіл, як правило, обладнують підставкою для ніг ширшою не менше 300 мм, глибиною не менше 400 мм з можливістю регулювання по висоті в границях 150 мм та кутом нахилу опорної поверхні в границях 20°. Підставка має бути з рифленою поверхнею і бортиком по передньому краю заввишки 10 мм [18].

Робоче сидіння (стілець, крісло) має складатися з таких основних елементів: сидіння, спинки, стаціонарних або змінних підлокітників. Стілець має бути підйомно-поворотним, таким, що регулюється за висотою, кутом нахилу сидіння і спинки, за відстанню спинки до передньою краю сидіння, висотою підлокітників. Крок регулювання для лінійних розмірів – від 15 до 20 мм, для кутових – від 2 до 5°. Зусилля регулювання – не більше 20 Н [18].

Клавіатуру слід розташовувати на поверхні столу або на спеціальній регульованій за висотою робочій поверхні окремій віл столу на відстані від 100 до 300 мм від краю, ближчого до працівника. Кут нахилу клавіатури має бути у границях від 5 до 15° [18].

Розміщення принтера або іншого пристрою вводу-виводу інформації на робочому місці має забезпечувати гарну видимість екрана монітора і зручність ручного керування в зоні досяжності моторного поля [18].

Робочі місця слід оснащувати пюпітром (тримачем) для документів, що легко перемішувати. Його встановлюють вертикально (або з нахилом) на тому ж рівні та відстані від очей користувачів, що і монітор [18].

5.1.5 Вимоги до виробничого середовища приміщень з моніторами, ПК і ноутбуками

Згідно з Гігієнічною класифікацією праці за показниками шкідливості та небезпечності чинників виробничого середовища, тяжкості та напруженості трудового процесу умови праці користувачів ПК і ноутбуків мають відповідати І класу (оптимальним) або ІІ класу (допустимим) умовам праці [18].

Приміщення з моніторами, ПК і ноутбуками мають бути забезпечені природним і штучним освітленням. Коефіцієнт природного освітлення (КПО) має бути не нижчим 1,5%. Розраховують площу світлових прорізів, яка забезпечує нормоване значення КПО в робочій зоні користувачів комп'ютерів, відповідно до ДБН В.2.5-28-2006.

Штучне освітлення має бути загальним, робочим і рівномірним. У випадку, коли робота переважно з документами, допускається додатково використовувати місцеве освітлення. Рівень освітленості в зоні розташування документів має бути в границях від 300 до 500 лк [17].

Систему загального освітлення має бути виконано у вигляді суцільних або переривчатих ліній світильників, що розмішують збоку від виробничих місць (переважно зліва), паралельно лінії зору працівників. Допускається застосовувати світильники таких класів світлорозподілу: світильники прямого світла – П; переважно прямого світла – Н; переважно відбитого світла – В [17].

У разі розташування моніторів, ПК по периметру приміщення лінії світильників мають бути розміщені локально над робочими місцями [17].

Уміст шкідливих речовин у повітрі робочої зони не має перевищувати ГДК відповідно до ГОСТ 12.1.005 – 88 уміст озону не більше 0,1 мг/м3, вміст оксидів азоту – не більше 5 мг/м3. уміст пилу – не більше 4 мг/м3 [17].

Параметри мікроклімату мають відповідати вимогам ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» [17].

Оптимальні величини температури, відносної вологості та швидкості руху повітря для приміщень з моніторами, ПК і ноутбуками наведені у табл 5.2 [17].

Таблиця 5.2 – Оптимальні величини температури, відносної вологості та швидкості руху повітря для приміщень з моніторами, ПК і ноутбуками [17].

Період року	Категорія робіт	Температура повітря, °С	Відносна вологість повітря, %	Швидкість руху повітря, м/с
холодний	Легка-1 а	від 22 до 24	від 40 до 60	0,1
	Легка-16	від 21 до 23	від 40 до 60	0,1
теплий	Легка-1а	від 23 до 25	від 40 до 60	0,1
	Легка-16	від 22 до 24	від 40 до 60	0,2

Рівні іонізації повітря приміщень під час роботи з моніторами, ПК і ноутбуками наведено у табл 5.3.

Таблиця 5.3 – Рівні іонізації повітря приміщень під час роботи з моніторами, ПК і ноутбуками [17].

Рівні іонізації повітря	Кількість іонів у 1 $см^3$ повітря	
	n^+	n^-
Мінімально необхідні	400	600
Оптимальні	Від 1500 до 3000	Від 3000 до 5000
Максимально допустимі	50000	50000

Допустимі рівні звуку, еквівалентні рівні звуку, рівні звукового тиску в октавних смугах частот представлений в табл 5.4.

Таблиця 5.4 – Допустимі рівні звуку, еквівалентні рівні звуку, рівні звукового тиску в октавних смугах частот [17].

Види трудової діяльності	Рівні звукового тиску, дБ, в октавних смугах із середньо геометричними частотами, Гц									Рівні звуку, еквівалентні рівні звуку, дБ/дБекв
	31.5	63	125	250	500	1000	2000	3000	4000	
Програмісти ПК	86	71	61	54	49	45	42	40	38	50
Оператори в залах оброблення інформації на ПК та оператори комп'ютерного набору	96	83	74	68	63	60	57	55	54	65

Для підтримання оптимальних значень параметрів повітря робочої зони потрібно застосовувати вентиляцію приміщень, кондиціонування повітря, використовувати установки або прилади зволожування та штучної іонізації [17].

У приміщеннях з ПК рівні звукового тиску, рівні звуку та еквівалентні рівні звуку мають відповідати вимогам ДСН 3.3.6.037-99 та ДСанПіН 3.3.2.007 98. Устаткування, яке є джерелом шуму, слід розташовувати поза приміщеннями з ПК і ноутбуками [17].

Значення напруженості електромагнітних полів на робочих місцях із моніторами мають відповідати нормативним значенням ГОСТ 12.1.006-84 і ДСанПіН 3.3.2.007-98 [17].

Допустимі рівні електромагнітних випромінювань радіочастотного діапазону наведено у табл 5.5.

Таблиця 5.5 – Допустимі рівні електромагнітних випромінювань радіочастотного діапазону [17].

Діапазон частот, МГц	Допустимі рівні ЕМП	
	Електрична складова *Е, В/м*	Магнітна складова Н, А/м
від 0.06 до3,0	50	5
від 3.0 до 30,0	20	-
від 30.0 до 50.0	10	0.3
від 50,0 до 300,0	5	-

5.2 Дотримання правил безпеки при збиранні статистики для автотранспортної моделі

При зборі статистики для імітаційної моделі слід пам’ятати про правила дорожнього руху, а саме:

– переходити дорогу необхідно тільки на зелене світло світлофора. Якщо перехрестя регулюється регулювальником, то слід підкорятися його сигналам. Пішоходи не повинні затримуватися чи зупинятися на дорозі, переходити перехрестя по діагоналі. За відсутності і світлофора, і регулювальника перехрестя є нерегульованим. Переходити його слід біля знаку «Пішохідний перехід» або по дорожній розмітці «зебра»;

– після висадки з транспорту автобус і тролейбус обходять позаду, а трамвай попереду;

– не вибігати на проїзну частину з тротуару, можна лише спокійно зійти, попередньо оцінивши ситуацію;

– ходити лише тротуарами, а якщо вони відсутні – по узбіччю, обов'язково повернувшись обличчям до транспорту, що рухається, тоді не тільки водій бачитиме пішохода, а й пішохід водія;

– на дорозі відстань до автомобіля залежить від швидкості, з якою той рухається, отже, навчіться розраховувати, коли до авто далеко, а коли – близько. При цьому пам'ятайте, що навіть при швидкості 60 км/год гальмовий шлях автомобіля буде довшим за 15 метрів.

5.3 Дії працівників (персоналу) в аварійних (надзвичайних) ситуаціях.

Типову інструкцію розроблено відповідно до ст. 130 Кодексу цивільного захисту України:

– при загрозі хімічного ураження оповіщаються всі працівники та відвідувачі, які знаходяться на території підприємства;

– вентиляційні установки та кондиціонери терміново виключаються, закриваються вікна, двері, кватирки, приміщення герметизуються. Вихід із будівлі й вхід до неї припиняється до особливого розпорядження адміністрації;

– працівникам видаються засоби індивідуального захисту, одночасно вживаються заходи із забезпечення відвідувачів ватно-марлевими пов’язками;

– відповідальні за забезпечення герметизації приміщень (посада, прізвище), за забезпечення працівників та відвідувачів засобами індивідуального захисту (посада, прізвище);

– при виявленні у приміщенні, де укриваються працівники, хімічно небезпечної речовини працівники повинні вийти (вказати куди) або з дозволу адміністрації залишити зону забруднення. Виходити із зони необхідно тільки у засобах індивідуального захисту та рухатися в напрямку, перпендикулярному напрямку вітру;

– при виникненні пожежі на підприємстві всі працівники зобов’язані суворо виконувати вимоги Інструкції з пожежної безпеки, евакуацію проводити згідно з Планом евакуації;

– відповідальність за дотримання заходів пожежної безпеки та організацію дій персоналу при загрозі або виникненні пожежі покладається на (посада, прізвище);

– при радіоактивному забрудненні території підприємства або при загрозі забруднення всі працівники повинні уважно слідкувати за мовним повідомленням управління з питань надзвичайних ситуацій, яке передається по радіо та телебаченню після попереджувального сигналу «Увага всім», за інформацією інших засобів масової інформації про обстановку в місті та суворо виконувати рекомендації із захисту від радіоактивного зараження;

– працівник (посада, прізвище) організовує на території підприємства контроль за радіаційною обстановкою за допомогою побутового дозиметру (називається тип приладу) та постійно інформує про результати вимірювань адміністрацію підприємства, управління з питань надзвичайних ситуацій;

– при перевищенні гранично припустимих норм опромінення організується облік доз опромінювання. Відповідальний за виконання цього заходу (посада, прізвище);

– скорочується до мінімуму вхід у будівлю та вихід з неї. Контроль за дотриманням режиму поводження й роботи працівників, який дозволяє максимально понизити наслідки радіоактивного опромінення, покладається на (посада, прізвище);

– при загрозі або виникненні катастрофічних стихійних лих працівник підприємства по розпорядженню адміністрації повинен зупинити виробництво, виконати необхідні протипожежні заходи, відключити від електромережі електрообладнання, підготуватися до евакуації або вивезення до безпечного місця найбільш цінних матеріальних засобів;

– контроль за обстановкою на території підприємства при стихійних лихах і за вжитими заходами захисту персоналу покладається на (посада, прізвище);

– якщо з’явилися постраждалі, їм надається перша медична допомога із залученням санітарних дружин або постів підприємства, вживаються заходи з госпіталізації постраждалих до медичних закладів;

– працівник (посада, прізвище) постійно слідкує за інформацією, яку надає управління з питань надзвичайних ситуацій, про обстановку в місті та доводить її до адміністрації й персоналу підприємства;

– при надходженні анонімної інформації про загрозу на території підприємства або поблизу нього терористичного акту працівник, який прийняв її, повинен терміново доповісти керівнику підприємства та до правоохоронних органів і діяти згідно з розпорядженнями та рекомендаціями.

5.4 Висновки

У цьому розділі було висвітлено питання і вимоги охорони безпеки під час використання ПК, виробничого середовища приміщень з моніторами і ноутбуками. Дотримання правил безпеки при збиранні статистики для автотранспортної моделі. Дії працівників (персоналу) в аварійних (надзвичайних) ситуаціях.

ВИСНОВКИ

Під час аналізу предметної сфери було висвітлено основні проблеми питання оптимізації дорожньої системи і імітаційного моделювання, виконано широкий огляд програмних аналогів та їх порівняння.

Проаналізовано технологічну платформу, наведено аргументацію використання обраного програмного рішення, а також розроблено комплексну модель транспортного потоку.

Було проаналізовано економічну ефективність розроблюваної системи. Виконані розрахунки свідчать про економічну доцільність розробки та використання імітаційного підходу.

Використання сучасних гнучких методологій розробки дозволить і надалі розширювати функціональність системи або вносити зміни до її поточної функціональності.

Під час проведення експериментів, було виявлено, що при звичайній роботі транспортної мережі, швидкість проїзду усього виду транспорту задовільний. При появі чинників, які створюють велике навантаження, збільшується час проїзду усього виду транспорту. Було виявлено і виділено такі фактори, які можуть вплинути на оптимальну роботу мережі навіть під великим навантаженням, а саме:

Для уникнення великого скупчення людей на зупинках:

– у час, коли трафік починає збільшуватися (у ранці і у ввечері) зменшити інтервал між автобусами. Тим саме час очікування на зупинках зменшиться, а бюджет автопарку збільшиться;

– збільшити кількість місць у автобусі (оновлення автопарку) ;

– надавати можливість пішоходам використовувати альтернативний транспорт, наприклад електросамокати;

– організація окремих полос для руху міського транспорту;

– побудова метро.

Для уникнення пробок на дорогах:

– у час, коли трафік починає збільшуватися (у ранці і у ввечері) змінювати фази перемикання світлофорів таким чином, щоб навантаженні ділянки дороги проїжджалися швидше;

– заохочувати автомобілістів пересаджуватися на міський транспорт у годину пік;

– розширювати основні навантажені дороги/перехрестя.

Реалізовано програмне середовище, здатне імітувати роботу транспортної мережі. Надає можливість змінювати параметри для імітації різних ситуацій, які можуть впливати на роботу мережі.

Велику увагу було приділено тестуванню та відлагодженню системи. Використання сучасних інструментів тестування та налагодження й використання різних підходів та методів тестування, що найкраще підходять до тестування кожного окремого методу, дозволили зробити систему надійною та стійкою до можливих помилок.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1 Імітаційне моделювання. URL: http://ni.biz.ua/5/5_3/5_32577_imitatsionnoe modelirovanie.html (дата звернення 19.09.2021).

2 Дослідження транспортних потоків в аспекті заторових станів дорожнього руху : Монографія / В.М. Першаков та ін. Київ : 2015. 32с.

3 Як Ford використовує імітаційне моделювання при розробці безпілотних автомобілів. URL: https://nfp2b.ru/2020/04/02/kak-ford-ispolzuet-imitatsionnoe-modelirovanie-pri-razrabotke-bespilotnyh-avtomobilejj/ (дата звернення 29.09.2021).

4 Система Actor Pilgrim URL: http://simulation.su/static/actor-pilgrim-full-info.html (дата звернення 29.09.2021).

5 AnyLogic. URL: https://www.anylogic.ru/ (дата звернення 01.09.2021).

6 Об'єктно-орієнтоване програмування на C++ / Ілья Труб. Москва :УДК, 2006. 330с.

7 Файловий архів студентів. URL: https://studfile.net/ (дата звернення 09.09.2021).

8 Студми. Навчальні матеріали для студентів. URL: https://studme.org/ (дата звернення 01.09.2021).

9 Імітаційне моделювання: Посібник / Л.Ф. Вьюненко. Москва : 2016. 266с.

10 Букліб URL: https://buklib.net/books/24846/ (дата звернення 09.09.2021).

11 Знаймо, Дискретно-подієве моделювання URL: https://znaimo.com.ua/Дискретно-подієве_моделювання (дата звернення 09.09.2021).

12 СтудКомЮа URL: https://stud.com.ua/145103/informatika/metodi_agentno_ modelyuvannya (дата звернення 09.09.2021).

13 Агентне моделювання в AnyLogic URL: http://eki.tneu.edu.ua(дата звернення 13.09.2021).

14 Наукова електронна бібліотека URL: https://elibrary.ru. (дата звернення 13.09.2021).

15 Вільна енциклопедія Wikipedia. URL: https://uk.wikipedia.org (дата звернення 09.09.2021).

16 Закон про охорону праці. URL: https://pon.org.ua/ohorona-praci/72-zakon-pro-okhoronu-praci.html (дата звернення 21.11.2021).

17 Бібліфонд. URL: https://www.bibliofond.ru/view.aspx?id=871469 (дата звернення 21.11.2021).

18 Вимоги до приміщень, розміщення в них ВДТ, ЕОМ, ПЕОМ та організації лінії робочих місць URL: https://lektsii.com/2-84145.html (дата звернення 21.11.2021).

19 Міністерство інфраструктури України. URL: https://mtu.gov.ua /timeline/Dorozhne-gospodarstvo.html (дата звернення 21.11.2021).

20 Імітаційне моделювання систем та процесів: Конспект / В.Б. Неруш та ін. Київ : 2012. 115с.

21 Імітаційне моделювання: Методичні вказівки / І.В. Буртняк. Івано-Франківськ : 2015. 97с.

22 Основи імітаційного і статистичного моделювання: Навч. Посібник / Ю.С. Харін та ін. Мінськ : 1997. 288с.

23 Імітаційне моделювання систем – мистецтво та наука: Навч. Посібник / Р. Шеннон : 1978. 418с.

24 Моделювання систем. Інструментальні засоби GPSS World: Навч. Посібник / В.Д. Боев : СПБ, 2004. 346с.

25 Аналітико-імітаційні дослідження систем та мереж масового обслуговування : Монографія / В.М. Задорожний : Омск, 2010. 280с.

26 Formation of recommendations for the selection of types of connections for different types: Monograph / A. Novikov : Krasnoyarsk, 2019. 10p.

27 Methodology for TRITIA transport model : Report / Transport Research Institute, JSC : 2019. 43p.

ДОДАТКИ

ДОДАТОК А

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ЗАТВЕРДЖУЮ
Проректор Дніпровського
національного університету
залізничного транспорту
імені академіка В. Лазаряна
Борис БОДНАР

Розробка програмного продукту «Імітаційна модель транспортних потоків»

Технічне завдання
ЛИСТ ЗАТВЕРДЖЕННЯ
1116130.01206-01 ЛЗ

Завідувача кафедри КІТ
Вадим ГОРЯЧКІН

Керівник розробки
Олександра ГОРБОВА

Виконавець
Олексій МЕРЗЛИЙ

Нормоконтролер
Олена КУРОП'ЯТНИК

2021

АНОТАЦІЯ

Документ 1116130.01206-13 «Імітаційна модель транспортних потоків» є керівництвом для проектування та розробки програмного додатку «Імітаційна модель транспортних потоків», та входить до складу документації на проект.

Розробка імітаційної моделі транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання та поліпшення такої системи в реальному житті. В документі містяться основні вимоги до розробки програмного додатку та його функціональні можливості.

ЗМІСТ

ВСТУП

Через збільшення автотранспортних потоків в транспортній мережі актуальною є проблема їх раціональної організації. Однак з урахуванням впливу різних чинників, таких як завантаженість ділянки дороги, стану дороги, дана задача не може бути вирішена за допомогою аналітичних моделей, заснованих на графових моделях.

Тому актуальна розробка комп'ютерних моделей, що дозволяють врахувати всі перераховані випадкові чинники, і раціонально організувати потоки в автомобільній мережі.

Моделювання є важливим засобом розв'язання багатьох економічних завдань і, зокрема, проведення аналітичного дослідження. Модель – це умовний об'єкт дослідження, тобто матеріальне чи образне відображення реального об'єкта, процесу його функціонування в конкретному середовищі.. Метод моделювання – це конструювання моделі на основі попереднього вивчення об'єкта, визначення його найбільш суттєвих характеристик, експериментальний і теоретичний аналіз створеної моделі, а також необхідне коригування на підставі одержаної інформації

Імітаційна модель – це комп'ютерна програма, яка описує структуру і відтворює поведінку реальної системи в часі. Імітаційна модель дозволяє отримувати докладну статистику про різні аспекти функціонування системи залежно від вхідних даних.

Необхідність розробки виникла у результаті аналізу ринку ПЗ і проблематики питання. Через високу ціну проектування і побудови реальної мережі, було прийнято рішення на створення імітаційної моделі транспортної мережі у відповідності з усіма вимогами для оцінки доцільності будування і раціонального використання коштів.

Ця модель буде корисна Укравтодору для коректної оптимізації і побудови дорожньої системи.

1 ПІДСТАВА ДЛЯ РОЗРОБКИ

Основою для розробки є наказ в.о. ректора Дніпровського національного університету залізничного транспорту імені академіка В. Лазаряна Боднара Б.Є. «Про затвердження тем та призначення керівників дипломних проектів» №01206 ст. від 10.12.2020 р.

Тема проекту: «Дослідження транспортних потоків засобами імітаційного моделювання».

Керівник дипломного проекту: доцент кафедри «КІТ» О.В. Горбова.

2 ПРИЗНАЧЕННЯ РОЗРОБКИ

2.1 Функціональне призначення

Основним функціональним призначенням програмного продукту є надання можливостей імітувати рух автомобілів на проектованій системі. Надати можливість оператору задавати навантаження трафіку на дорогах і щільність потоку. Надати можливість регулювати роботу світлофорів. Надати можливість регулювати час появи авто і громадського транспорту. Надати можливість регулювати кількість людей на зупинках. Збирати статистику та надавати можливість перегляду статистики в форматі графіків та гістограм.

2.2 Експлуатаційне призначення

Імітаційна модель транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання такої системи в реальному житті.

3 ВИМОГИ ДО ПРОГРАМИ

3.1 Вимоги до функціональних характеристик

Вимоги до функціональних характеристик наступні:

– можливість задавати навантаження трафіку на дорогах і щільність потоку;

– можливість регулювати роботу світлофорів;

– можливість регулювати час появи авто і громадського транспорту;

– можливість регулювати кількість людей на зупинках.

До додаткових можливостей відносяться:

– перегляд імітаційної моделі в форматі 2D і 3D;

– можливість зміни деяких показників моделі в процесі її роботи;

– перегляд статистики в форматі графіків та гістограм.

3.2 Вхідні дані

Метод організації вхідних даних полягає в наданні оператору зручного інтерфейсу для введення даних у поля зі строгим дотриманням встановленого формату.

3.3 Вихідні дані

До вихідних даних відносяться:

– результати роботи моделі: кращий час проходження агентом моделі;

– графіки, гістограми.

3.4 Вимоги до надійності

Програмний продукт має забезпечити стійку роботу, коректне виконання своїх основних функцій та вимог. Повинні виконуватися наступні вимоги:

– стійка робота при вводі некоректної вхідної інформації;

– відновлення працездатності програми не пізніше однієї год. після збою;

– наявність резервної копії моделі на зовнішньому носії;

– наявність архівної копії тексту програми на зовнішньому носії;

– час роботи на відмову не менше 1500 години (відмовою системи слід вважати короткочасне порушення робочого стану системи, яке самоусувається при повторній спробі).

3.5 Умови експлуатації

Для нормального функціонування програмного продукту необхідно виконання наступних вимог:

– обчислювальна техніка, на якій буде експлуатуватися програма, повинна знаходитися в приміщенні з температурою повітря від 21° С до 25° С, вологістю повітря 40 – 60%, швидкістю руху повітря не більш 0,2 м/с;

– з метою полегшення надійного функціонування програми періодично проводити резервне копіювання даних на зовнішні носії;

– стан технічних пристроїв повинен задовольняти відповідні норми і вимоги;

– працювати з програмою може людина, яка володіє навичками роботи з комп’ютером на рівні користувача та ознайомлена з керівництвом користувача.

3.6 Вимоги до складу та параметрів технічних засобів

Мінімальна конфігурація, що забезпечить нормальну експлуатацію програмного продукту:

– процесор: Intel Core 2 Duo 2.4 GHz або краще;

– оперативна Пам'ять: 4 GB DDR3 або краще;

– вільний дисковий простір 500 Mb;

– наявність CD/DVD приводу або USB роз’ємну для встановлення необхідного ПЗ;

– монітор з роздільною здатністю екрану 1024x768;

– периферійний печатний пристрій;

– клавіатура;

– маніпулятор «миша».

3.7 Вимоги до інформаційної і програмної сумісності

Імітаційна модель працює у таких ОС:

– Microsoft Windows, 10, x86-32 і x64;

– Microsoft Windows 8, x86-32 і x64;

– Microsoft Windows 7 SP1, x86-32 і x64;

– AppleMac OS X 10.7.3 (Lion) або вище, універсальний;

– SuSELinux, x86-32 (з встановленими GTK +, libwebkitgtk-1.0-0, libudev, libssl 0.9.8 і вище);

– Ubuntu Linux 10,04 або вище, x86-32 (з встановленими GTK +, libwebkitgtk-1.0-0, libudev, libssl 0.9.8 і вище).

Для роботи моделі потрібно Java 2 Standard Edition 8.0 або вище. Повинен бути встановлений користувачем самостійно.

3.8 Вимоги до маркування та упаковки

Програма може зберігатись на жорсткому диску або на знімних носіях (флеш-носії, CD диски). На упаковці повинно бути вказано назву продукту, його серійний номер, версія, вимоги до складу та параметрів технічних засобів, вимоги до інформаційної та програмної сумісності.

Приклад маркування наведено на рис. 1.1.

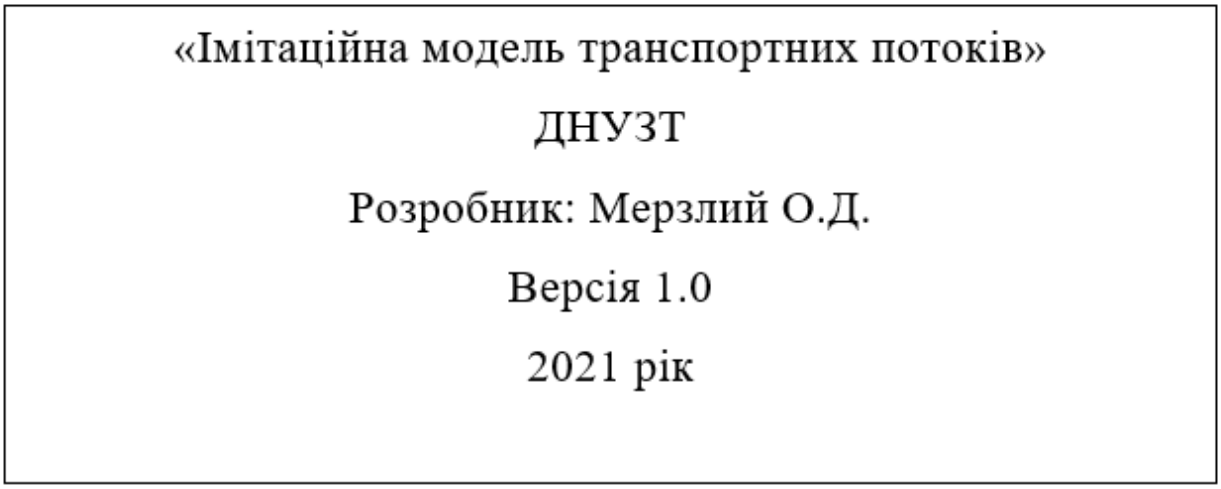


Рисунок 1.1 – Приклад маркування

3.9 Вимоги по транспортуванню та зберіганню

Транспортування програмного продукту може виконуватися шляхом його переносу на знімних інформаційних носіях чи по інформаційних каналах зв’язку через мережу Інтернет. Не допускається механічний вплив на носії. Інформаційні носії повинні зберігатися в сухому теплому місці, не допускати попадання прямих сонячних променів, температура повітря 18-40^{0}С, при вологості 50-70%, не допускати попадання пилу.

4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

Програмна документація на даний програмний продукт повинна включати наступні документи:

- специфікації програми;
- текст програми;
- опис програми;
- керівництво користувача. Керівництво експерта з моделювання та прогнозування.

5 ТЕХНІКО–ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ ПРОЕКТУ РОЗРОБКИ ПРОГРАМНОГО ПРОДУКТУ

Техніко–економічне обґрунтування (ТЕО) – це обов'язкова складова частина будь–якого інвестиційного проекту, що потребує певних фінансових витрат. Основjна мета розробки ТЕО – дати фінансову оцінку передбачуваних витрат та одержуваного корисного результату, оцінити прибутковість проекту і, в кінцевому підсумку, економічну доцільність його розробки та впровадження.

Аналіз техніко-економічних показників є обов’язковою умовою прийняття рішень щодо вкладення фінансових коштів у реконструкцію, будівництво та введення в експлуатацію нових технічних засобів будь-якого призначення.

Початковим етапом розрахунку величини трудових витрат розробників є оцінка розміру програмного забезпечення. Основні відмінності методик, що застосовуються в оцінці трудовитрат, полягають у використовуваному типі критерію оцінки якості (кількісний або якісний) [6].

Згідно моделі COCOMO, розмір проекту S вимірюється в рядках коду LOC (KLOC), а трудовитрати в людино-місяцях.

$$E = a \cdot S^b \cdot EAF, \tag{5.1}$$

де E – витрати праці на проект (в людино-місяцях);

S^b – розмір коду (в KLOC);

EAF – фактор уточнення витрат (effort adjustment factor).

Для простих систем, a = 2,4; b = 1,05

Розмір програмного коду програмного засобу – 1000 рядків:

$$E = 2{,}4 \cdot 1{,}00^{1{,}05} \cdot 1 = 2{,}4 \tag{5.2}$$

Отже, згідно моделі COCOMO, орієнтовні трудовитрати на проект складуть приблизно 2,4 людино-місяці.

Згідно із завданням дипломного проекту необхідно визначити собівартість і ціну розробленого проекту «Дослідження транспортних потоків засобами імітаційного моделювання».

Основними статтями витрат прийняті:

- основна заробітна плата;
- відрахування на соціальні потреби;
- накладні витрати;
- витрати на персональний комп'ютер і ліцензійні базові програмні засоби.

Основна заробітна плата (ОЗП) оцінює працю інженера–програміста зі створення програмного продукту і визначається виходячи з кількості розробників, часу виконання розробки (годин), а також заробітної плати в розрахунку на одну годину. Рекомендована кількість виконавців – 1 *чол*; тривалість розробки – 2,4 місяців. Розрахунок зарплати проводиться по формі табл 5.1. Середня заробітня плата взята с ресурсу work.ua на момент 01.10.2021.

Таблиця 5.1 – Фонд місячної заробітної плати

№ п/п	Посада виконавця	Оклад, грн/міс	Кількість		Сума зарплати, грн
			чол	місяців	
1	інженер-програміст	11500	1	2,4	34500

Описаний в проекті програмний продукт розроблений одним програмістом в період з 01.09.21 до 05.12.21, що складає 68 днів або 13,6 робочих тижнів. Витрати робочого часу приймемо 40 часів у тиждень. Погодинна ставка кваліфікованого інженера–програміста складає 63,42 *грн/год*. Таким чином, витрачено робочого часу:

$$t_{\text{розробки}} = N_{\text{чол}} \cdot N_{\text{тиж}} \cdot N_{\text{год}}, \quad (5.3)$$

де $N_{\text{чол}}$ – кількість виконавців, *чол*;

$N_{\text{тиж}}$ – тривалість розробки;

$N_{\text{год}}$ – витрати робочого часу, *год*;

$$t_{\text{розробки}} = 1 \cdot 13{,}6 \cdot 40 = 544 \text{ чол/год}$$

ОЗП визначається за формулою:

$$\text{ОЗП} = t_{\text{розробки}} \cdot N \cdot K_{\text{КВ}}, \quad (5.4)$$

де $t_{\text{розробки}}$ – витрати праці у *чол/год;*

N – погодинна ставка;

$K_{КВ}$ – коефіцієнт кваліфікації програміста, приймаємо 0.75.

$$\text{ОЗП} = 544 \cdot 63{,}42 \cdot 0{,}75 = 25\,875{,}36 \text{ грн.}$$

Відрахування на соціальні потреби встановлюються у відсотках від суми заробітної плати:

$$C_{\text{соц}} = \frac{\text{ОЗП} \cdot 22\%}{100\%}, \quad (5.5)$$

$$C_{\text{соц}} = \frac{25875{,}36 \cdot 22\%}{100\%} = 5\,692{,}57 \text{ грн.}$$

Отримані результати за (1.2) – (1.3) підсумовуються. Вони складають 31 567,94 *грн.* та визначають основні прямі витрати.

Накладні витрати враховують загально-господарчі витрати по забезпеченню проведення роботи: витрати на опалення, електроенергію, амортизація будівель, зарплату адміністративного персоналу та інше. Вони визначаються в процентах (30–40 %) від суми прямих витрат:

$$C_{\text{накл}} = \frac{(\text{ОЗП} + C_{\text{соц}}) \cdot 35\%}{100\%}, \quad (5.6)$$

$$C_{\text{накл}} = \frac{(25\,875{,}36 + 5\,692{,}57) \cdot 35\%}{100\%} = 11\,048{,}77 \text{ грн.}$$

Протягом усього терміну використання нової техніки підприємство щорічно витрачає певні кошти, пов'язані з її експлуатацією.

Експлуатаційні витрати на персональний комп'ютер визначаються протягом терміну розробки програмного засобу в залежності від вартості комп'ютеру. В експлуатаційні витрати входять:

– витрати на електроенергію;

– вартість витратних матеріалів;

– витрати на ремонт;

– заробітна плата ремонтника;

– додаткові витрати – прибирання приміщення, охорона, оренда, комунальні послуги;

– амортизаційні витрати на персональний комп'ютер і програмне забезпечення.

Витрати на електроенергію ($C_{\text{ел}}$) визначаються за формулою:

$$C_{\text{ел}} = \text{P} \cdot \text{B} \cdot \text{T}_{\text{розр}}, \quad (5.7)$$

де P – потужність комп'ютера та допоміжних споживачів електричної енергії, приймаємо 0,45 *кВт/год*;

B – вартість 1 *кВт/год* складає 1,68 *грн,* за даними minfin.com.ua від 01.10.2021*;*

$T_{\text{розр}}$ – час роботи з ЕВМ, прийнято рівним робочому часу.

$$\text{C}_{\text{ел}} = 0{,}45 \cdot 1{,}68 \cdot 544 = 411{,}26 \text{ грн.}$$

Витрати на витратні матеріали ($C_{\text{вм}}$) протягом всього терміну експлуатації приблизно 10 % від вартості комп'ютеру. Вартість комп'ютеру 35000 *грн*, це комп'ютер QUBE QB i5 10400F RTX 2060 6GB 81 [2]. Термін

експлуатації – 5 роки. Отже, можна визначити ці витрати за період створення програмного засобу:

$$C_{\text{вм}} = \text{В} \cdot \frac{N_{\text{д}}}{N_{\text{екс}} \cdot 365} \cdot \frac{10\%}{100\%}, \quad (5.8)$$

де $B_{\text{ком}}$ – вартість персонального комп'ютеру;

$N_{\text{Д}}$ – кількість днів розробки програмного продукту;

$N_{\text{експ}}$ – термін експлуатації персонального комп'ютеру.

$$C_{\text{вм}} = 35000 \cdot \frac{40}{5 \cdot 365} \cdot \frac{10}{100} = 76{,}71 \text{ грн}$$

Заробітна плата ремонтника ($C_{\text{рем}}$) визначена наступним чином: на ремонт 50 комп'ютерів потрібен один інженер–системотехнік. Його середньомісячна заробітна плата приймається 9000 *грн,* за даними сайту work.ua на момент 01.10.2021. Тоді в перерахунку на один комп'ютер його заробітна плата складає:

$$C_{\text{рем}} = \frac{C'_{\text{рем}}}{N_{\text{ком}}} \cdot T_{\text{міс}}, \quad (5.9)$$

де $C'_{\text{рем}}$ – середньомісячна заробітна плата;

$N_{\text{ком}}$ – кількість комп'ютерів на одного ремонтника.

$T_{міс}$ – час розробки програмного продукта, міс.

Заробітна плата ремонтника ($C_{\text{рем}}$) буде складати:

$$C_{\text{рем}} = \frac{9000}{50} \cdot 2 = 360 \text{ грн.}$$

За статистикою витрати на комплектуючи вироби ($C_{\text{ком}}$) для ремонту персонального комп'ютера складає 10 % від його вартості за термін його експлуатації, тобто рівні витратам на витратні матеріали.

$$C_{\text{ком}} = C_{\text{вм}} = 40 \text{ грн.} \quad (5.10)$$

Амортизаційні відрахування на персональний комп'ютер (АПК) визначені з положення, що амортизаційний період в даний час дорівнює терміну морального старіння обчислювальної техніки і складає 3 роки. Отже, за 3 роки амортизаційні відрахування на персональний комп'ютер дорівнюють вартості комп'ютера. За період проектування амортизаційні відрахування складуть:

$$\text{АПК} = \text{В}_{\text{ком}} \cdot \frac{N_{\text{д}}}{N_{\text{експ}} \cdot 365}, \quad (5.11)$$

$$\text{АПК} = 35000 \cdot \frac{40}{3 \cdot 365} = 1278{,}54 \text{ грн.}$$

Амортизаційні відрахування на програмне забезпечення (АПЗ) залежать від його циклу заміни. Якщо прийняти термін морального старіння таким же, як у персонального комп'ютера, то амортизаційні відрахування на програмне забезпечення за 3 роки дорівнюють його вартості. Для функціонування персонального комп'ютера використовувалася операційна система Windows 10 Professional, для написання експертної системи оболонка програми AnyLogic.

$$\text{АКП}_w = 1735 \cdot \frac{2}{5 \cdot 12} = 57{,}83$$

$$\text{АКП}_w = 110935 \cdot \frac{2}{12} = 18489$$

Розрахунок амортизаційних відрахувань на програмне забезпечення зведений в табл. 5.2.

Таблиця 5.2 – Використовуване програмне забезпечення

Найменування програмного забезпечення	Вартість програмного забезпечення, *грн*	Джерело придбання	Амортизаційні відрахування, *грн*
Windows 10 Professional	1735	http://mtsoft.kiev.ua/product/windows-10-professional	57,83
AnyLogic	110935	https://www.anylogic.ru/	18489
Всього:	112670	–	18546,8

Додаткові витрати ($C_{\text{дод}}$): прибирання приміщень, охорона, аренда, комунальні послуги важко оцінити точно і прийняти рівними 50 % заробітної плати інженера–системотехніка, тобто 7875 *грн*.

Оренду приміщень приймемо рівною 6000 гривень на місяць, за даними сайту dom.ria від 01.10.2021. Розмір приміщення 15 квадратних метрів. Ціна одного кв.м 400 грн.

Сумарні експлуатаційні витрати на один персональний комп'ютер складають:

$$C_{\text{експ}} = C_{\text{ел}} + C_{\text{вм}} + C_{\text{рем}} + C_{\text{ком}} + \text{АПК} + \text{АПО} + C_{\text{ор}} + C_{\text{дод}}, \qquad (5.12)$$

$$C_{\text{експ}} = 411{,}26 + 76{,}71 + 360 + 40 + 1278{,}54 + 18546{,}8 + 12000 + 7875 = 40\,588{,}31 \text{ грн.}$$

Результати розрахунків зводимо у табл. 5.3.

Таблиця 5.3 – Експлуатаційні витрати на ПК і ПО

Найменування витрат	Витрати, *грн*
Витрати на електроенергію	411,26
Вартість витратних матеріалів	76,71
Витрати на ремонт	400
Амортизація персонального комп’ютера	1278,54
Амортизація програмного забезпечення	18546,8
Оренда приміщення	12000
Додаткові витрати	7878
Всього	40 588,31

Таким чином, витрати на створення програмного продукту складають:

$$C_{\text{розробки}} = \text{ОЗП} + C_{\text{соц}} + C_{\text{накл}} + C_{\text{експ}}, \quad (5.13)$$

$$\text{C}_{\text{розробки}} = 25\,875{,}36 + 5\,692{,}57 + 11\,048{,}77 + 40\,588{,}31 = 83\,205{,}01 \text{ грн.}$$

Розрахунок витрат зводимо у табл. 5.4.

Таблиця 5.4 – Кошторис витрат на розробку програмного засобу

Найменування витрат	Витрати, *грн*
Основна заробітна плата	25 875,36
Відрахування на соціальні потреби	5 692,57
Накладні витрати	11 048,77
Експлуатаційні витрати	40 588,31
Всього	83 205,01

Висновок: таким чином розрахунок показав, що собівартість розробленого проекту «Дослідження транспортних потоків засобами імітаційного моделювання» складає 83 205,01 грн., при цьому найбільша питома вага складають експлуатаційні витрати.

6. СТАДІЇ ТА ЕТАПИ РОЗРОБКИ

Стадії та етапи розробки програмного продукту «Дослідження часових рядів навантаженості мережевих систем» представлені в табл 6.1.

Таблиця 6.1 – Стадії та етапи розробки

№ пор.	Назва розділів дипломної роботи	Термін виконання розділів роботи	При-мітка
1	Вступ	1.09.21	
2	Аналіз сучасного стану дослідження проблеми за науковими літературними джерелами	2.09.21 – 15.09.21	від 70 джерел
2.1	Дослідження існуючих рішень та інших аспектів дослідження	4.09.21 – 10.09.21	
3	Аналіз сучасного стану програмно-апаратного забезпечення, яке потребує вдосконалення для вирішення проблем дослідження	16.09.21 – 10.10.21	
4	Постановка задачі, технічне завдання	11.10.21 – 17.10.21	30%
4.1	Розробка постановки задачі	11.10.21 – 13.10.21	
4.2	Розробка технічного завдання	14.10.21 – 16.10.21	
4.3	Затвердження постановки задачі та технічного завдання	17.10.21	
5	Техніко-економічні показники	18.10.21 – 19.10.21	
5.1	Розробка техніко-економічного обґрунтування розробки проекту	18.10.21	
5.2	Затвердження техніко-економічного обґрунтування розробки проекту	19.10.21	
6	Розробка інструментальних засобів дослідження	20.10.21 – 7.11.21	
6.1	Проектування інструментального засобу	20.10.21 – 25.10.21	
6.2	Розробка алгоритмів та програмування інструментального засобу	26.10.21 – 3.11.21	

Продовження таблиці 6.1

6.3	Розробка та програмування інтерфейсу користувача	4.11.21 – 6.11.21	
6.4	Тестування та відлагодження інструментального засобу	7.11.21	
7	Виконання досліджень	8.11.21 – 14.11.21	60%
8	Оформлення тез доповідей	15.11.21 – 18.11.21	
9	Оформлення статті у фаховий журнал	19.11.21 – 22.11.21	
10	Оформлення пояснювальної записки	23.11.21 – 28.11.21	
11	Розробка демонстраційних матеріалів	29.11.21 – 5.12.21	100%

7. ПОРЯДОК КОНТРОЛЮ ТА ПРИЙОМУ

Контроль здійснюється за допомогою виконання набору тестів з метою виявлення помилок в програмному продукті та його специфікаціях. Контроль за виконанням роботи здійснює головним керівником з розробки. Прийом здійснюється державною комісією.

БІБЛІОГРАФІЧНИЙ СПИСОК

1. Івченко, Ю.М. Основи стандартизації програмних систем [Текст]: методичні вказівки до дипломного проектування та лабораторних робіт / уклад.: Ю. М. Івченко, В. І. Шинкаренко, В. Г. Івченко; Дніпропетр. нац. ун-т залізн. трансп. ім. акад. В. Лазаряна. – Д.: Вид-во Дніпропетр. нац. ун-ту залізн. трансп. ім. акад. В. Лазаряна, 2009. - 38 с.

2. Магазин техніки COMFY. URL: https://comfy.ua/ (дата звернення 13.12.2021).

ДОДАТОК Б

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ЗАТВЕРДЖУЮ
Проректор Дніпровського
національного університету
залізничного транспорту
імені академіка В. Лазаряна
Борис БОДНАР

Програмний продукт «Імітаційна модель транспортних потоків»

Робочий проект
ЛИСТ ЗАТВЕРДЖЕННЯ
1116130.01206 ЛЗ

Завідувача кафедри КІТ
Вадим ГОРЯЧКІН

Керівник розробки
Олександра ГОРБОВА

Виконавець
Олексій МЕРЗЛИЙ

Нормоконтролер
Олена КУРОП'ЯТНИК

2021

ЗАТВЕРДЖЕНО

1116130.01206-01-ЛЗ

ДОСЛІДЖЕННЯ ТРАНСПОРТНИХ ПОТОКІВ ЗАСОБАМИ ІМІТАЙІЙНОГО МОДЕЛЮВАННЯ

Специфікація

1116130.01206-01

Аркушів 2

2021

Позначення	Найменування	Примітка
1116130.01206-01-ЛЗ	Лист затвердження	
1116130.01206-01 12 01-ЛЗ	Лист затвердження	
1116130.01206-01 12 01	Текст програми	
1116130.01206-01 13 01-ЛЗ	Лист затвердження	
1116130.01206-01 13 01	Опис програми	
1116130.01206-01 І3 01-ЛЗ	Лист затвердження	
1116130.01206-01 І3 01	Керівництво користувача Керівництво експерта з моделювання та прогнозування	

ЗАТВЕРДЖЕНО

1116130.01206-01 12 01-ЛЗ

ДОСЛІДЖЕННЯ ТРАНСПОРТНИХ ПОТОКІВ ЗАСОБАМИ ІМІТАЙІЙНОГО МОДЕЛЮВАННЯ

Текст програми

1116130.01206-01 12 01-ЛЗ

Листів 16

2021

АНОТАЦІЯ

Документ 1116130.01206-01 12 01-ЛЗ «Дослідження транспортних потоків засобами імітаційного моделювання» є опис програмних модулів і коду продукту «Імітаційна модель транспортних потоків», та входить до складу документації на проект.

Розробка імітаційної моделі транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання та поліпшення такої системи в реальному житті. В документі містяться основні вимоги до розробки програмного додатку та його функціональні можливості. Об’єм оперативної пам’яті, що займає програмний комплекс складає 400 мб. Конфігурація комп’ютера стандартна. Програма функціонує в середовищі Windows 10.

ЗМІСТ

СХЕМА ВЗАЕМОДІЇ МОДУЛІВ ПРОГРАМИ

Схема взаєсодії модулів програми наведено на рис 1.1.

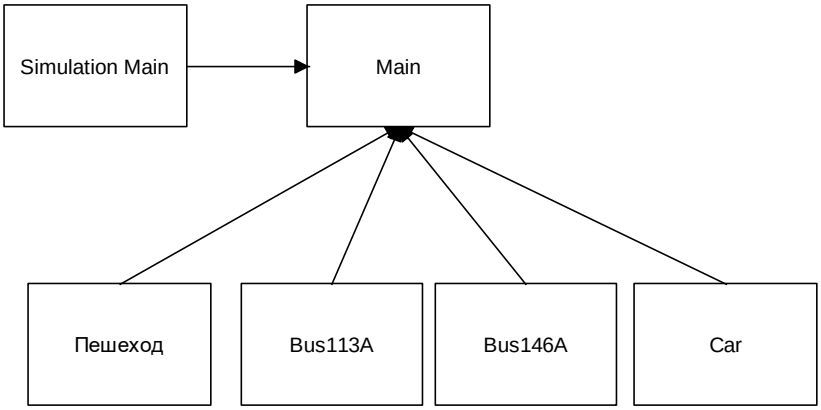


Рисунок 1.1 – Схема взаємодії модулів програми

ТЕКСТ ПРОГРАМИ

1 Модуль Car

```
package diplom_final;
import java.io.Serializable;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.ArrayDeque;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.Currency;
import java.util.Date;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Hashtable;
import java.util.Iterator;
import java.util.LinkedHashMap;
import java.util.LinkedHashSet;
import java.util.LinkedList;
import java.util.List;
import java.util.ListIterator;
import java.util.Locale;
import java.util.Map;
import java.util.PriorityQueue;
import java.util.Random;
import java.util.Set;
import java.util.SortedMap;
import java.util.SortedSet;
import java.util.Stack;
import java.util.Timer;
import java.util.TreeMap;
import java.util.TreeSet;
import java.util.Vector;
import java.awt.Color;
import java.awt.Font;
import com.anylogic.engine.connectivity.ResultSet;
import com.anylogic.engine.connectivity.Statement;
import com.anylogic.engine.elements.*;
import com.anylogic.engine.markup.Network;
import com.anylogic.engine.Position;
import com.anylogic.engine.markup.PedFlowStatistics;
import com.anylogic.engine.markup.DensityMap;


import static java.lang.Math.*;
import static com.anylogic.engine.UtilitiesArray.*;
import static com.anylogic.engine.UtilitiesCollection.*;
import static com.anylogic.engine.presentation.UtilitiesColor.*;
import static com.anylogic.engine.HyperArray.*;

import com.anylogic.engine.*;
import com.anylogic.engine.analysis.*;
import com.anylogic.engine.connectivity.*;
import com.anylogic.engine.database.*;
import com.anylogic.engine.gis.*;
import com.anylogic.engine.markup.*;
import com.anylogic.engine.presentation.*;
import com.anylogic.engine.gui.*;

import com.anylogic.libraries.pedestrian.*;
import com.anylogic.libraries.road.*;
import com.anylogic.libraries.processmodeling.*;

import java.awt.geom.Arc2D;

public class Car extends diplom_final.Vehicle
{
 // Параметры

 public
double  ВремяПоявления;

 /**
  * Возвращает значение по умолчанию параметра
<code>ВремяПоявления</code>.
  * <i>Пользователь не должен вызывать этот метод</i>
  */
 @AnyLogicInternalCodegenAPI
 public double _ВремяПоявления_DefaultValue_xjal() {
  final Car self = this;
  return
time()
;
 }

 public void set_ВремяПоявления( double ВремяПоявления ) {
  if (ВремяПоявления == this.ВремяПоявления) {
   return;
  }
  double _oldValue_xjal = this.ВремяПоявления;
  this.ВремяПоявления = ВремяПоявления;
  onChange_ВремяПоявления_xjal( _oldValue_xjal );
  onChange();
 }

 /**
  * Calls "On change" action for parameter ВремяПоявления.<br>
  * Note that 'oldValue' in that action will be unavailable if this
method is called by user
  * (current parameter value will be passed as 'oldValue').<br>
  * Please call <code>set_ВремяПоявления()</code> method
instead.
  */
 protected void onChange_ВремяПоявления() {
  onChange_ВремяПоявления_xjal( ВремяПоявления );
 }

 @AnyLogicInternalCodegenAPI
 protected void onChange_ВремяПоявления_xjal( double oldValue
) {
 }


 /**
  * Возвращает значение по умолчанию параметра
<code>animationType</code>.
  * <i>Пользователь не должен вызывать этот метод</i>
  */
 @AnyLogicInternalCodegenAPI
 public int _animationType_DefaultValue_xjal() {
  final Car self = this;
  return
uniform_discr( 1, 5 )
;
 }


 @Override
 public void setParametersToDefaultValues() {
  super.setParametersToDefaultValues();
  ВремяПоявления = _ВремяПоявления_DefaultValue_xjal();
```

```
  }

  @Override
  public boolean setParameter(String _name_xjal, Object _value_xjal,
boolean _callOnChange_xjal) {
    switch ( _name_xjal ) {
    case "ВремяПоявления":
      if ( _callOnChange_xjal ) {
        set_ВремяПоявления( ((Number) _value_xjal).doubleValue()
);
      } else {
        ВремяПоявления = ((Number) _value_xjal).doubleValue();
      }
      return true;
    default:
      return super.setParameter( _name_xjal, _value_xjal,
_callOnChange_xjal );
    }
  }

  @Override
  public <T> T getParameter(String _name_xjal) {
    Object _result_xjal;
    switch ( _name_xjal ) {
    case "ВремяПоявления": _result_xjal = ВремяПоявления; break;
    default: _result_xjal = super.getParameter( _name_xjal ); break;
    }
    return (T) _result_xjal;
  }

  @AnyLogicInternalCodegenAPI
  private static String[] _parameterNames_xjal;

  @Override
  public String[] getParameterNames() {
    String[] result = _parameterNames_xjal;
    if (result == null) {
      List<String> list = new ArrayList<>( Arrays.asList(
super.getParameterNames() ) );
      list.add( "ВремяПоявления" );
      result = list.toArray( new String[ list.size() ] );
      _parameterNames_xjal = result;
    }
    return result;
  }
  @AnyLogicInternalCodegenAPI
  private static Map<String, IElementDescriptor>
elementDesciptors_xjal = createElementDescriptors( Car.class );

  @AnyLogicInternalCodegenAPI
  @Override
  public Map<String, IElementDescriptor> getElementDesciptors() {
    return elementDesciptors_xjal;
  }
  @AnyLogicCustomProposalPriority(type =
AnyLogicCustomProposalPriority.Type.STATIC_ELEMENT)
  public static final Scale scale = new Scale( 10.0 );

  @Override
  public Scale getScale() {
    return scale;
  }


  // Области
  @AnyLogicInternalCodegenAPI
  protected static final Pair<String, Color>[]
_car1_customColors_xjal = new Pair[] {
    new Pair<String, Color>( "Material__1__Surf", null ),
    new Pair<String, Color>( "Material__2__Surf", null ),
    new Pair<String, Color>( "Material__4__Surf", white ),
    new Pair<String, Color>( "Material__3__Surf", null ),
  };
  @AnyLogicInternalCodegenAPI
  protected static final Pair<String, Color>[]
_car2_customColors_xjal = new Pair[] {
    new Pair<String, Color>( "Material__1__Surf", null ),
    new Pair<String, Color>( "Material__2__Surf", null ),
    new Pair<String, Color>( "Material__4__Surf", dodgerBlue ),
    new Pair<String, Color>( "Material__3__Surf", null ),
  };
  @AnyLogicInternalCodegenAPI
  protected static final Pair<String, Color>[]
_car3_customColors_xjal = new Pair[] {
    new Pair<String, Color>( "Material__1__Surf", null ),
    new Pair<String, Color>( "Material__2__Surf", null ),
    new Pair<String, Color>( "Material__4__Surf", black ),
    new Pair<String, Color>( "Material__3__Surf", null ),
  };
  @AnyLogicInternalCodegenAPI
  protected static final Pair<String, Color>[]
_car4_customColors_xjal = new Pair[] {
    new Pair<String, Color>( "Material__1__Surf", null ),
    new Pair<String, Color>( "Material__2__Surf", null ),
    new Pair<String, Color>( "Material__4__Surf", maroon ),
    new Pair<String, Color>( "Material__3__Surf", null ),
  };
  @AnyLogicInternalCodegenAPI
  protected static final int _car =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 1;
  @AnyLogicInternalCodegenAPI
  protected static final int _car1 =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 2;
  @AnyLogicInternalCodegenAPI
  protected static final int _car2 =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 3;
  @AnyLogicInternalCodegenAPI
  protected static final int _car3 =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 4;
  @AnyLogicInternalCodegenAPI
  protected static final int _car4 =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 5;

  /** Internal constant, shouldn't be accessed by user */
  @AnyLogicInternalCodegenAPI
  protected static final int _SHAPE_NEXT_ID_xjal =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 6;

  @AnyLogicInternalCodegenAPI
  public boolean isPublicPresentationDefined() {
    return super.isPublicPresentationDefined() || true;
  }

  @AnyLogicInternalCodegenAPI
  public boolean isEmbeddedAgentPresentationVisible( Agent _a ) {
    return super.isEmbeddedAgentPresentationVisible( _a );
  }


  /**
   * <i>Пользователь не должен вызывать этот метод</i>
   */
  @AnyLogicInternalCodegenAPI
  private void _car_SetDynamicParams_xjal( Shape3DObject shape )
{
    boolean _visible =
animationType == 1
;
    shape.setVisible( _visible );
              if ( _visible ) {
              }
  }

  protected Shape3DObject car;

  /**
   * <i>Пользователь не должен вызывать этот метод</i>
   */
  @AnyLogicInternalCodegenAPI
  private void _car1_SetDynamicParams_xjal( Shape3DObject shape
) {
    boolean _visible =
```

```
animationType == 2
;
    shape.setVisible( _visible );
                if ( _visible ) {
                }
  }

  protected Shape3DObject car1;

  /**
   * <i>Пользователь не должен вызывать этот метод</i>
   */
  @AnyLogicInternalCodegenAPI
  private void _car2_SetDynamicParams_xjal( Shape3DObject shape
) {
    boolean _visible =
animationType == 3
;
    shape.setVisible( _visible );
                if ( _visible ) {
                }
  }

  protected Shape3DObject car2;

  /**
   * <i>Пользователь не должен вызывать этот метод</i>
   */
  @AnyLogicInternalCodegenAPI
  private void _car3_SetDynamicParams_xjal( Shape3DObject shape
) {
    boolean _visible =
animationType == 4
;
    shape.setVisible( _visible );
                if ( _visible ) {
                }
  }

  protected Shape3DObject car3;

  /**
   * <i>Пользователь не должен вызывать этот метод</i>
   */
  @AnyLogicInternalCodegenAPI
  private void _car4_SetDynamicParams_xjal( Shape3DObject shape
) {
    boolean _visible =
animationType == 5
;
    shape.setVisible( _visible );
                if ( _visible ) {
                }
  }

  protected Shape3DObject car4;
  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsBP0_xjal() {
    car = new Shape3DObject(
                        Car.this, SHAPE_DRAW_2D3D, true, 0.0,
0.0, 0.0, 0.0,
                                    1.0, false, "/diplom_final/",
                                    "3d/car.dae",
OBJECT_3D_YZX_AXIS_ORDER, true, -25.0, -10.0,
                                    49.0, 20.0, null ) {
      @Override
      public void updateDynamicProperties(boolean publicOnly) {
        _car_SetDynamicParams_xjal( this );
        super.updateDynamicProperties(publicOnly);
      }
    };

    car1 = new Shape3DObject(
                        Car.this, SHAPE_DRAW_2D3D, true, 0.0,
0.0, 0.0, 0.0,
                                    1.0, false, "/diplom_final/",
                                    "3d/car.dae",
OBJECT_3D_YZX_AXIS_ORDER, true, -25.0, -10.0,
                                    49.0, 20.0, 1496561912180L,
_car1_customColors_xjal ) {
      @Override
      public void updateDynamicProperties(boolean publicOnly) {
        _car1_SetDynamicParams_xjal( this );
        super.updateDynamicProperties(publicOnly);
      }
    };

    car2 = new Shape3DObject(
                        Car.this, SHAPE_DRAW_2D3D, true, 0.0,
0.0, 0.0, 0.0,
                                    1.0, false, "/diplom_final/",
                                    "3d/car.dae",
OBJECT_3D_YZX_AXIS_ORDER, true, -25.0, -10.0,
                                    49.0, 20.0, 1496561912274L,
_car2_customColors_xjal ) {
      @Override
      public void updateDynamicProperties(boolean publicOnly) {
        _car2_SetDynamicParams_xjal( this );
        super.updateDynamicProperties(publicOnly);
      }
    };

    car3 = new Shape3DObject(
                        Car.this, SHAPE_DRAW_2D3D, true, 0.0,
0.0, 0.0, 0.0,
                                    1.0, false, "/diplom_final/",
                                    "3d/car.dae",
OBJECT_3D_YZX_AXIS_ORDER, true, -25.0, -10.0,
                                    49.0, 20.0, 1496561912352L,
_car3_customColors_xjal ) {
      @Override
      public void updateDynamicProperties(boolean publicOnly) {
        _car3_SetDynamicParams_xjal( this );
        super.updateDynamicProperties(publicOnly);
      }
    };

    car4 = new Shape3DObject(
                        Car.this, SHAPE_DRAW_2D3D, true, 0.0,
0.0, 0.0, 0.0,
                                    1.0, false, "/diplom_final/",
                                    "3d/car.dae",
OBJECT_3D_YZX_AXIS_ORDER, true, -25.0, -10.0,
                                    49.0, 20.0, 1496561912414L,
_car4_customColors_xjal ) {
      @Override
      public void updateDynamicProperties(boolean publicOnly) {
        _car4_SetDynamicParams_xjal( this );
        super.updateDynamicProperties(publicOnly);
      }
    };

  }

  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsAP0_xjal() {
  }

  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsBS0_xjal() {
  }


  // Статическая инициализация элементов, у которых разрешено
программное управление
  {
    _createPersistentElementsBP0_xjal();
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public ShapeTopLevelPresentationGroup getPresentationShape() {
```

```
    return presentation;
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public ShapeModelElementsGroup getModelElementsShape() {
    return icon;
  }

  /**
   * Конструктор
   */
  public Car( Engine engine, Agent owner, AgentList<? extends Car>
ownerPopulation ) {
    super( engine, owner, ownerPopulation );
    instantiateBaseStructureThis_xjal();
  }

  @AnyLogicInternalCodegenAPI
  public void onOwnerChanged_xjal() {
    super.onOwnerChanged_xjal();
    setupReferences_xjal();
  }

  @AnyLogicInternalCodegenAPI
  public void instantiateBaseStructure_xjal() {
    super.instantiateBaseStructure_xjal();
    instantiateBaseStructureThis_xjal();
  }

  @AnyLogicInternalCodegenAPI
  private void instantiateBaseStructureThis_xjal() {
    setupReferences_xjal();
  }

  @AnyLogicInternalCodegenAPI
  private void setupReferences_xjal() {
    main = get_Main();
  }

  /**
   * Simple constructor. Please add created agent to some population
by calling goToPopulation() function
   */
  public Car() {
  }

  /**
   * Simple constructor. Please add created agent to some population
by calling goToPopulation() function
   */
  public Car( int animationType, double ВремяПоявления ) {
    super( animationType );
    this.ВремяПоявления = ВремяПоявления;
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public void doCreate() {
    super.doCreate();
    // Присвоение начальных значений простым переменным
    setupPlainVariables_Car_xjal();
    // Динамическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsAP0_xjal();
    presentation.initialize_xjal( false, false , car, car1, car2, car3, car4
);
    icon.initialize_xjal(
this.<ModelElementDescriptor[]>getElementProperty(
"diplom_final.Car.icon",
IElementDescriptor.MODEL_ELEMENT_DESCRIPTORS ), false,
true  );
    icon.setIconOffsets( 0.0, 0.0 );
                // Соединители с нереплицированными объектами
    // Создание реплицированных вложенных объектов
    setupInitialConditions_xjal( Car.class );
    // Динамическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsBS0_xjal();
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public void doStart() {
    super.doStart();
  }

  @AnyLogicInternalCodegenAPI
  public void onStartup() {
    super.onStartup();

main.ВсегоАвто++;
main.СтатистикаАвто.add(main.Авто.size());
  }

  /**
   * Присвоение начальных значений простым переменным<br>
   * <em>This method isn't designed to be called by user and may be
removed in future releases.</em>
   */
  @AnyLogicInternalCodegenAPI
  public void setupPlainVariables_xjal() {
    super.setupPlainVariables_xjal();
    setupPlainVariables_Car_xjal();
  }

  /**
   * Присвоение начальных значений простым переменным<br>
   * <em>This method isn't designed to be called by user and may be
removed in future releases.</em>
   */
  @AnyLogicInternalCodegenAPI
  private void setupPlainVariables_Car_xjal() {
  }

  // API пользователя ------------------------------------------------------
  public Main get_Main() {
    {
      Agent owner = getOwner();
      if ( owner instanceof Main ) return (Main) owner;
    }
    return null;
  }

  /**
   * Переменная только для чтения. <em>Значение переменной
не должно изменяться пользователем.</em>
   */

@AnyLogicCustomSerialization(AnyLogicCustomSerializationMod
e.REFERENCE)
  public transient diplom_final.Main main;

  public AgentList<? extends Car> getPopulation() {
    return (AgentList<? extends Car>) super.getPopulation();
  }

  public List<? extends Car> agentsInRange( double distance ) {
    return (List<? extends Car>) super.agentsInRange( distance );
  }

  @AnyLogicInternalCodegenAPI
  public void onDestroy() {
    // Действие при уничтожении

main.ВремяПроезда.add( time()- ВремяПоявления );
    super.onDestroy();
  }



}
```

2 Модуль Bus 113A

```
package diplom_final;
import java.io.Serializable;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.ArrayDeque;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.Currency;
import java.util.Date;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Hashtable;
import java.util.Iterator;
import java.util.LinkedHashMap;
import java.util.LinkedHashSet;
import java.util.LinkedList;
import java.util.List;
import java.util.ListIterator;
import java.util.Locale;
import java.util.Map;
import java.util.PriorityQueue;
import java.util.Random;
import java.util.Set;
import java.util.SortedMap;
import java.util.SortedSet;
import java.util.Stack;
import java.util.Timer;
import java.util.TreeMap;
import java.util.TreeSet;
import java.util.Vector;
import java.awt.Color;
import java.awt.Font;
import com.anylogic.engine.connectivity.ResultSet;
import com.anylogic.engine.connectivity.Statement;
import com.anylogic.engine.elements.*;
import com.anylogic.engine.markup.Network;
import com.anylogic.engine.Position;
import com.anylogic.engine.markup.PedFlowStatistics;
import com.anylogic.engine.markup.DensityMap;


import static java.lang.Math.*;
import static com.anylogic.engine.UtilitiesArray.*;
import static com.anylogic.engine.UtilitiesCollection.*;
import static com.anylogic.engine.presentation.UtilitiesColor.*;
import static com.anylogic.engine.HyperArray.*;

import com.anylogic.engine.*;
import com.anylogic.engine.analysis.*;
import com.anylogic.engine.connectivity.*;
import com.anylogic.engine.database.*;
import com.anylogic.engine.gis.*;
import com.anylogic.engine.markup.*;
import com.anylogic.engine.presentation.*;
import com.anylogic.engine.gui.*;

import com.anylogic.libraries.pedestrian.*;
import com.anylogic.libraries.road.*;
import com.anylogic.libraries.processmodeling.*;

import java.awt.geom.Arc2D;

public class Bus113A extends diplom_final.Vehicle
       implements ICar
{
  // Параметры
  public
double  ВремяПоявления113;

  /**
   * Возвращает значение по умолчанию параметра
<code>ВремяПоявления113</code>.
   * <i>Пользователь не должен вызывать этот метод</i>
   */
  @AnyLogicInternalCodegenAPI
  public double _ВремяПоявления113_DefaultValue_xjal() {
    final Bus113A self = this;
    return
time()
;
  }

  public void set_ВремяПоявления113( double
ВремяПоявления113 ) {
    if (ВремяПоявления113 == this.ВремяПоявления113) {
      return;
    }
    double _oldValue_xjal = this.ВремяПоявления113;
    this.ВремяПоявления113 = ВремяПоявления113;
    onChange_ВремяПоявления113_xjal( _oldValue_xjal );
    onChange();
  }

  /**
   * Calls "On change" action for parameter
ВремяПоявления113.<br>
   * Note that 'oldValue' in that action will be unavailable if this
method is called by user
   * (current parameter value will be passed as 'oldValue').<br>
   * Please call <code>set_ВремяПоявления113()</code> method
instead.
   */
  protected void onChange_ВремяПоявления113() {
    onChange_ВремяПоявления113_xjal( ВремяПоявления113 );
  }

  @AnyLogicInternalCodegenAPI
  protected void onChange_ВремяПоявления113_xjal( double
oldValue ) {
  }



  @Override
  public void setParametersToDefaultValues() {
    super.setParametersToDefaultValues();
    ВремяПоявления113 =
_ВремяПоявления113_DefaultValue_xjal();
  }

  @Override
  public boolean setParameter(String _name_xjal, Object _value_xjal,
boolean _callOnChange_xjal) {
    switch ( _name_xjal ) {
    case "ВремяПоявления113":
      if ( _callOnChange_xjal ) {
        set_ВремяПоявления113( ((Number)
_value_xjal).doubleValue() );
      } else {
        ВремяПоявления113 = ((Number) _value_xjal).doubleValue();
      }
      return true;
    default:
      return super.setParameter( _name_xjal, _value_xjal,
_callOnChange_xjal );
    }
  }
```

```
  @Override
  public <T> T getParameter(String _name_xjal) {
    Object _result_xjal;
    switch ( _name_xjal ) {
    case "ВремяПоявления113": _result_xjal =
ВремяПоявления113; break;
    default: _result_xjal = super.getParameter( _name_xjal ); break;
    }
    return (T) _result_xjal;
  }

  @AnyLogicInternalCodegenAPI
  private static String[] _parameterNames_xjal;

  @Override
  public String[] getParameterNames() {
    String[] result = _parameterNames_xjal;
    if (result == null) {
      List<String> list = new ArrayList<>( Arrays.asList(
super.getParameterNames() ) );
      list.add( "ВремяПоявления113" );
      result = list.toArray( new String[ list.size() ] );
      _parameterNames_xjal = result;
    }
    return result;
  }
  @AnyLogicInternalCodegenAPI
  private static Map<String, IElementDescriptor>
elementDesciptors_xjal = createElementDescriptors( Bus113A.class
);

  @AnyLogicInternalCodegenAPI
  @Override
  public Map<String, IElementDescriptor> getElementDesciptors() {
    return elementDesciptors_xjal;
  }
  @AnyLogicCustomProposalPriority(type =
AnyLogicCustomProposalPriority.Type.STATIC_ELEMENT)
  public static final Scale scale = new Scale( 10.0 );

  @Override
  public Scale getScale() {
    return scale;
  }




  // Области
  @AnyLogicInternalCodegenAPI
  protected static final int _bus_1 =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 1;

  /** Internal constant, shouldn't be accessed by user */
  @AnyLogicInternalCodegenAPI
  protected static final int _SHAPE_NEXT_ID_xjal =
diplom_final.Vehicle._SHAPE_NEXT_ID_xjal + 2;

  @AnyLogicInternalCodegenAPI
  public boolean isPublicPresentationDefined() {
    return super.isPublicPresentationDefined() || true;
  }

  @AnyLogicInternalCodegenAPI
  public boolean isEmbeddedAgentPresentationVisible( Agent _a ) {
    return super.isEmbeddedAgentPresentationVisible( _a );
  }

  protected Shape3DObject bus_1;
  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsBP0_xjal() {
    bus_1 = new Shape3DObject(
                        Bus113A.this, SHAPE_DRAW_2D3D, true,
0.0, 0.0, 0.0, 0.0,
                                    1.0, true, "/diplom_final/",
                                    "3d/bus_1.dae",
OBJECT_3D_XYZ_AXIS_ORDER, true, -39.0, -10.0,
                                    79.0, 20.0, null );


  }

  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsAP0_xjal() {
  }

  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsBS0_xjal() {
  }


  // Статическая инициализация элементов, у которых разрешено
программное управление
  {
    _createPersistentElementsBP0_xjal();
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public ShapeTopLevelPresentationGroup getPresentationShape() {
    return presentation;
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public ShapeModelElementsGroup getModelElementsShape() {
    return icon;
  }

  /**
   * Конструктор
   */
  public Bus113A( Engine engine, Agent owner, AgentList<? extends
Bus113A> ownerPopulation ) {
    super( engine, owner, ownerPopulation );
    instantiateBaseStructureThis_xjal();
  }

  @AnyLogicInternalCodegenAPI
  public void onOwnerChanged_xjal() {
    super.onOwnerChanged_xjal();
    setupReferences_xjal();
  }

  @AnyLogicInternalCodegenAPI
  public void instantiateBaseStructure_xjal() {
    super.instantiateBaseStructure_xjal();
    instantiateBaseStructureThis_xjal();
  }

  @AnyLogicInternalCodegenAPI
  private void instantiateBaseStructureThis_xjal() {
    setupReferences_xjal();
  }

  @AnyLogicInternalCodegenAPI
  private void setupReferences_xjal() {
    main = get_Main();
  }

  /**
   * Simple constructor. Please add created agent to some population
by calling goToPopulation() function
   */
  public Bus113A() {
  }

  /**
   * Simple constructor. Please add created agent to some population
by calling goToPopulation() function
   */
```

```
  public Bus113A( int animationType, double ВремяПоявления113 )
{
    super( animationType );
    this.ВремяПоявления113 = ВремяПоявления113;
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public void doCreate() {
    super.doCreate();
    // Присвоение начальных значений простым переменным
    setupPlainVariables_Bus113A_xjal();
    // Динамическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsAP0_xjal();
    presentation.initialize_xjal( false, false , bus_1 );
    icon.initialize_xjal(
this.<ModelElementDescriptor[]>getElementProperty(
"diplom_final.Bus113A.icon",
IElementDescriptor.MODEL_ELEMENT_DESCRIPTORS ), false,
true  );
    icon.setIconOffsets( 0.0, 0.0 );
                // Соединители с нереплицированными объектами
    // Создание реплицированных вложенных объектов
    setupInitialConditions_xjal( Bus113A.class );
    // Динамическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsBS0_xjal();
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public void doStart() {
    super.doStart();
  }

  @AnyLogicInternalCodegenAPI
  public void onStartup() {
    super.onStartup();

main.ВсегоАвто++;
  }

  /**
   * Присвоение начальных значений простым переменным<br>
   * <em>This method isn't designed to be called by user and may be
removed in future releases.</em>
   */
  @AnyLogicInternalCodegenAPI
  public void setupPlainVariables_xjal() {
    super.setupPlainVariables_xjal();
    setupPlainVariables_Bus113A_xjal();
  }

  /**
   * Присвоение начальных значений простым переменным<br>
   * <em>This method isn't designed to be called by user and may be
removed in future releases.</em>
   */
  @AnyLogicInternalCodegenAPI
  private void setupPlainVariables_Bus113A_xjal() {
  }

  // API пользователя ----------------------------------------------------
  public Main get_Main() {
    {
      Agent owner = getOwner();
      if ( owner instanceof Main ) return (Main) owner;
    }
    return null;
  }

  /**
   * Переменная только для чтения. <em>Значение переменной
не должно изменяться пользователем.</em>
   */

@AnyLogicCustomSerialization(AnyLogicCustomSerializationMod
e.REFERENCE)
  public transient diplom_final.Main main;

  public AgentList<? extends Bus113A> getPopulation() {
    return (AgentList<? extends Bus113A>) super.getPopulation();
  }

  public List<? extends Bus113A> agentsInRange( double distance )
{
    return (List<? extends Bus113A>) super.agentsInRange( distance
);
  }

  @AnyLogicInternalCodegenAPI
  public void onDestroy() {
    // Действие при уничтожении

main.ВремяПроезда113.add( time()- ВремяПоявления113 );
    super.onDestroy();
  }



  public RoadNetwork getRoadNetwork() {
    return
ext(com.anylogic.libraries.road.CarExtension.class).getRoadNetwork
();
  }

  public com.anylogic.libraries.road.RoadNetworkDescriptor
getRoadNetworkDescriptor() {
            return
ext(com.anylogic.libraries.road.CarExtension.class).getRoadNetwork
Descriptor();
  }

  public double getLength() {
            return
ext(com.anylogic.libraries.road.CarExtension.class).getLength();
  }

  public double getLength( LengthUnits units ) {
            return
ext(com.anylogic.libraries.road.CarExtension.class).getLength(units)
;
  }

  public double getWidth() {
            return
ext(com.anylogic.libraries.road.CarExtension.class).getWidth();
  }

  public double getWidth( LengthUnits units ) {
            return
ext(com.anylogic.libraries.road.CarExtension.class).getWidth(units);
  }

  public void setLength( double length ) {

            ext(com.anylogic.libraries.road.CarExtension.class).setLe
ngth(length);
  }

  public void setLength( double length, LengthUnits units ) {

            ext(com.anylogic.libraries.road.CarExtension.class).setLe
ngth(length, units);
  }

  public void highlight( boolean yes ) {

            ext(com.anylogic.libraries.road.CarExtension.class).highli
ght(yes);
  }
```

```
  public boolean isHighlighted() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).isHighlighted();
  }

  @AnyLogicCustomProposalType(
AnyLogicCustomProposalType.Label.MPS )
  public double getSpeed() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getSpeed();
  }

  public double getSpeed( SpeedUnits units ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getSpeed(units);
  }

  public double getPreferredSpeed() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getPreferredSpeed();
  }

  public double getPreferredSpeed( SpeedUnits units ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getPreferredSpeed(units);
  }

  public void setPreferredSpeed( double speed ) {

        ext(com.anylogic.libraries.road.CarExtension.class).setPreferredSpeed(speed);
  }

  public void setPreferredSpeed( double speed, SpeedUnits units ) {

        ext(com.anylogic.libraries.road.CarExtension.class).setPreferredSpeed(speed, units);
  }

  @AnyLogicCustomProposalType(
AnyLogicCustomProposalType.Label.MPS_SQ )
  public double getAcceleration() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getAcceleration();
  }

  public double getAcceleration( AccelerationUnits units ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getAcceleration(units);
  }

  @AnyLogicCustomProposalType(
AnyLogicCustomProposalType.Label.MPS_SQ )
  public double getMaxAcceleration() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getMaxAcceleration();
  }

  public double getMaxAcceleration( AccelerationUnits units ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getMaxAcceleration(units);
  }

  @AnyLogicCustomProposalType(
AnyLogicCustomProposalType.Label.MPS_SQ )
  public double getMaxDeceleration() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getMaxDeceleration();
  }

  public double getMaxDeceleration( AccelerationUnits units ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getMaxDeceleration(units);
  }

  public void setMaxDeceleration( double maxDeceleration ) {

        ext(com.anylogic.libraries.road.CarExtension.class).setMaxDeceleration(maxDeceleration);
  }

  public void setMaxDeceleration( double maxDeceleration,
AccelerationUnits units ) {

        ext(com.anylogic.libraries.road.CarExtension.class).setMaxDeceleration(maxDeceleration, units);
  }

  public void setMaxAcceleration( double maxAcceleration ) {

        ext(com.anylogic.libraries.road.CarExtension.class).setMaxAcceleration(maxAcceleration);
  }

  public void setMaxAcceleration( double maxAcceleration,
AccelerationUnits units ) {

        ext(com.anylogic.libraries.road.CarExtension.class).setMaxAcceleration(maxAcceleration, units);
  }

  public boolean isAccelerating() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).isAccelerating();
  }

  public boolean isDecelerating() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).isDecelerating();
  }

  public boolean isOnForwardSide() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).isOnForwardSide();
  }

  public Road getRoad() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getRoad();
  }

  public Intersection getIntersection() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getIntersection();
  }

  public ParkingLot getParkingLot() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getParkingLot();
  }

  public BusStop getBusStop() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getBusStop();
  }

  public boolean isCarOn( AbstractRoadMarkup roadNetworkPart ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).isCarOn(roadNetworkPart);
  }
```

```
  public double getRoadOffset() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getRoadOffset();
  }

  public double getRoadOffset(LengthUnits units) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getRoadOffset(u
nits);
  }

  public boolean isChangingLane() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).isChangingLane(
);
  }

  public int getLaneIndex() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getLaneIndex();
  }

  public Agent getCarInFront() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getCarInFront();
  }

  public double getDistanceDriven() {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getDistanceDriv
en();
  }

  public double getDistanceDriven( LengthUnits units ) {
        return
ext(com.anylogic.libraries.road.CarExtension.class).getDistanceDriv
en(units);
  }

  public void resetDistanceDriven() {

        ext(com.anylogic.libraries.road.CarExtension.class).reset
DistanceDriven();
  }
}
```

3 Модуль Simulation Main

.

```
package diplom_final;

import java.io.Serializable;
import java.sql.Connection;
import java.sql.SQLException;
import java.util.ArrayDeque;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.Currency;
import java.util.Date;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Hashtable;
import java.util.Iterator;
import java.util.LinkedHashMap;
import java.util.LinkedHashSet;
import java.util.LinkedList;
import java.util.List;
import java.util.ListIterator;
import java.util.Locale;
import java.util.Map;
import java.util.PriorityQueue;
import java.util.Random;
import java.util.Set;
import java.util.SortedMap;
import java.util.SortedSet;
import java.util.Stack;
import java.util.Timer;
import java.util.TreeMap;
import java.util.TreeSet;
import java.util.Vector;
import java.awt.Color;
import java.awt.Font;
import com.anylogic.engine.connectivity.ResultSet;
import com.anylogic.engine.connectivity.Statement;
import com.anylogic.engine.elements.*;
import com.anylogic.engine.markup.Network;
import com.anylogic.engine.Position;
import com.anylogic.engine.markup.PedFlowStatistics;
import com.anylogic.engine.markup.DensityMap;
import static java.lang.Math.*;
import static com.anylogic.engine.UtilitiesArray.*;
import static com.anylogic.engine.UtilitiesCollection.*;
import static com.anylogic.engine.presentation.UtilitiesColor.*;
import static com.anylogic.engine.HyperArray.*;

import com.anylogic.engine.*;
import com.anylogic.engine.analysis.*;
import com.anylogic.engine.connectivity.*;
import com.anylogic.engine.database.*;
import com.anylogic.engine.gis.*;
import com.anylogic.engine.markup.*;
import com.anylogic.engine.presentation.*;
import com.anylogic.engine.gui.*;

import com.anylogic.libraries.pedestrian.*;
import com.anylogic.libraries.road.*;
import com.anylogic.libraries.processmodeling.*;



public class Simulation extends ExperimentSimulation<Main> {
  @AnyLogicInternalCodegenAPI
  public static String[] COMMAND_LINE_ARGUMENTS_xjal =
new String[0];
  {
    setCommandLineArguments_xjal(
COMMAND_LINE_ARGUMENTS_xjal );
  }
  @AnyLogicInternalCodegenAPI
  private static Map<String, IElementDescriptor>
elementDesciptors_xjal = createElementDescriptors(
Simulation.class );

  @AnyLogicInternalCodegenAPI
  @Override
  public Map<String, IElementDescriptor> getElementDesciptors() {
    return elementDesciptors_xjal;
  }
  // Области
  @AnyLogicInternalCodegenAPI
  protected static final Font _text3_Font = new Font("SansSerif", 0,
16 );
  @AnyLogicInternalCodegenAPI
  protected static final Font _text100_Font = new Font("SansSerif", 0,
26 );
  @AnyLogicInternalCodegenAPI
```

```
  protected static final Font _text_Font = _text100_Font;
  @AnyLogicInternalCodegenAPI
  protected static final Font _text2_Font = _text100_Font;
  protected static final Color _rectangle_Fill_Color = new Color(
0xB4FFFFFF, true );
  protected static final Color _rectangle7_Fill_Color = new Color(
0xFF505050, true );
  protected static final Color _rectangle110_Fill_Color = new Color(
0xFF505050, true );
  @AnyLogicInternalCodegenAPI
  protected static final int _image = 1;
  @AnyLogicInternalCodegenAPI
  protected static final int _rectangle = 2;
  @AnyLogicInternalCodegenAPI
  protected static final int _text3 = 3;
  @AnyLogicInternalCodegenAPI
  protected static final int _image1 = 4;
  @AnyLogicInternalCodegenAPI
  protected static final int _rectangle16 = 5;
  @AnyLogicInternalCodegenAPI
  protected static final int _rectangle7 = 6;
  @AnyLogicInternalCodegenAPI
  protected static final int _polyline1 = 7;
  @AnyLogicInternalCodegenAPI
  protected static final int _rectangle110 = 8;
  @AnyLogicInternalCodegenAPI
  protected static final int _text100 = 9;
  @AnyLogicInternalCodegenAPI
  protected static final int _text = 10;
  @AnyLogicInternalCodegenAPI
  protected static final int _text2 = 11;

  /** Internal constant, shouldn't be accessed by user */
  @AnyLogicInternalCodegenAPI
  protected static final int _SHAPE_NEXT_ID_xjal = 12;

  @AnyLogicInternalCodegenAPI
  protected static final double[] _polyline1_pointsDX_xjal() {
    return new double[] { 0.0, 0.0, 26.723, 0.0,  };
  }
  @AnyLogicInternalCodegenAPI
  protected static final double[] _polyline1_pointsDY_xjal() {
    return new double[] { 0.0, 30.0, 15.0, 0.0,  };
  }
  @AnyLogicInternalCodegenAPI
  protected static final double[] _polyline1_pointsDZ_xjal() {
    return new double[] { 0.0, 0.0, 0.0, 0.0,  };
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public boolean onShapeClick( int _shape, int index, double clickx,
double clicky ){
    switch( _shape ){
      case _rectangle7:
        if (true) {
          ShapeRectangle self = this.rectangle7;

if ( getState() == IDLE )
                run();
getExperimentHost().setPresentable( getEngine().getRoot() );
        }
        break;
      case _polyline1:
        if (true) {
          ShapePolyLine self = this.polyline1;

if ( getState() == IDLE )
                run();
getExperimentHost().setPresentable( getEngine().getRoot() );
        }
        break;
      default: return super.onShapeClick( _shape, index, clickx, clicky
);
                }
                return false;
  }

  protected ShapeImage image;
  protected ShapeRectangle rectangle;
  protected ShapeText text3;
  protected ShapeImage image1;
  protected ShapeRectangle rectangle16;
  protected ShapeRectangle rectangle7;
  protected ShapePolyLine polyline1;
  protected ShapeRectangle rectangle110;
  protected ShapeText text100;
  protected ShapeText text;
  protected ShapeText text2;
  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsBP0_xjal() {
    image = new ShapeImage(
                        Simulation.this, SHAPE_DRAW_2D, true,
0.0, 20.0, 0.0, 0.0,
                                    1240.0, 640.0, "/diplom_final/",
                                    new String[]{"image.png",}          );

    rectangle = new ShapeRectangle(
      SHAPE_DRAW_2D, true,0.0, 40.0, 0.0, 0.0,
        null, _rectangle_Fill_Color,
                                    1240.0, 685.0, 10.0, 1.0,
LINE_STYLE_SOLID );

    text3 = new ShapeText(
       SHAPE_DRAW_2D, true,20.0, 140.0, 0.0, 0.0,
       black,"Данная модель показывает работу  реальной
транспортной сети. Все светофоры имеют реальное время фаз
переключения. \r\nВ модели реализована система парковок, сеть
автобусных остановок.\r\nЦель модели - показать, справляется
ли транспортная система при большом потоке авто и пешеходов,
возможные решения при загруженности дороги.",
        _text3_Font, ALIGNMENT_LEFT );

    image1 = new ShapeImage(
                        Simulation.this, SHAPE_DRAW_2D, true,
20.0, 500.0, 0.0, 0.0,
                                    302.0, 59.0, "/diplom_final/",
                                    new
String[]{"anylogic_logo.png",}     );

    rectangle16 = new ShapeRectangle(
      SHAPE_DRAW_2D, true,277.916, 0.0, 0.0, 0.0,
        null, silver,
                                    962.016, 50.0, 10.0, 1.0,
LINE_STYLE_SOLID );

    rectangle7 = new ShapeRectangle(
      SHAPE_DRAW_2D, true,171.025, 0.0, 0.0, 0.0,
        null, _rectangle7_Fill_Color,
                                    128.269, 50.0, 10.0, 1.0,
LINE_STYLE_SOLID ) {

      @Override
      @AnyLogicInternalCodegenAPI
      public boolean onClick( double clickx, double clicky ) {
        return onShapeClick( _rectangle7, 0, clickx, clicky );
      }
    };

    polyline1 = new ShapePolyLine(
                        SHAPE_DRAW_2D, true, 185.0, 10.0, 0.0,
null, silver,
          4, _polyline1_pointsDX_xjal(), _polyline1_pointsDY_xjal(),
_polyline1_pointsDZ_xjal(), false, 10.0, 1.0, LINE_STYLE_SOLID
) {

      @Override
      @AnyLogicInternalCodegenAPI
      public boolean onClick( double clickx, double clicky ) {
        return onShapeClick( _polyline1, 0, clickx, clicky );
      }
    };
```

```
    rectangle110 = new ShapeRectangle(
      SHAPE_DRAW_2D, true,0.0, 0.0, 0.0, 0.0,
          null, _rectangle110_Fill_Color,
                                      171.025, 50.0, 10.0, 1.0,
LINE_STYLE_SOLID );

    text100 = new ShapeText(
      SHAPE_DRAW_2D, true,5.0, 10.0, 0.0, 0.0,
      white,"Мерзлий О.Д.",
      _text100_Font, ALIGNMENT_LEFT );

    text = new ShapeText(
      SHAPE_DRAW_2D, true,331.361, 10.0, 0.0, 0.0,
      white,"Імітаційна модель транспортних потоків",
      _text_Font, ALIGNMENT_LEFT );

    text2 = new ShapeText(
      SHAPE_DRAW_2D, true,215.0, 10.0, 0.0, 0.0,
      white,"Запуск",
      _text2_Font, ALIGNMENT_LEFT );

  }

  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsAP0_xjal() {
  }

  @AnyLogicInternalCodegenAPI
  private void _createPersistentElementsBS0_xjal() {
  }


  protected ShapeTopLevelPresentationGroup presentation;
  protected ShapeModelElementsGroup icon;

  @Override
  @AnyLogicInternalCodegenAPI
  public ShapeTopLevelPresentationGroup getPresentationShape() {
    return presentation;
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public ShapeModelElementsGroup getModelElementsShape() {
    return icon;
  }

  @Override
  public int getWindowWidth() {
    return 1240;
  }

  @Override
  public int getWindowHeight() {
    return 880;
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public void onDestroy_xjal() {
    super.onDestroy_xjal();
  }


  @Override
  @AnyLogicInternalCodegenAPI
  public void initDefaultRandomNumberGenerator(Engine _e) {
    _e.getDefaultRandomGenerator().setSeed( 1 );
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public Main createRoot( Engine engine ) {
    // Создание корневого объекта
    return new Main( engine, null, null );
  }

  @Override
  @AnyLogicInternalCodegenAPI
  public void setupRootParameters( final Main self, boolean
callOnChangeActions ) {
    final Main root = self; // for compatibility
    double p1_xjal;
    p1_xjal = self._p1_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_p1( p1_xjal );
    } else {
      self.p1 = p1_xjal;
    }
    double p2_xjal;
    p2_xjal = self._p2_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_p2( p2_xjal );
    } else {
      self.p2 = p2_xjal;
    }
    double p3_xjal;
    p3_xjal = self._p3_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_p3( p3_xjal );
    } else {
      self.p3 = p3_xjal;
    }
    double p10_xjal;
    p10_xjal = self._p10_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_p10( p10_xjal );
    } else {
      self.p10 = p10_xjal;
    }
    double p11_xjal;
    p11_xjal = self._p11_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_p11( p11_xjal );
    } else {
      self.p11 = p11_xjal;
    }
    double p12_xjal;
    p12_xjal = self._p12_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_p12( p12_xjal );
    } else {
      self.p12 = p12_xjal;
    }
    double Intensivnost1_xjal;
    Intensivnost1_xjal = self._Intensivnost1_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_Intensivnost1( Intensivnost1_xjal );
    } else {
      self.Intensivnost1 = Intensivnost1_xjal;
    }
    double Intensivnost2_xjal;
    Intensivnost2_xjal = self._Intensivnost2_DefaultValue_xjal();
    if (callOnChangeActions) {
      self.set_Intensivnost2( Intensivnost2_xjal );
    } else {
      self.Intensivnost2 = Intensivnost2_xjal;
    }
  }


  /**
   * Инициализация исполняющего модуля
   */
  @Override
  @AnyLogicInternalCodegenAPI
  public void setupEngine(Engine engine) {
    engine.setATOL( 1.0E-5 );
    engine.setRTOL( 1.0E-5 );
    engine.setTTOL( 1.0E-5 );
    engine.setHTOL( 0.001 );
```

```
    engine.setSolverODE( Engine.SOLVER_ODE_EULER );
    engine.setSolverNAE(
Engine.SOLVER_NAE_MODIFIED_NEWTON );
    engine.setSolverDAE( Engine.SOLVER_DAE_RK45_NEWTON
);
    engine.setVMethods( 427029 );
    engine.setSimultaneousEventsSelectionMode(
Engine.EVENT_SELECTION_LIFO );

    engine.setStartTime( 0.0 );
    engine.setTimeUnit( SECOND );
    engine.setStartDate( toDate( 2017, MAY, 2, 0, 0, 0 ) );
    engine.setRealTimeMode( true );
    engine.setRealTimeScale( 1.0 );
  }

  /**
   * Инициализация эксперимента
   */
  @Override
  @AnyLogicInternalCodegenAPI
  // TODO AL-12850 merge this setup(..) generation code in all
experiments - it has the same contents
  public void setup( IExperimentHost _e ) {
    setName( "Diplom_Final : Simulation" );
    // Статическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsBP0_xjal();
s
    // Динамическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsAP0_xjal();
    presentation = new ShapeTopLevelPresentationGroup(
Simulation.this, true, 0, 0, 0, 0 , image, rectangle, text3, image1,
rectangle16, rectangle7, polyline1, rectangle110, text100, text, text2
);
    // Динамическая инициализация элементов, у которых
разрешено программное управление
    _createPersistentElementsBS0_xjal();
    icon = new ShapeModelElementsGroup( Simulation.this,
getElementProperty( "diplom_final.Simulation.icon",
IElementDescriptor.MODEL_ELEMENT_DESCRIPTORS )  );
    _e.setZoomAndPanningEnabled( true );
    _e.setDeveloperPanelEnabled( true );
    _e.setDeveloperPanelVisibleOnStart( false );
  }

}
```

4 Модуль Main

//Инициализация каждого компонента и связей между ними

ЗАТВЕРДЖЕНИЙ

1116130.01206-01 13 01-ЛЗ

ДОСЛІДЖЕННЯ ТРАНСПОРТНИХ ПОТОКІВ ЗАСОБАМИ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

Опис програми

1116130.01206-01 13 01

Листів 26

2021

АНОТАЦІЯ

Документ 1116130.01206-01 13 01 «Дослідження транспортних потоків засобами імітаційного моделювання» є керівництвом для проектування та розробки програмного додатку «Імітаційна модель транспортних потоків», та входить до складу документації на проект.

Розробка імітаційної моделі транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання та поліпшення такої системи в реальному житті. В документі містяться основні вимоги до розробки програмного додатку та його функціональні можливості. Об’єм оперативної пам’яті, що займає програмний комплекс складає 400 мб. Конфігурація комп’ютера стандартна. Програма функціонує в середовищі Windows 10.

ЗМІСТ

1 ЗАГАЛЬНІ ВІДОМОСТІ

Розроблений додаток має назву «Імітаційна модель транспортних потоків».

Додаток був розрахований на функціонування в операційних системах MS Windows 7/8/8.1/10.

Додаток був реалізований на мові Java у програмному середовищі AnyLogic. Об’єм фізичної пам’яті, що займає програмний комплекс складає 13Мб. Конфігурація комп’ютера стандартна, але мінімальний об’єм оперативної пам’яті – 6 ГБ.

2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розробка імітаційної моделі транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання такої системи в реальному житті.

На основі моделі можна зібрати статистику і оптимізувати реальну транспортну мережу і запобігти появленню заторів і транспортного колапсу.

3 ОПИС ЛОГІЧНОЇ СТРУКТУРИ

3.1 Алгоритми роботи програми

Початок руху авто починається водному із блоків CarSourse. Приклад властивостей блоку приведено на рис 3.1.

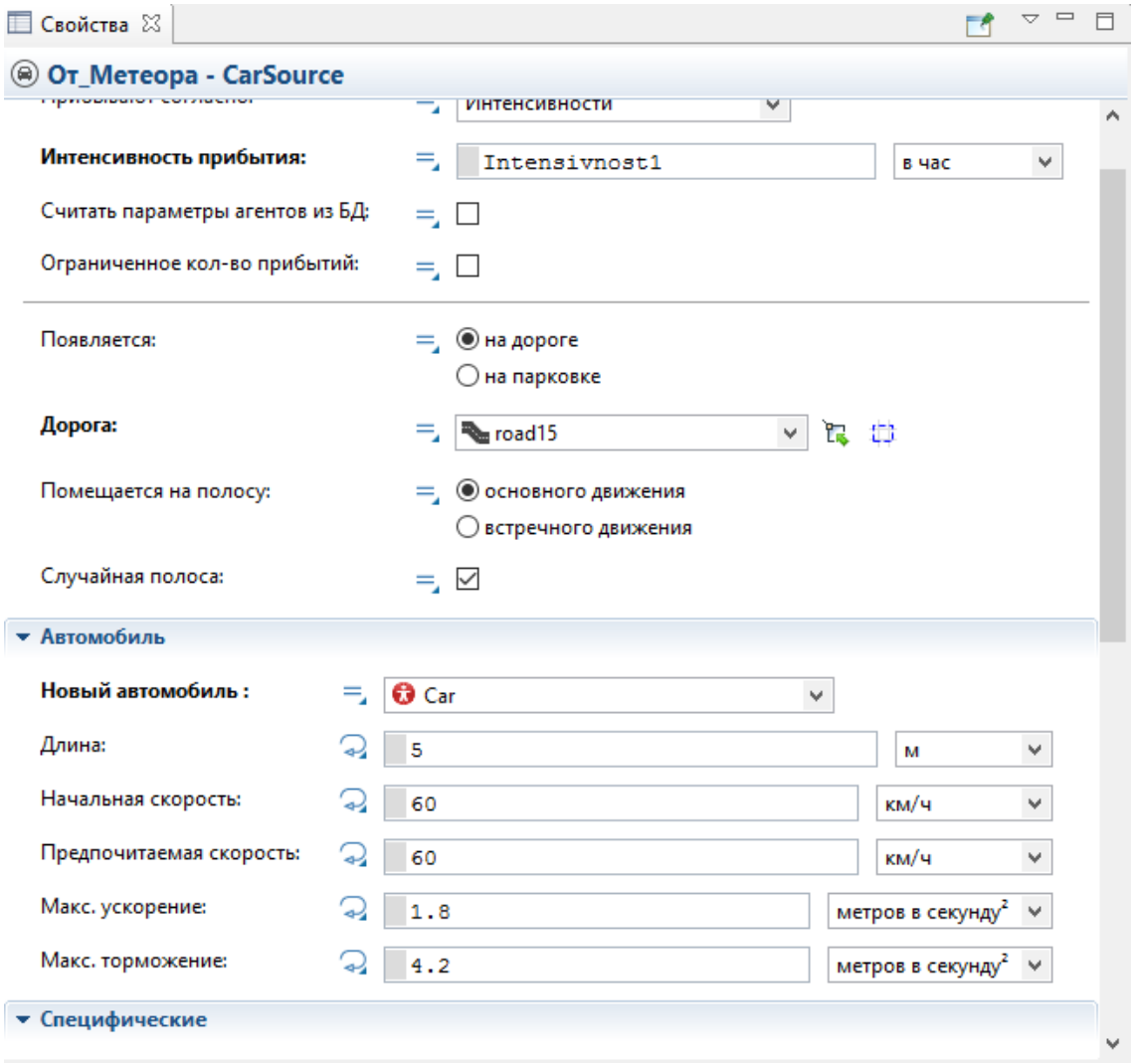


Рисунок 3.1 – Властивості блоку CarSourse

В кожному з блоків CarSourse можна вказати початкову дорогу або парковку, на якій будуть з'являтися автомобілі. Обрати тип автомобіля (агента, який відповідає за автомобіль), вказати початкові дані транспорту (початкову швидкість, середню швидкість, тощо.), задати інтенсивність потоку.

Після цього потік потрапляє в блок selectOutput5, який розподіляє, куди буде рухатися те чи інше авто. В блоці selectOutput5 є п'ять виходів із своїми

ймовірностями. Кожен із виходів має свою вірогідність використання, це показано на рис 3.2.

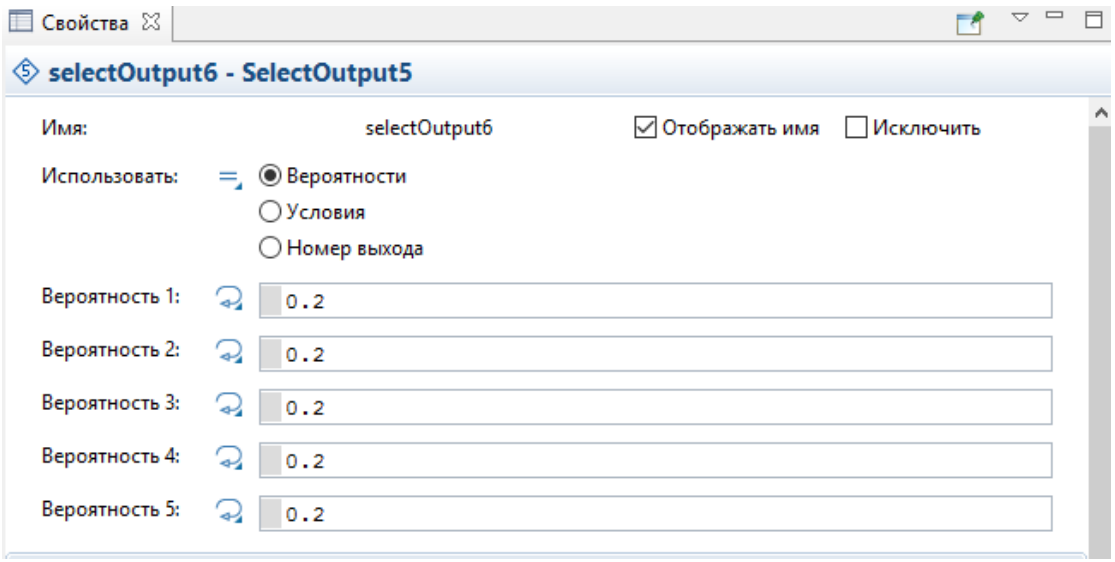


Рисунок 3.2 – Властивості блоку selectOutput5

Після обрання шляху авто потрапляє в блок selectOutput, який по властивостям схожий з блоком selectOutput5, але має лише 2 ймовірності з виходами:

1 По першій ймовірності, авто потрапляє в блок CarMoveTo, у властивостях якого вказано, до якої кінцевої дороги потрібно їхати автомобілю. Після приїзду до кінця обраної дороги, авто потрапляє до блоку carDispose, який видаляє авто з системи.

2 По другій ймовірності, авто потрапляє до блоку selectOutput5. Після цього блоку авто потрапляє в блок CarMoveTo, в якому вказано місце знаходження парковки. Якщо на парковці не має місця, то авто їде далі по логіці. Якщо місце вільне то авто паркується і потрапляє в блок Delay, в якому вказано скільки автомобілю потрібно очікувати часу. Після потрібного очікування авто їде далі по логіці моделі

Після пункту 2, авто знов потрапляє в блок вибору шляху, де по ймовірності обирається блок CarMoveTo. Авто їде до потрібної дороги, після чого видаляється з системи.

Логіка проїзду автомобілів в системі показано на рис 3.3.

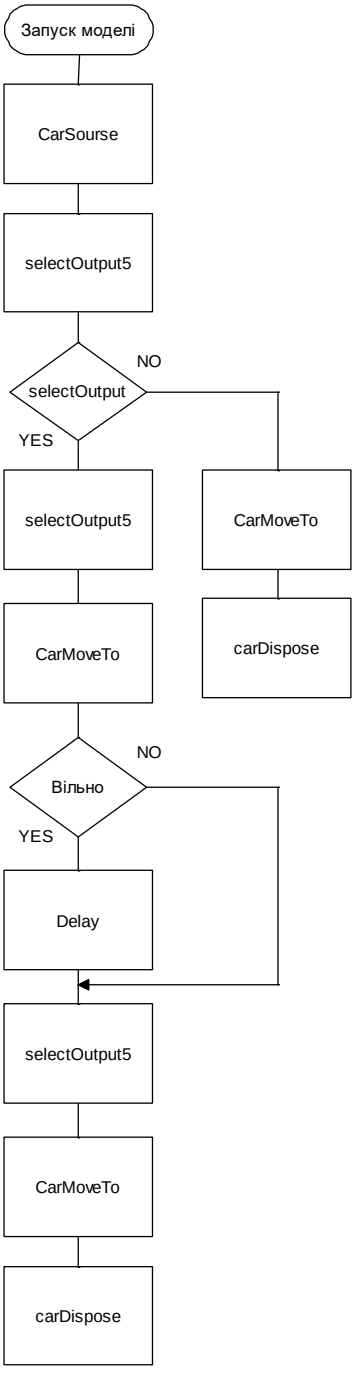


Рисунок 3.3 – Схема логіки автомобілів

Початок руху автобуса починається з блоку CarSourse. Властивості його такі, як показано на рис 3.1 окрім того, що блок має інший тип автомобілю, інші параметри та інтенсивність появи. Паралельно з ним працює блок

pedSourse, який відповідає за появу пішоходів в системі. Властивості цього блоку представлено на рис 3.4.

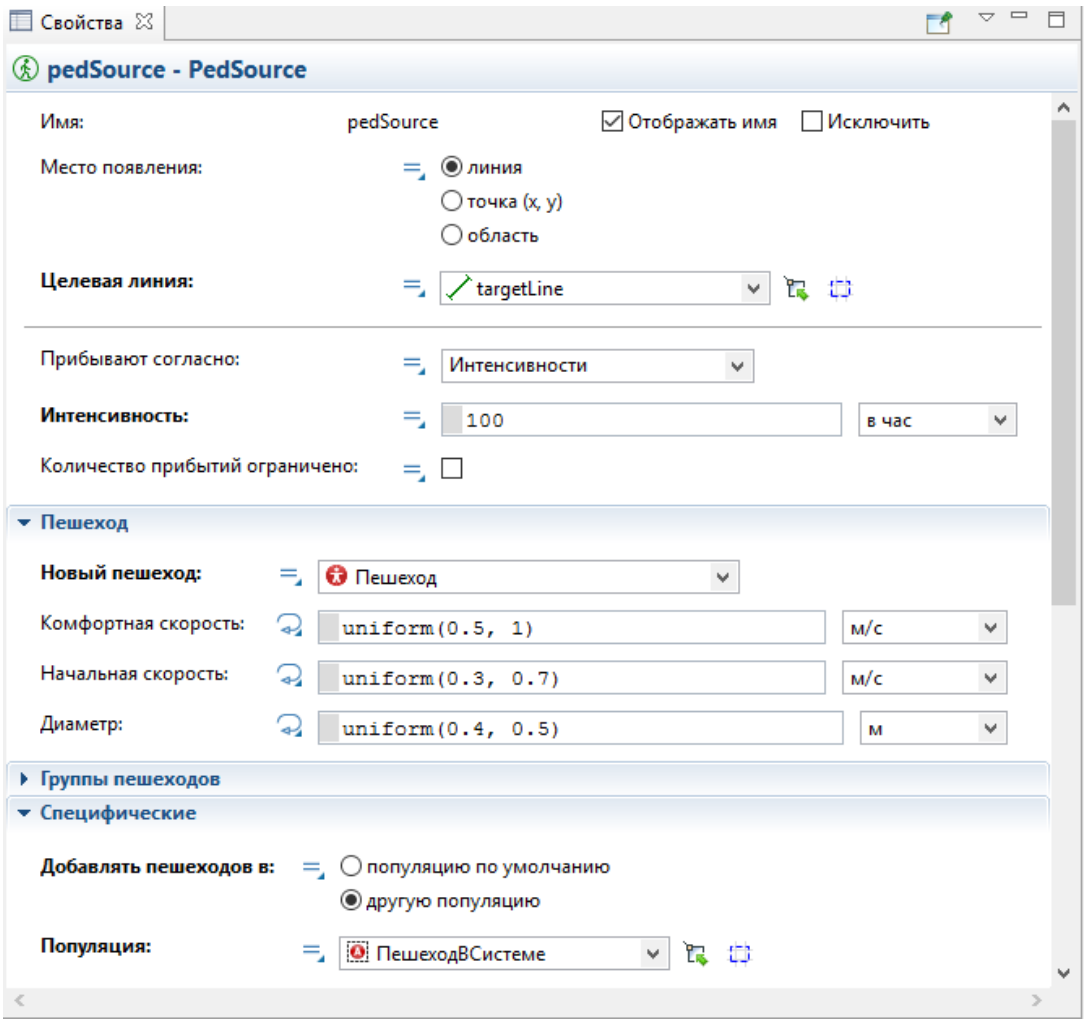


Рисунок 3.4 – Властивості блоку pedSourse

В властивостях блоку указано лінію, звідки з'являються пішоходи. Вказана їх інтенсивність появи, агент, популяція та швидкість.

Логіка автобуса наступна: автобуси з'являються в блоці CarSourse на дорозі, яка вказана в властивостях блоку. Після цього потрапляють в блок CarMoveTo, в якому вказано місце знаходження першої зупинки. Їдуть до неї після чого потрапляють в блок queue. Це блок черги, тобто якщо на зупинці стоять інші автобуси, автобус під'їжджає до зупинки і чекає поки звільниться місце, після чого заїжджає на зупинку і зупиняється, та потрапляє в блок pickup. Автобус підбирає пішоходів, як вказано в блоці pickup (максимальна

місткість автобуса – 45 пішоходів), після чого потрапляє в блок delay, в якому чекає відповідно до вказаного часу в блоці delay, і після чого їде до наступної зупинки. Ця логіка працює до поки автобус не під'їде до останньої зупинки, де потрапляє до блоку dropoff. На цьому етапі з автобуси виходять всі пішоходи і автобус від'їжджає від зупинки, їде до кінця дороги і потрапляє в блок CarDispose.

Логіка руху автобусів показано на рис 3.5.

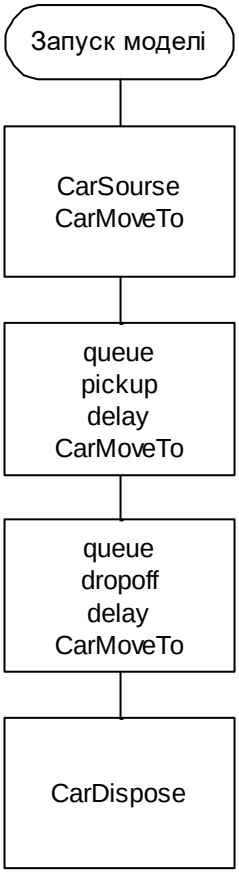


Рисунок 3.5 – Схема логіки автобусів

Логіка пішоходів така: пішоходи з'являються в області, яка вказана в блоці pedSourse. Після цього потрапляють в блок pedGoTo, в властивостях якого вказано, куди повинні йти пішоходи. Якщо через зупинку проїжджає декілька різних автобусів, то пішохід обирає, в який автобус йому сісти. І так на всіх зупинках, окрім останньої. На останній зупинці всі пішоходи виходять з

автобуса і йдуть до місця, яке вказано в блоці pedGoTo, після чого потрапляють в блок pedSink, який видаляє пішохода з системи.

Логіка руху пішоходів показано на рисунку 3.6.

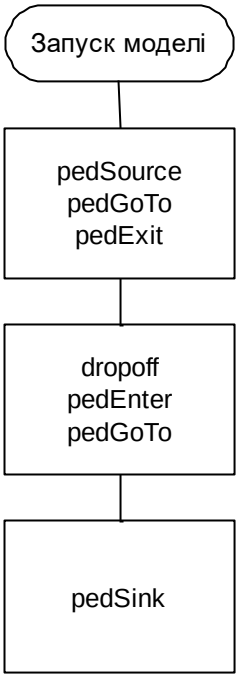


Рисунок 3.6 – Схема логіки пішоходів

Всі автомобілі рухаються за правилами дорожнього руху.

3.2 Використані методи

Було використано три основні методи:

– системна динаміка;
– дискретно-подієвого моделювання (процесно-орієнтований);
– агентне моделювання.

Дискретно-подійне моделювання – різновид імітаційного моделювання. В дискретно-подійному моделюванні функціонування системи представляється як хронологічна дискретна послідовність подій. Подія відбувається у визначений момент часу та призводить до зміни стану системи. Між цими подіями система перебуває у незмінному стані.

Агентне моделювання можна визначити як метод імітаційного моделювання, який досліджує поведінку децентралізованих агентів і то, як ця

поведінка визначає поведінку всієї системи в цілому. При розробці агентної моделі, інженер вводить параметри агентів (це можуть бути люди, компанії, активи, проекти, транспортні засоби, міста, тварини, тощо.), визначає їх поведінку, поміщає їх в навколишнє середовище, встановлює можливі зв'язки, після чого запускає моделювання. Індивідуальна поведінка кожного агента утворює глобальну поведінку моделі.

Системна динаміка – це підхід імітаційного моделювання, своїми методами і інструментами дозволяє зрозуміти структуру та динаміку складних систем. Також системна динаміка — це метод моделювання, який використовується для створення точних комп'ютерних моделей складних систем для подальшого використання з метою проектування більш ефективної організації і політики взаємин з даною системою. Разом, ці інструменти дозволяють створювати мікросвіти-симулятори, де простір і час можуть бути стиснуті і уповільнені так, щоб можна було вивчити наслідки рішень, швидко освоїти методи і зрозуміти структуру складних систем, спроектувати тактики і стратегії для більшого успіху

3.3 Структура програми з описом функції

Програма складається з трьох основних модулів. Призначення модулів програми:

Project.alp. – виконавчий модуль запуску моделі.

3d – папка с файлами формату .dae, у яких знаходяться логіка моделей.

x3d – папка с файлами формату .3d у яких знаходяться моделі проекту.

3.4 Зв’язки програми з іншими програмами

Для роботи повинен буди встановлений пакет SDK від компанії Oracle.

4 ТЕХНІЧНІ ЗАСОБИ, ЩО ВИКОРИСТОВУЮТЬСЯ

Мінімальна конфігурація:

– процесор: Intel Core i3 2.4 GHz або новіше;

– оперативна Пам'ять: 6 GB DDR3 або новіше;

– операційна система: Windows 7 або новіше;

– вільний дисковий простір 50 Mb;

– наявність CD/DVD приводу або USB роз’ємну для встановлення необхідного ПЗ;

– монітор з роздільною здатністю екрану 1024x1080;

– клавіатура;

– маніпулятор «миша».

5 ВИКЛИК ТА ЗАВАНТАЖЕННЯ

Для початку роботи моделі необхідно встановити та налаштувати середовище імітаційного моделювання AnyLogic. У середовищі вказати шлях до моделі. Якщо модель знаходиться на зовнішньому носії, то її можна завантажити з них або скопіювавши на жорсткий диск.

Запуск моделі показано на рис 5.1.

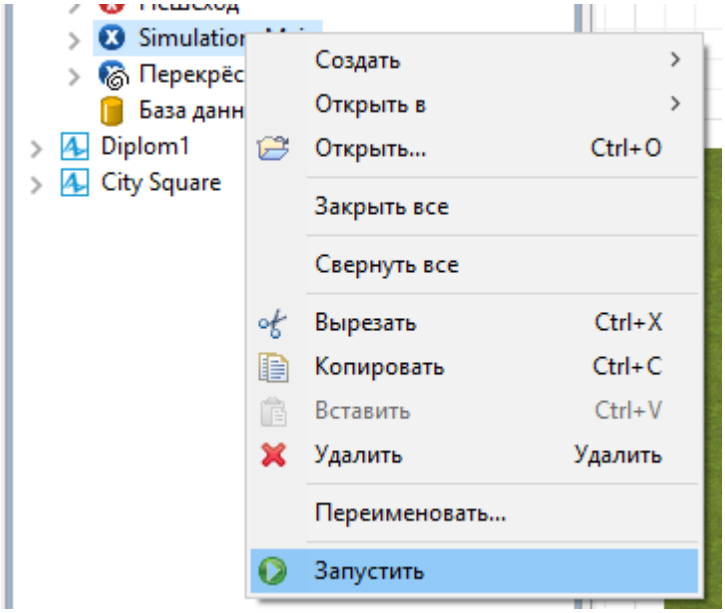


Рисунок 5.1 – Запуск моделі

При запуску моделі на екрані з'являється початкове вікно, представлене на рис 5.2.

Для запуску моделі потрібно натиснути на кнопку . Після цього почнеться повна робота імітаційної моделі.

Рисунок 5.2 – Початкове вікно моделі

6 ВХІДНІ ДАНІ

Вхідними даними для моделі служать статистичні показники, такі як:

- час роботи кожного світлофора в моделі;
- кількість автомобілів у системи за годину;
- кількість міського транспорту на годину;
- кількість людей на зупинках на годину;

7 ВИХІДНІ ДАНІ

До вихідних даних відносяться результати роботи імітаційної моделі, такі як сітьові графіки, статистичні данні проїзду системи кожного автомобіля і робота загалом кожного агенту.

8 ОПИС ІНТЕРФЕЙСУ КОРИСТУВАЧА

Для запуску моделі потрібно натиснути на кнопку  .

Після цього почнеться повна робота імітаційної моделі. Перше що буде показано це 2D модель, як показано на рис 8.1.

Рисунок 8.1 – 2D модель перехрестя

На цій 2D моделі показано, як рухаються авто, як працюють світлофори. Якщо натиснути на Show density map, то модель покаже, на якій ділянці дороги ускладнений рух. Зеленим кольором – ця ділянка дороги пуста. Жовтим кольором – на ділянці дороги рух не ускладнений. Червоним – ділянка дороги, де дуже ускладнений рух транспорту. Це продемонстровано на рис 8.2.

Щоб перейти до наступного пункту потрібно на панелі навігації натиснути потрібну кнопку.

При натисканні кнопки 3D з'явиться вікно перегляду у форматі 3D. Що представлено на рис 8.3.

Рисунок 8.2 – Натиснутий Show density map

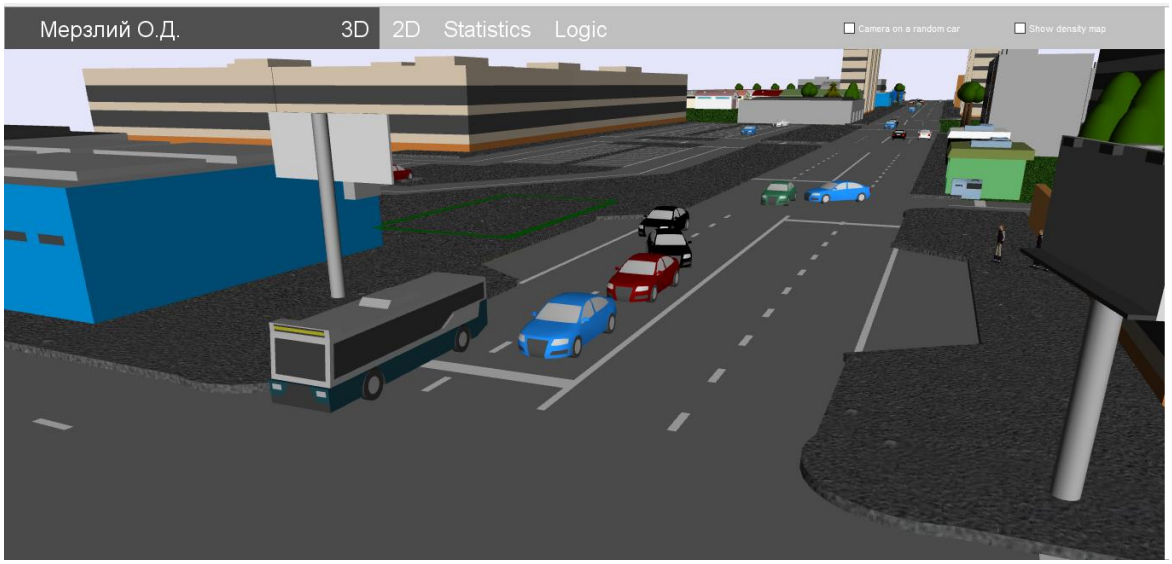


Рисунок 8.3 – Вкладка 3D

Аналогічно вікну 2D є кнопка Show density map. При натисканні на Camera on random car область перегляду буде сфокусована на один будь-який автомобіль доти, поки він не поїде геть з системи, показано на рис 8.4.

Рисунок 8.4 – Camera on random car

Навігація в вікні відбувається мишею.

При натисканні кнопки Statistics буде показано статистику моделі, представлено на рис 8.5 – 8.6.

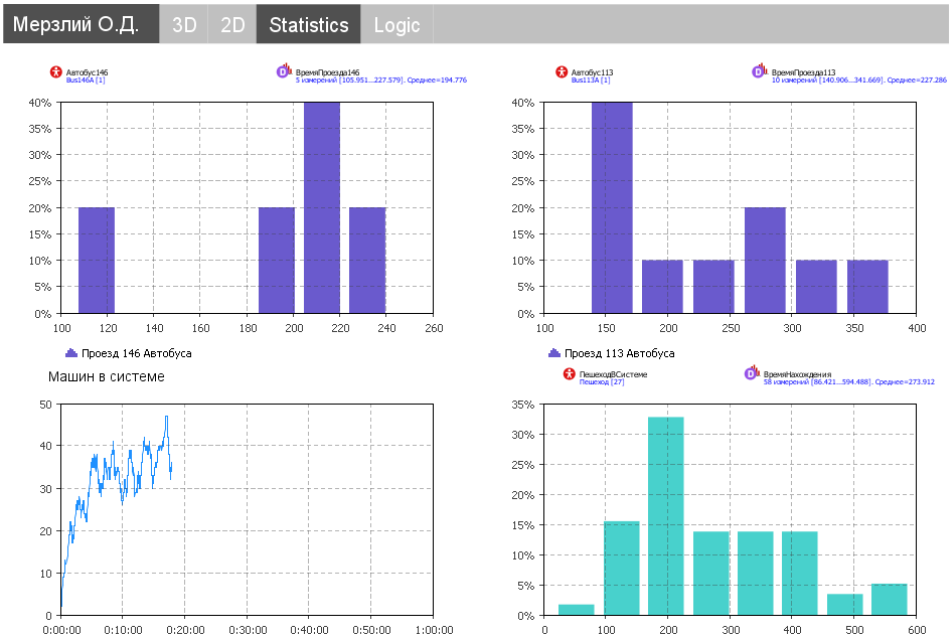


Рисунок 8.5 – Статистика моделі

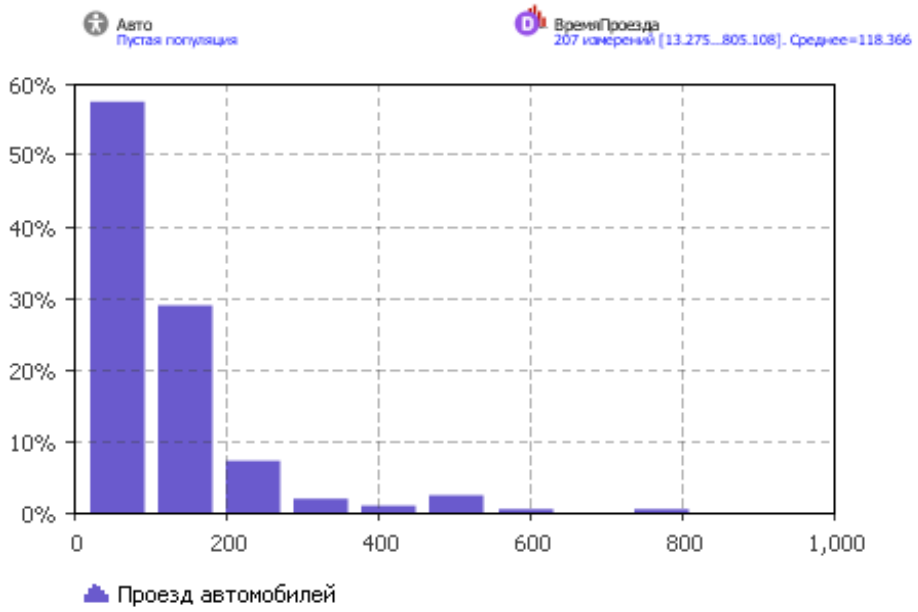


Рисунок 8.6 – Статистика моделі

Перша гістограма показує середній час проїзду по всім автобусним зупинкам і виходу з моделі автобуса під номером 146А.

Друга гістограма аналогічно з першою показує час середній проїзду автобуса під номером 113А.

Третій графік покаже, скільки автомобілів знаходиться одночасно в системі.

Четверта гістограма – як довго пішоходи будуть знаходитися на зупинках, і в системі цілому.

П'ята гістограма покаже середній час проїзду автомобілів через систему.

При натисканні на змінні буде виведено статистику по роботі цих змінних, показано на рис 8.7.

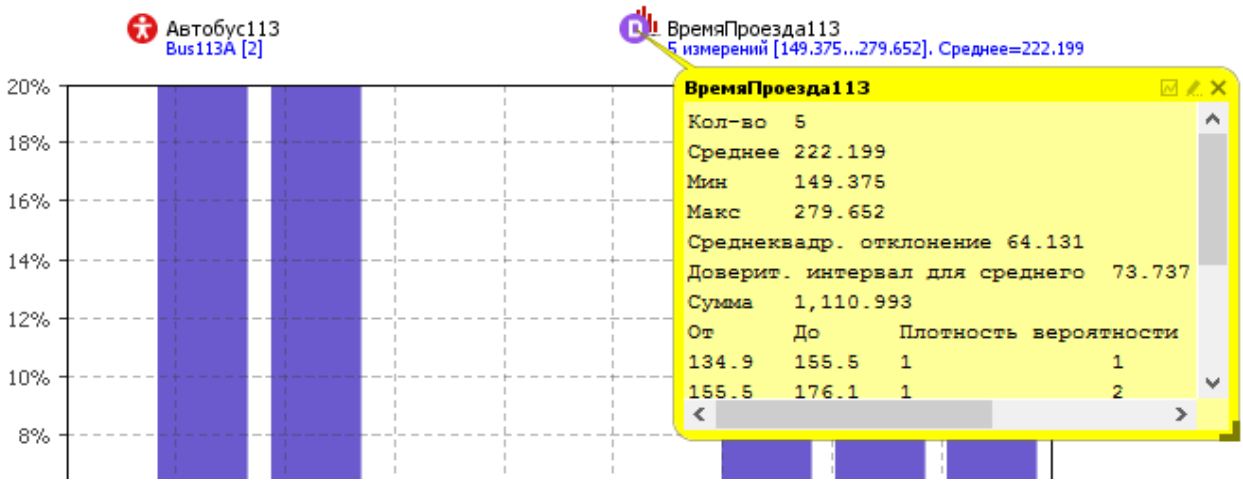


Рисунок 8.7 – Статистика змінної

При натисканні кнопки Logic буде показано область з логікою моделі. В ній можна побачити як працює імітаційна модель і зробити висновки щодо коректності моделі. При натисканні на будь-який елемент буде показано статистику по його дії. Приклад приведено на рис 8.8 – 8.9.

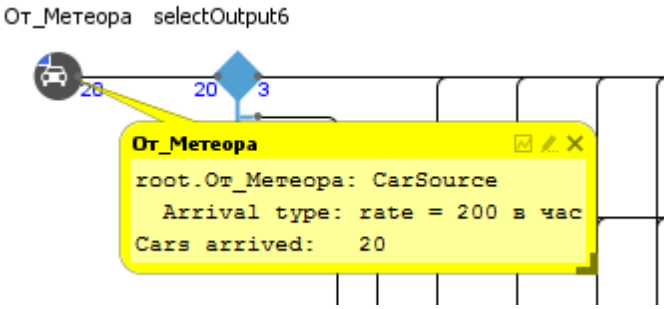


Рисунок 8.8 – Статистика елемента CarSourse

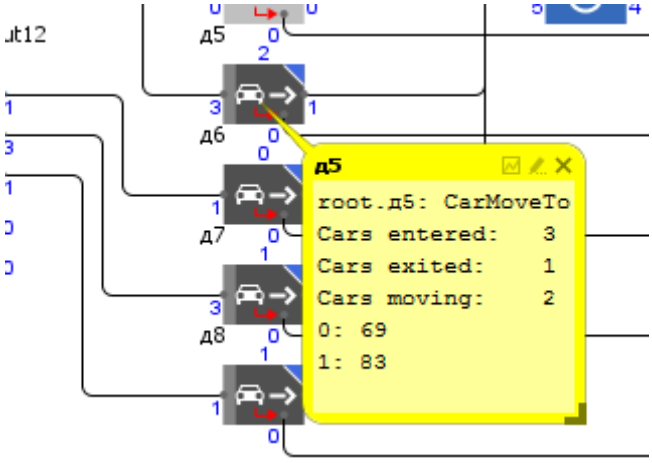


Рисунок 8.9 – Статистика елемента CarMoveTo

Для запуску процесу оптимізації потрібно обрати відповідний процес, процес запуску показано на рис 8.10.

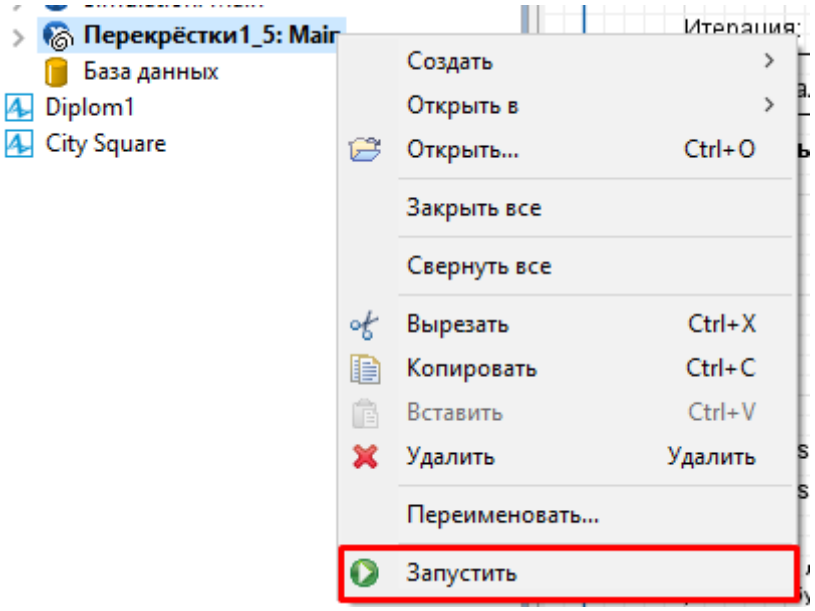


Рисунок 8.10 – Запуск процесу оптимізації

Цей експеримент дозволить оптимізувати імітаційну модель, а саме оптимізувати фази світлофорів, щоб не утворювалися затори, або їх мінімізувати. Після виконання процесу оптимізації буде показано результат, приклад приведено на рис 8.11.

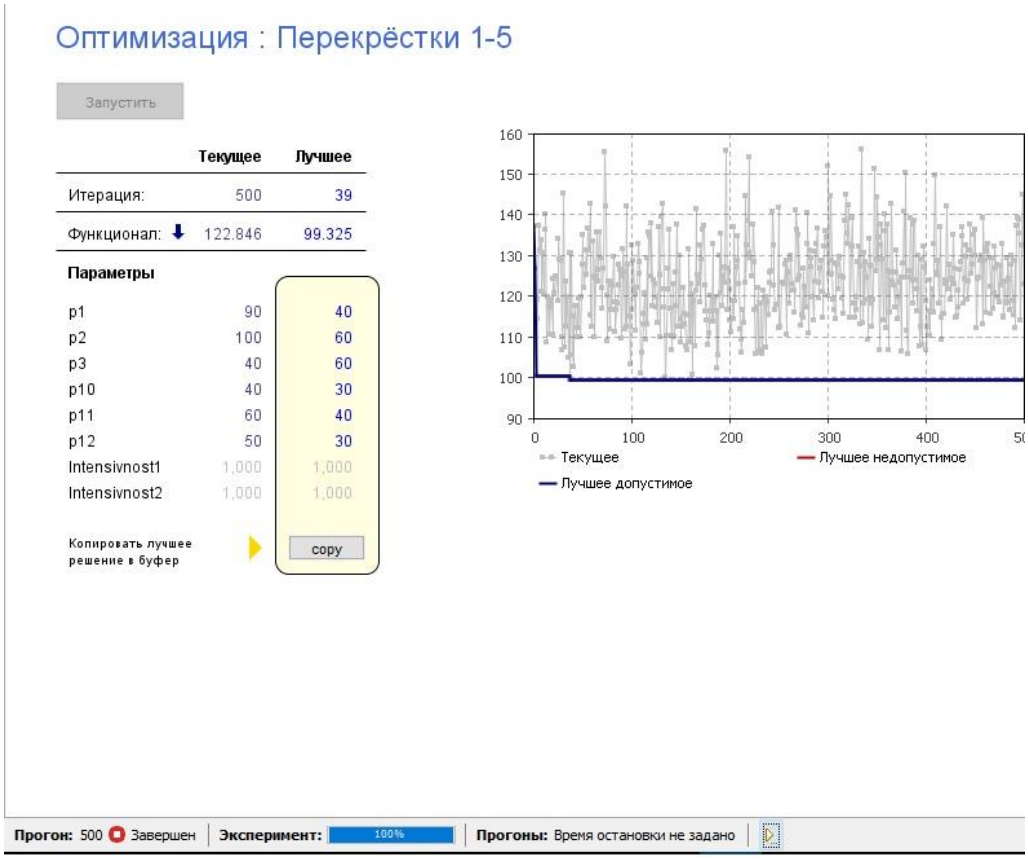


Рисунок 8.11 – Процес оптимізації

При виконанні оптимізації імітаційна модель виконується приблизно 500 раз, і кожен раз змінюються параметри фаз світлофорів. На рис 8.11 ми бачимо кращий час проїзду автомобілю по системі, кращий час фаз світлофорів. Бачимо графік, на якому зображений час проїзду авто при кожній ітерації.

Після цього процесу ми можемо в моделі вказати дані, які отримали, тим саме оптимізувати імітаційну модель

9 ПОРЯДОК РАБОТИ З ПРОГРАМОЮ

Технологія роботи і порядок виконання моделі:

1 Внести до моделі попередні налаштування, вказати час роботи світлофорів, інтенсивність руху і щільність трафіку.

2 Виконати запуск моделі.

3 Дочекатись виконання моделі.

4 Запустити процес оптимізації.

5 Дані після оптимізації внести до моделі і запустити модель.

6 Повторити пункт 1-5 декілька разів. Кожен раз змінювати інтенсивність і змінювати навантаження.

7 Експериментами визначити найкращій час роботи світлофорів, оптимальну кількість і час появи міського транспорту, час роботи світлофорів.

10 ПОВІДОМЛЕННЯ КОРИСТУВАЧУ

У таблиці 10.1 наведені повідомлення користувачу, що можуть з’явитись у процесі моделювання.

Таблиця 10.1 – Повідомлення користувачу

Текст повідомлення	Опис ситуації	Рекомендовані дії
1	2	3
“Помилка! Введено не вірне значення”	Було вказано не правильне значення	Перевірити правильність введених даних
“Помилка! Не правильні налаштування агенту”	При налаштуванні агента, було введено не коректні значення	Перевірити правильність введених даних
“Робота моделі завершена з часом XXX”	Успішне завершення імітації	Натиснути кнопку «Ок»

ЗАТВЕРДЖЕНИЙ

1116130.01206-01 ІЗ 01-ЛЗ

ДОСЛІДЖЕННЯ ТРАНСПОРТНИХ ПОТОКІВ ЗАСОБАМИ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

Керівництво користувача

Керівництво експерта з моделювання та прогнозування

1116130.01206-01 ІЗ 01

Листів 17

2021

АНОТАЦІЯ

Документ 1116130.01206-01 ІЗ 01 «Дослідження транспортних потоків засобами імітаційного моделювання» є керівництвом користувача програмного додатку «Імітаційна модель транспортних потоків», та входить до складу документації на проект.

Розробка імітаційної моделі транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання та поліпшення такої системи в реальному житті. В документі містяться основні вимоги до розробки програмного додатку та його функціональні можливості. Об’єм оперативної пам’яті, що займає програмний комплекс складає 400 мб. Конфігурація комп’ютера стандартна. Програма функціонує в середовищі Windows 10.

ЗМІСТ

1 ВСТУП

Розроблений додаток має назву «Імітаційна модель транспортних потоків».

Додаток був розрахований на функціонування в операційних системах MS Windows 7/8/8.1/10.

Додаток застосовується для створення імітації роботи транспортної мережі для того, щоб ставити експерименти і збирати необхідну статистику для оптимізації транспортної мережі.

Додаток надає такі можливості:

- імітувати роботу транспортної системи;
- змінювати час фаз переключення світлофорів;
- змінювати інтенсивність появи авто у системі;
- змінювати інтенсивність появи пішоходів у системі;
- переглядати статистику по всій моделі.

Користувачу не потрібні знання програмування, лише базові навички роботи з ПК та знати керівництво користувача.

2 ПРИЗНАЧЕННЯ ТА УМОВИ ЗАСТОСУВАННЯ

Цей додаток призначений для спеціалістів, які займаються побудовою автодоріг, або операторів, які слідкують за цими дорогами.

Імітаційна модель працює у таких ОС:

- Microsoft Windows, 10, x86-32 і x64;
- Microsoft Windows 8, x86-32 і x64;
- Microsoft Windows 7 SP1, x86-32 і x64;
- AppleMac OS X 10.7.3 (Lion) або вище, універсальний;
- SuSELinux, x86-32 (з встановленими GTK +, libwebkitgtk-1.0-0, libudev, libssl 0.9.8 і вище);
- Ubuntu Linux 10,04 або вище, x86-32 (з встановленими GTK +, libwebkitgtk-1.0-0, libudev, libssl 0.9.8 і вище).

Для роботи моделі потрібно Java 2 Standard Edition 8.0 або вище. Повинен бути встановлений користувачем самостійно.

Мінімальна конфігурація:

- процесор: Intel Core i3 2.4 GHz або новіше;
- оперативна Пам'ять: 6 GB DDR3 або новіше;
- операційна система: Windows 7 або новіше;
- вільний дисковий простір 50 Mb;
- наявність CD/DVD приводу або USB роз’ємну для встановлення необхідного ПЗ;
- монітор з роздільною здатністю екрану 1024x1080;
- клавіатура;
- маніпулятор «миша».

3 ПІДГОТОВКА ДО РОБОТИ

До складу дистрибутиву повинні входити файли, показано на рис 3.1.

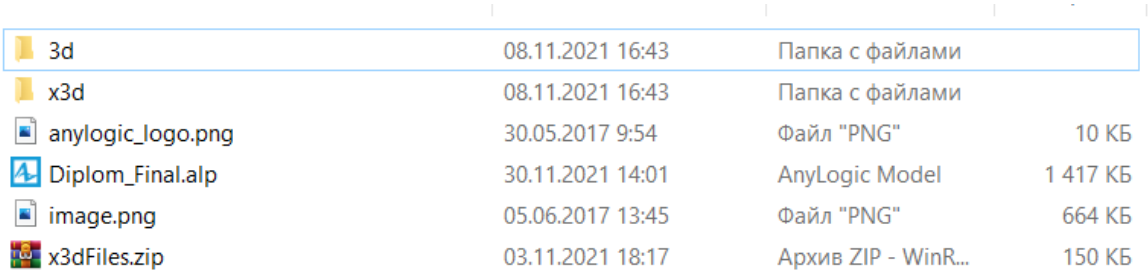


Рисунок 3.1 – Дистрибутив додатку

Для запуску моделі натиснути два рази на файл Diplom_final.alp.

Для початку роботи моделі необхідно встановити та налаштувати середовище імітаційного моделювання AnyLogic. У середовищі вказати шлях до моделі. Якщо модель знаходиться на зовнішньому носії, то її можна завантажити з них або скопіювавши на жорсткий диск.

Перед початком експерименту потрібно перевірити логіку роботи моделі. Перевірити усі зв’язки, правильність налаштування світлофорів і об’єктів в цілому. Перевірити правильність налаштування часових характеристик. Перевірити кількість вільної оперативної пам’яті (потрібно не менше 4 гігабайт). Перевірити навантаження процесору (не більше 20%).

4 ОПИС ОПЕРАЦІЙ

Запуск моделі показано на рис 4.1.

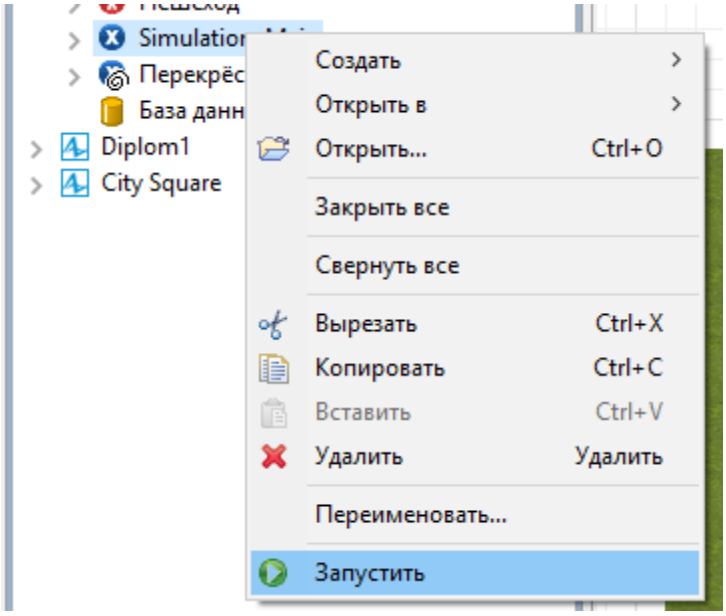


Рисунок 4.1 – Запуск моделі

При запуску моделі на екрані з'являється початкове вікно, представлене на рис 4.2.

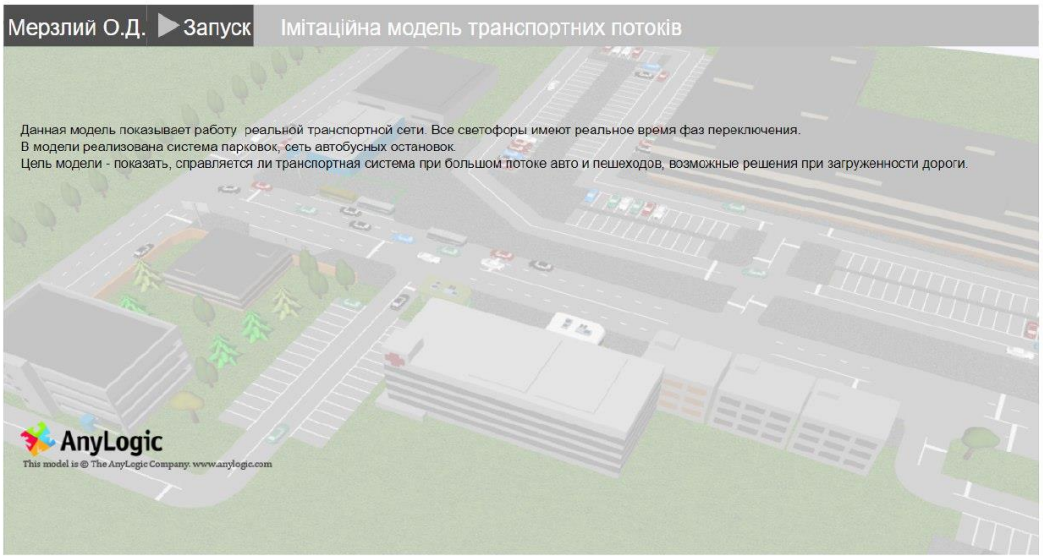


Рисунок 4.2 – Початкове вікно моделі

Для запуску моделі потрібно натиснути на кнопку . Після цього почнеться повна робота імітаційної моделі.

Після цього почнеться повна робота імітаційної моделі. Перше що буде показано це 2D модель, як показано на рис 4.3.

Рисунок 4.3 – 2D модель перехрестя

На цій 2D моделі показано, як рухаються авто, як працюють світлофори. Якщо натиснути на Show density map, то модель покаже, на якій ділянці дороги ускладнений рух. Зеленим кольором – ця ділянка дороги пуста. Жовтим кольором – на ділянці дороги рух не ускладнений. Червоним – ділянка дороги, де дуже ускладнений рух транспорту. Це продемонстровано на рис 4.4.

Щоб перейти до наступного пункту потрібно на панелі навігації натиснути потрібну кнопку.

При натисканні кнопки 3D з'явиться вікно перегляду у форматі 3D. Що представлено на рис 4.5.

Рисунок 4.4 – Натиснутий Show density map

Рисунок 4.5 – Вкладка 3D

Аналогічно вікну 2D є кнопка Show density map. При натисканні на Camera on random car область перегляду буде сфокусована на один будь-який автомобіль доти, поки він не поїде геть з системи, показано на рис 4.6.

Рисунок 4.6 – Camera on random car

Навігація в вікні відбувається мишею.

При натисканні кнопки Statistics буде показано статистику моделі, представлено на рис 4.7 – 4.8.

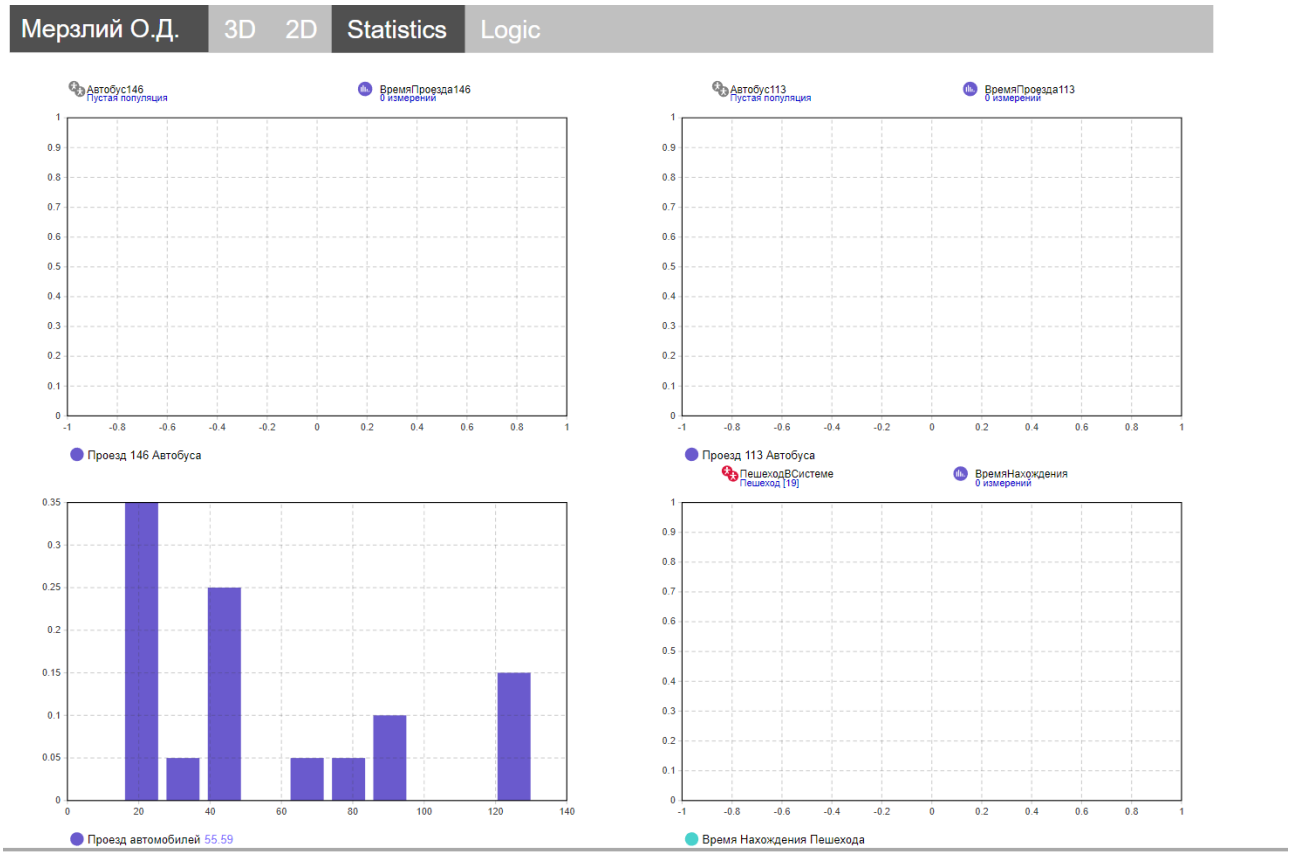


Рисунок 4.7 – Статистика моделі

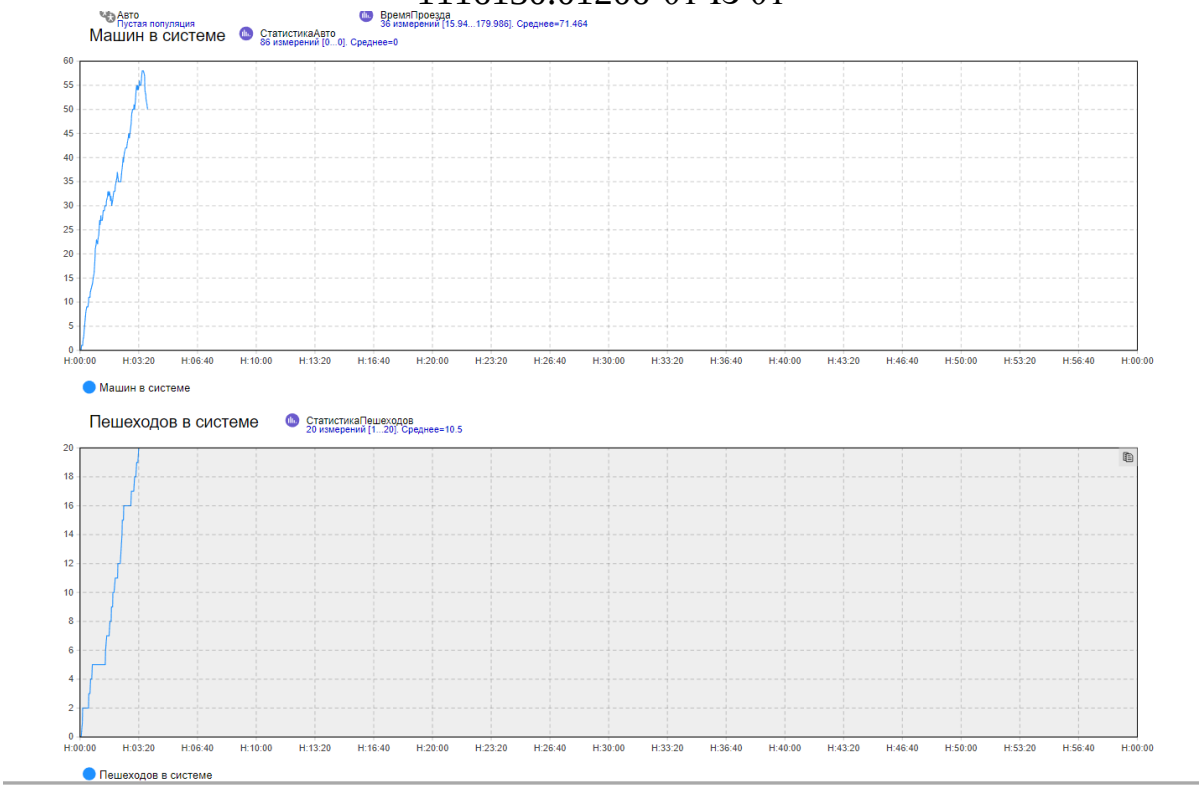


Рисунок 4.8 – Статистика моделі

Перша гістограма показує середній час проїзду по всім автобусним зупинкам і виходу з моделі автобуса під номером 146А.

Друга гістограма аналогічно з першою показує час середній проїзду автобуса під номером 113А.

Третій графік покаже середній час проїзду автомобілів через систему.

Четверта гістограма – як довго пішоходи будуть знаходитися на зупинках, і в системі цілому.

П’ята гістограма покаже, скільки автомобілів знаходиться одночасно в системі.

Шоста покаже скільки пішоходів одночасно у системі.

При натисканні на змінні буде виведено статистику по роботі цих змінних, показано на рис 4.9.

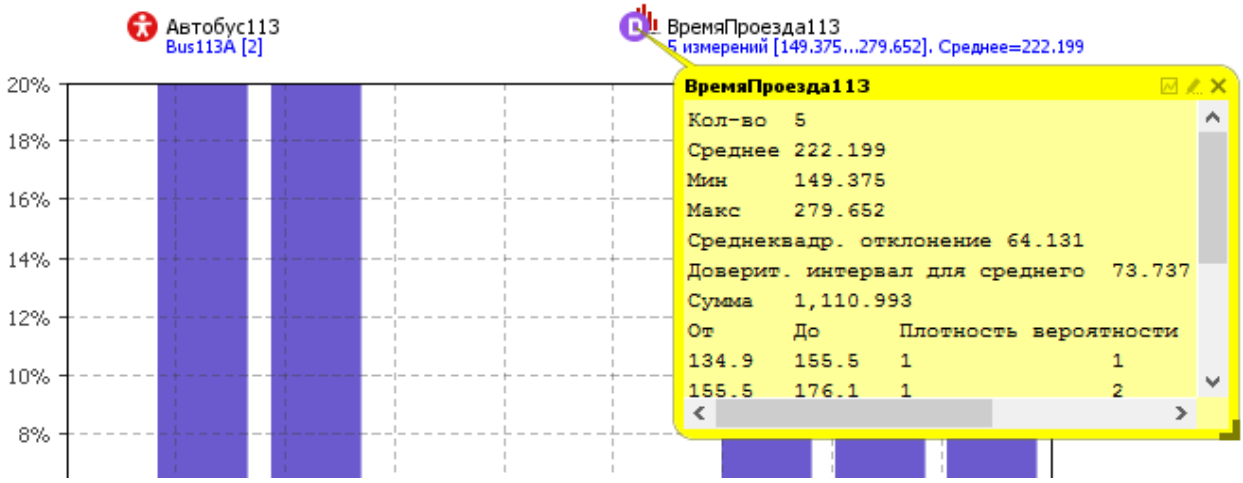


Рисунок 4.9 – Статистика змінної

При натисканні кнопки Logic буде показано область з логікою моделі. В ній можна побачити як працює імітаційна модель і зробити висновки щодо коректності моделі. При натисканні на будь-який елемент буде показано статистику по його дії. Приклад приведено на рис 4.10 – 4.11.

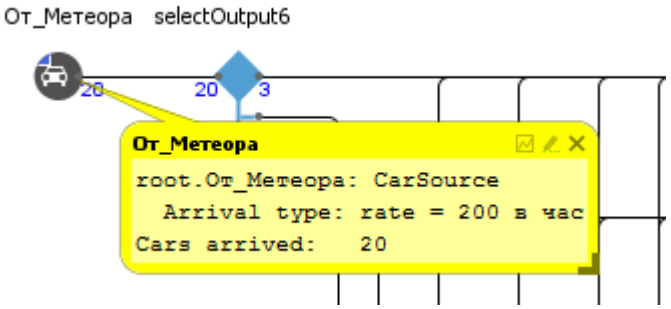


Рисунок 4.10 – Статистика елемента CarSourse

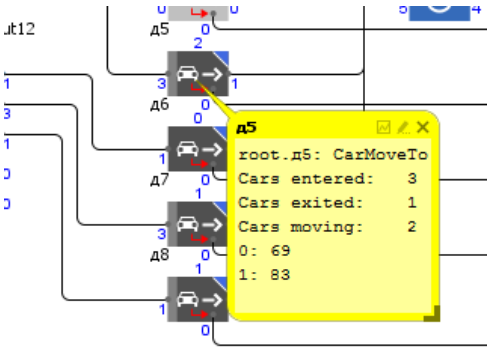


Рисунок 4.11 – Статистика елемента CarMoveTo

Для запуску процесу оптимізації потрібно обрати відповідний процес, процес запуску показано на рис 4.12.

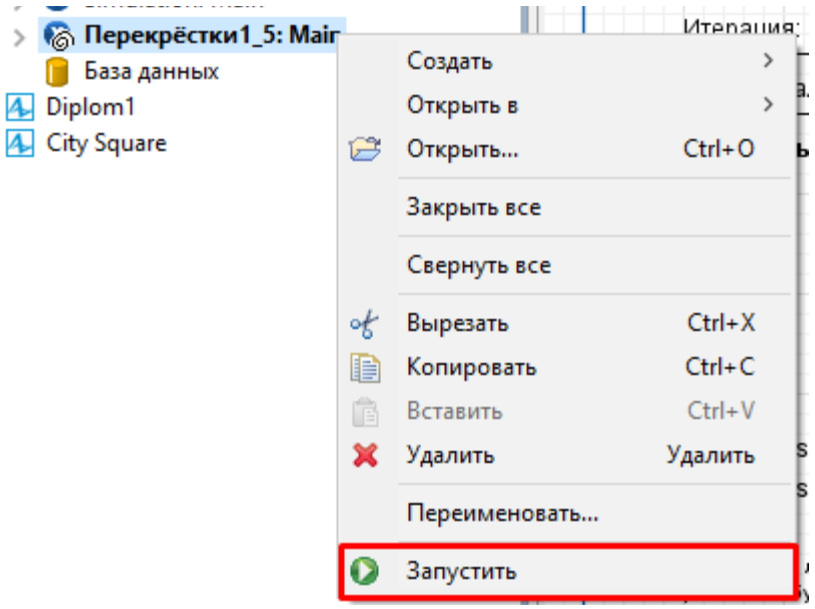


Рисунок 4.12 – Запуск процесу оптимізації

Цей експеримент дозволить оптимізувати імітаційну модель, а саме оптимізувати фази світлофорів, щоб не утворювалися затори, або їх мінімізувати. Після виконання процесу оптимізації буде показано результат, приклад приведено на рис 4.13.

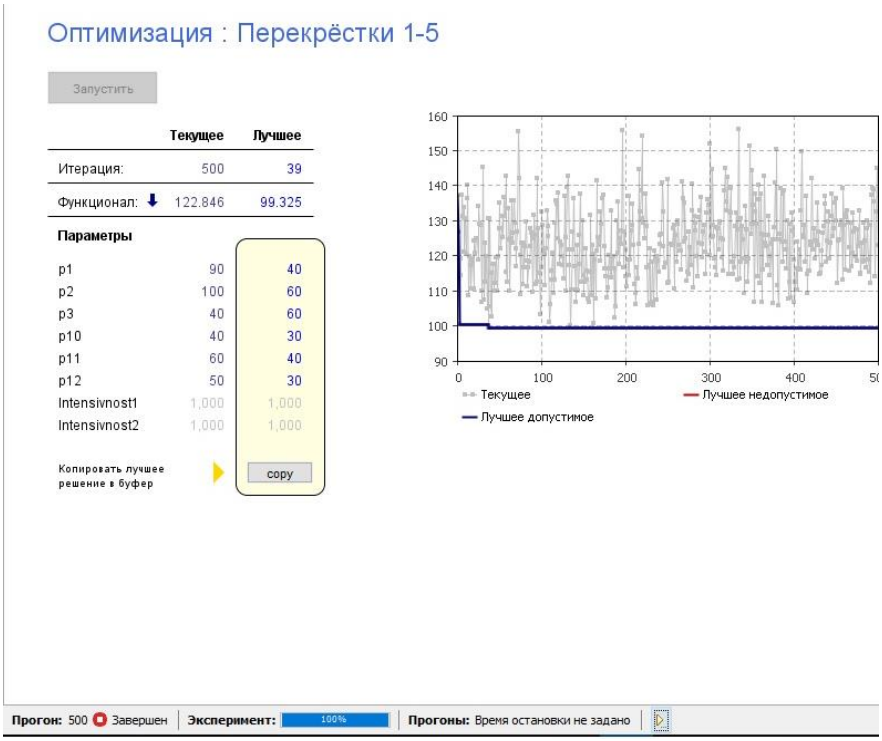


Рисунок 4.13 – Процес оптимізації

При виконанні оптимізації імітаційна модель виконується приблизно 500 раз, і кожен раз змінюються параметри фаз світлофорів. На рис 4.13 ми бачимо кращий час проїзду автомобілю по системі, кращий час фаз світлофорів. Бачимо графік, на якому зображений час проїзду авто при кожній ітерації.

Після цього процесу ми можемо в моделі вказати дані, які отримали, тим саме оптимізувати імітаційну модель.

Для проведення експерименту з різними даними потрібно запустити моделі, і у полях вводу змінити параметри, приклад показано на рисунку 4.14.

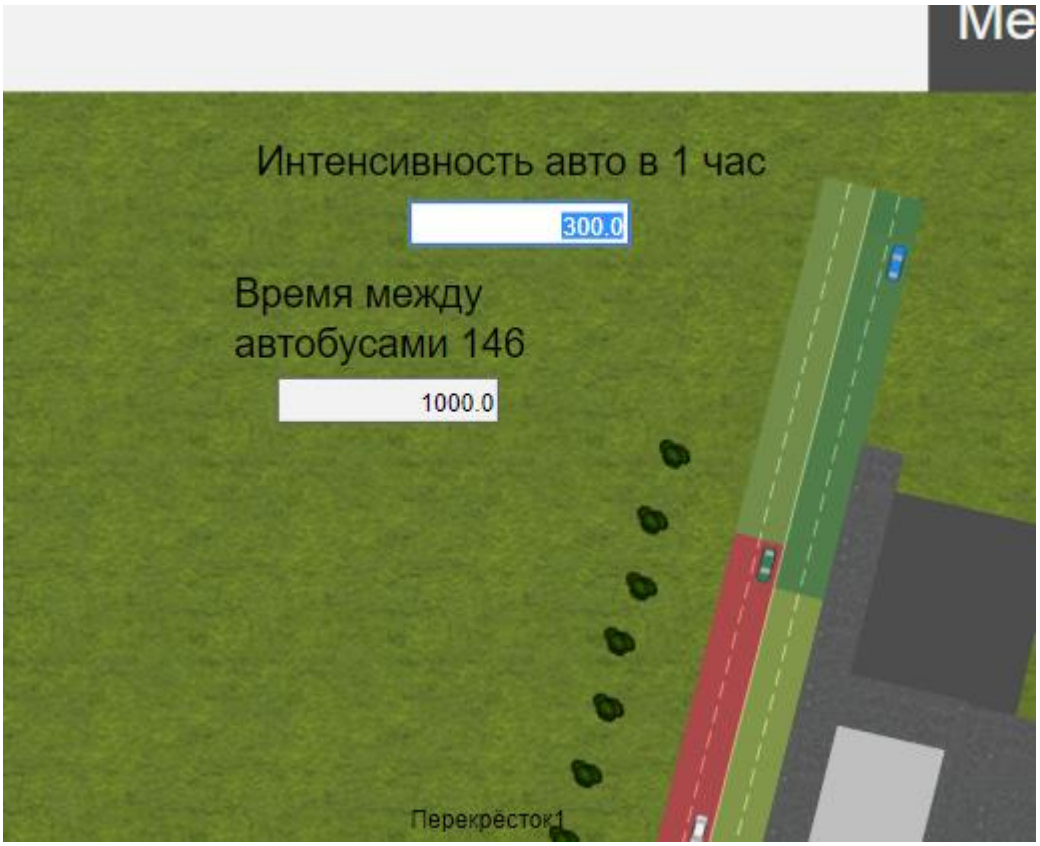


Рисунок 4.14 – Зміна параметру

Технологія роботи і порядок виконання моделі:

1 Внести до моделі попередні налаштування, вказати час роботи світлофорів, інтенсивність руху і щільність трафіку.

2 Виконати запуск моделі.

3 Дочекатись виконання моделі.

4 Запустити процес оптимізації.

5 Дані після оптимізації внести до моделі і запустити модель.

6 Повторити пункт 1-5 декілька разів. Кожен раз змінювати інтенсивність і змінювати навантаження.

Експериментами визначити найкращій час роботи світлофорів, оптимальну кількість і час появи міського транспорту, час роботи світлофорів.

5 АВАРІЙНІ СИТУАЦІЇ

У таблиці 4.1 наведені повідомлення користувачу, що можуть з’явитись у процесі моделювання.

Таблиця 4.1 – Повідомлення користувачу

Текст повідомлення	Опис ситуації	Рекомендовані дії
1	2	3
“Помилка! Введено не вірне значення”	Було вказано не правильне значення	Перевірити правильність введених даних
“Помилка! Не правильні налаштування агенту”	При налаштуванні агента, було введено не коректні значення	Перевірити правильність введених даних
“Робота моделі завершена з часом XXX”	Успішне завершення імітації	Натиснути кнопку «Ок»

При появі аварійної ситуації, слід перезавантажити модель з параметрами, які були до аварії. При повторній появі звернутися до служби підтримки.

ДОДАТОК В

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

АТ «УКРАЇНСЬКА ЗАЛІЗНИЦЯ»

ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЗАЛІЗНИЧНОГО ТРАНСПОРТУ ІМЕНІ АКАДЕМІКА В. ЛАЗАРЯНА

УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЗАЛІЗНИЧНОГО ТРАНСПОРТУ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФРАСТРУКТУРИ ТА ТЕХНОЛОГІЙ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ АВТОМОБІЛЬНО-ДОРОЖНІЙ УНІВЕРСИТЕТ

ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ ТА КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ

ЗБІРНИК ТЕЗ ДОПОВІДЕЙ
81 Всеукраїнської науково-технічної конференції
молодих учених, магістрантів та студентів
«НАУКА І СТАЛИЙ РОЗВИТОК ТРАНСПОРТУ»
28 жовтня 2021 року

INFORMATION-TELECOMMUNICATION TECHNOLOGY TA COMPUTER MODELING

CONFERENCE PROCEEDINGS
81th all Ukrainian Scientific and Technical Conference
of young scientists, masters and students
"SCIENCE AND SUSTAINABLE DEVELOPMENT OF TRANSPORT"
October 28, 2021

ДНІПРО
2021

ЗМІСТ

ДОСЛІДЖЕННЯ І МОДЕЛЮВАННЯ АВТОТРАНСПОРТНИХ ПОТОКІВ

Автор: Мерзлий О.Д.
Науковий керівник: к.т.н., доцент Горбова О.В.
Дніпровський національний університет залізничного
транспорту імені академіка В. Лазаряна

Через збільшення автотранспортних потоків в транспортній мережі актуальною є проблема їх раціональної організації. Однак з урахуванням впливу різних чинників, таких як завантаженість ділянки дороги, стану дороги, дана задача не може бути вирішена за допомогою аналітичних моделей, заснованих на графових моделях.

Метою вирішення поставленої задачі є розробка безпечної моделі дорожнього руху методом імітаційного моделювання координованих транспортних потоків в міський дорожньої мережі та розробка необхідної для досягнення поставленої мети системи комп'ютерного моделювання.

Моделювання є важливим засобом розв'язання багатьох економічних завдань і, зокрема, проведення аналітичного дослідження. Модель — це умовний об'єкт дослідження, тобто матеріальне чи образне відображення реального об'єкта, процесу його функціонування в конкретному середовищі. Метод моделювання — це конструювання моделі на основі попереднього вивчення об'єкта, визначення його найбільш суттєвих характеристик, експериментальний і теоретичний аналіз створеної моделі, а також необхідне коригування на підставі одержаної інформації.

Імітаційне моделювання дозволяє імітувати поведінку системи в часі. Причому перевагою є те, що часом в моделі можна управляти: уповільнювати у випадку з швидкоплинними процесами і прискорювати для моделювання систем з повільною несталістю. Імітаційне моделювання дає можливість розглянути поведінку тих об'єктів, реальні експерименти з якими є дуже витратними, неможливими або небезпечними. З популяризацією використання комп'ютерів, моделювання складних і унікальних виробів (процесів) супроводжується комп'ютерним тривимірним імітаційним моделюванням. Ця точна і відносно швидка технологія дозволяє накопичити всі необхідні знання, обладнання та напівфабрикати для майбутнього(ї) виробу (системи) до початку виробництва (впровадження у технологічний процес). Комп'ютерне 3D моделювання тепер не рідкість навіть для невеликих компаній.

Методика досліджень дозволили створити комплексний підхід до вирішення задач наведеного типу та містить симбіоз теоретичних та експериментальних досліджень.

Результати роботи покладені в основу системи імітаційного моделювання транспортних потоків, що дозволяє аналізувати властивості існуючих і проектованих транспортних вузлів. Система реалізована у вигляді програмного комплексу, який може бути використаний в установах державного управління, проектних організаціях і консалтингових компаніях, що займаються проектуванням і реорганізацією схем дорожнього руху. Запропонована модель агента може бути використана в складі більш складних імітаційних моделей організаційно-технічних систем.

В моделюванні моделі використана системна динаміка, дискретно-подієве моделювання (процесно-орієнтоване) та агентне моделювання.

Таким чином, розв’язок задачі полягає в створенні методу імітаційної моделі координованих транспортних потоків і реалізації її в вигляді як програмного модуля, так і реалізації моделі у вигляді спеціалізованого програмного забезпечення для розрахунку параметрів управління дорожнім рухом .

ДОДАТОК Г

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЗАЛІЗНИЧНОГО ТРАНСПОРТУ ІМЕНІ АКАДЕМІКА В. ЛАЗАРЯНА

АТ «УКРАЇНСЬКА ЗАЛІЗНИЦЯ»

ПАТ «КРЮКІВСЬКИЙ ВАГОНОБУДІВНИЙ ЗАВОД»

АТ «ДНІПРОПЕТРОВСЬКИЙ СТРІЛОЧНИЙ ЗАВОД»

ТОВ «ЗАВОД РЕЙКОВИХ СКРІПЛЕНЬ»

INSTYTUT KOLEJNICTWA

КОРПОРАЦІЯ «ДЕТАЛЬ ВАГОН ГРУП»

ПРОБЛЕМИ ТА ПЕРСПЕКТИВИ РОЗВИТКУ ЗАЛІЗНИЧНОГО ТРАНСПОРТУ

МАТЕРІАЛИ
81 МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ
22.04.2021–23.04.2021

Дніпро – 2021

ДОДАТОК Д

Міністерство освіти і науки України

Дніпровський національний університет
залізничного транспорту ім. акад. В. Лазаряна

Східний науковий центр транспортної академії наук

TEMPUS: CITISET & SEREIN & CRENG

ТЕЗИ
XIV Міжнародної науково-практичної конференції
«СУЧАСНІ ІНФОРМАЦІЙНІ ТА КОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ НА ТРАНСПОРТІ, В ПРОМИСЛОВОСТІ ТА ОСВІТІ»

ABSTRACTS
of the XIV International Conference
«MODERN INFORMATION AND COMMUNICATION TECHNOLOGIES ON A TRANSPORT, IN INDUSTRY AND EDUCATION»

ТЕЗИСЫ
XIV Международной научно-практической конференции
«СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ И КОММУНИКАЦИОННЫЕ ТЕХНОЛОГИИ НА ТРАНСПОРТЕ, В ПРОМЫШЛЕННОСТИ И ОБРАЗОВАНИИ»

15.12.2020 – 16.12.2020

Дніпро
2020

Дослідження і оптимізація автомобільних потоків засобами імітаційного моделювання

Горбова О.В., Мерзлий О.Д., Дніпровський національний університет залізничного транспорту імені академіка В.Лазаряна, Україна

Мета імітаційного моделювання полягає у відтворенні поведінки досліджуваної системи на основі результатів аналізу найбільш суттєвих взаємозв'язків між її елементами або іншими словами - розробці симулятора досліджуваної предметної області для проведення різних експериментів.

Імітаційне моделювання дозволяє імітувати поведінку системи в часі. Причому перевагою є те, що часом в моделі можна управляти: уповільнювати у випадку з швидкоплинними процесами і прискорювати для моделювання систем з повільною несталістю. Імітаційне моделювання дає можливість розглянути поведінку тих об'єктів, реальні експерименти з якими є дуже витратними, неможливими або небезпечними. З популяризацією використання комп'ютерів, виробництво складних і унікальних виробів супроводжується комп'ютерним тривимірним імітаційним моделюванням. Ця точна і відносно швидка технологія дозволяє накопичити всі необхідні знання, обладнання та напівфабрикати для майбутнього виробу до початку виробництва. Комп'ютерне 3D моделювання тепер не рідкість навіть для невеликих компаній.

Для пошуку ефективних стратегій управленій транспортними потоками в мегаполісі, оптимальних рішень з проектування вулично-дорожньої мережі та організації дорожнього руху необхідно враховувати широкий спектр характеристик транспортного потоку, закономірності впливу зовнішніх і внутрішніх факторів на динамічні характеристики змішаного транспортного потоку. Застосування моделювання і створення адекватної моделі транспортного потоку є актуальним завданням в процесі організації та управління дорожнім рухом.

Розробка імітаційної моделі транспортних потоків необхідна для отримання відповідної статистики, для подальшої оптимізації системи, та для обґрунтування щодо доцільності використання такої системи в реальному житті.

Метою вирішення поставленої задачі є розробка безпечної моделі дорожнього руху методом імітаційного моделювання координованих транспортних потоків в міський дорожньої мережі та розробка необхідної для досягнення поставленої мети системи комп'ютерного моделювання.

Математичний опис впливу різних характеристик середовища, в якому рухаються транспортні потоки, взаємний вплив транспортних потоків між собою в конфліктних ситуаціях, пошук характеристик і методики її обчислення, як показника стійкості параметрів управління координованого транспортного потоку до випадкових коливань інтенсивності є основними інструментами у досягненні поставленої задачі.

Методика досліджень дозволить створити комплексний підхід до вирішення задач наведеного типу та буде міститиме симбіоз теоретичних та експериментальних досліджень.

Для моделювання використовується системна динаміка, дискретно-подієвого моделювання (процесно-орієнтоване) та агентне моделювання.

Таким чином, розв'язок задачі полягає в створенні методу імітаційної моделі координованих транспортних потоків і реалізації її в вигляді як програмного модуля, так і реалізації моделі у вигляді спеціалізованого програмного забезпечення для розрахунку параметрів управління дорожнім рухом.

Подано до друку ДОДАТОК Е

Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2021, № 5 (95)

ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ ТА МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ

УДК 656.21.07

О. В. ГОРБОВА[1], О. МЕРЗЛИЙ[2]

[1]Кафедра «Комп'ютерні інформаційні технології», Дніпровський національний університет залізничного транспорту імені академіка В.Лазаряна, вул. Лазаряна, 2, Дніпро, Україна, 49010, тел. +38 (056) 373-15-35, ел. пошта alexandra.gorbova@gmail.com, ORCID - 0000-0002-5612-2715.

[2]Кафедра «Комп'ютерні інформаційні технології», Дніпровський національний університет залізничного транспорту імені академіка В.Лазаряна, вул. Лазаряна, 2, Дніпро, Україна, 49010, тел. +38 (056) 373-15-35, ел. пошта zver343434@gmail.com, 0000-0002-5219-5028.

ДОСЛІДЖЕННЯ АВТОМОБІЛЬНИХ ПОТОКІВ ЗАСОБАМИ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

Мета. Автомобільні перевезення стали дуже важливою частиною процесу перевезення і взаємодії з усіма видами транспорту, виконуючи перевезення вантажів і пасажирів. Через збільшення автотранспортних потоків в транспортній мережі актуальною є проблема їх раціональної організації. Однак з урахуванням впливу різних чинників, таких як завантаженість ділянки дороги, стану дороги, дана задача не може бути вирішена за допомогою аналітичних моделей, заснованих на графових моделях. Метою вирішення поставленої задачі є розробка безпечної моделі дорожнього руху методом імітаційного моделювання координованих транспортних потоків в міський дорожньої мережі та розробка необхідної для досягнення поставленої мети системи комп'ютерного моделювання. **Методика.** Для пошуку ефективних стратегій управленій транспортними потоками в мегаполісі, оптимальних рішень з проектування вулично-дорожньої мережі та організації дорожнього руху необхідно враховувати широкий спектр характеристик транспортного потоку, закономірності впливу зовнішніх і внутрішніх факторів на динамічні характеристики змішаного транспортного потоку. Застосування моделювання і створення адекватної моделі транспортного потоку є актуальним завданням в процесі організації та управління дорожнім рухом. Методика досліджень дозволить створити комплексний підхід до вирішення задач наведеного типу та буде містити симбіоз теоретичних та експериментальних досліджень. **Результати.** Під час проведення експериментів, було виявлено, що при звичайній роботі транспортної мережі, швидкість проїзду усього виду транспорту задовільний. При появі чинників, які створюють велике навантаження, збільшується час проїзду усього виду транспорту. Було виявлено і виділено такі фактори, які можуть вплинути на оптимальну роботу мережі навіть під великим навантаженням **Наукова новизна.** Полягає у створенні загальної методології при моделюванні процесів за допомогою методу поетапного моделювання та удосконаленні формалізації методу агентного моделювання. **Практична значимість.** Результати роботи покладені в основу системи імітаційного моделювання транспортних потоків, що дозволяє аналізувати властивості існуючих і проектованих транспортних вузлів. Система реалізована у вигляді програмного комплексу, який може бути використаний в установах державного управління, проектних організаціях і консалтингових компаніях, що займаються проектуванням і реорганізацією схем дорожнього руху. Запропонована модель агента може бути використана в складі більш складних імітаційних моделей організаційно-технічних систем.

Ключові слова: автотранспортний потік; імітаційне моделювання; графова модель; стратегія управлення транспортними потоками; дорожня мережа.

doi: 10.15802/stp2021/

Вступ

Україна має досить розвинуту мережу автодоріг загального користування протяжністю 172,4 тис. км, в тому числі 164,1 тис. км з твердим покриттям. За оцінками Міністерства інфраструктури, з числа міжнародних автотрас (близько 8200 км) в хорошому стані знаходиться 24%, в задовільному – ще 65%. Для доріг національного значення (4800 км) ці цифри складають 21% і 68% відповідно, а ось для регіональних (9800 км) - лише 10% і 50%. Очевидно, вітчизняні автомагістралі потребують серйозної модернізації[1].

Провівши аналіз, можна побачити швидкий темп зростання автотранспорту. На момент 1900 року в усьому світі нараховувалось близько 12 тисяч автомобілів. 1920 році – 10922 тисяч, 1950 року – 70388, 1957 року – 102827 тисяч. На даний час більше 1,5 млрд авто.

На момент 2014 року в усьому світі нараховувалось на кожних 1000 чоловік у світі припадало 118 легкових авто. У деяких країнах показник склав 2–10 автомобілів на 1000 жителів, але в деяких країнах цей показник перевищує 500 автомобілів на 1000 мешканців. Це нормальний показник для багатьох європейських країн. Рівень автомобілізації (250–300 автомобілів) дані країни досягли ще в 60–70-х роках минулого століття. Україна в порівнянні з іншими країнами має показник у чотири рази менший. Треба відмітити, що на 1000 мешканців в Україні припадає 191 автомобіль, а всього нараховується 8 млн. 700 тисяч автомобілів, що ставить Україну на 69 місце по рівню автомобілізації серед 145 країн.

Процес автомобілізації найбільших міст Західної Європи, що почався в 50-ті роки минулого сторіччя, проходив практично за однією закономірністю для усіх країн: лінійний ріст кількості автомобілів до рівня 300-350 авт./1000 жителів, потім уповільнення зростання та стабілізація при 550±50 авт./1000 жителів[2].

У цілому в розвинутих країнах проходить зниження темпів росту автомобільного парку, який в останні роки складає 1–2 %. У Північній Америці вже достатньо довго спостерігається ріст чисельності сімей, які мають два автомобілі і більше. У США їх доля вже досягла 58 %. У Західній Європі ріст чисельності середнього класу і його благополуччя супроводжується аналогічними процесами. Так, наприклад, в Ільде-Франс (агломерація Парижу) у 1991 році 75 % домогосподарств мали один автомобіль, а 20 % – два і більше. Такі тенденції спостерігалися і у Великобританії, наприклад, за період 1992–2000 роки відбулася зміна: кількість домогосподарств без автомобілів знизилась з 32 до 26 %; доля господарств з одним автомобілем залишилася сталою – 45 %, а доля домогосподарств з двома автомобілями і більше збільшилась із 24 % до 27 %; середня кількість автомобілів, яка приходиться на одне домогосподарство збільшилась з 0,86 до 1,04. Особливо цікавою для нас є динаміка зростання автомобільного парку в містах країн Східної Європи та прогнозовані показники, що приймаються фахівцями цих країн. Темпи зростання автомобільного парку в містах Східної Європи, звичайно, вищий, ніж у містах Західної Європи. Так, зростання парку Парижа становило 1 %, Варшави – 7 %, а зростання парку автомобільного транспорту для України в середньому складає 4,8 % , хоча у 2008 р. цей показник складав – 6,9 %(рис. 1.1). Слід зазначити, що у країн Східної Європи збільшення частки поїздок, що здійснюються з використанням легкового автомобіля, виявилося менше, ніж зростання автомобільного парку. Україна проходить той самий етап розвитку і формування автомобільного парку, який 30–50 років тому пройшли розвинуті країни світу[2].

Рис. 1.– Зростання автомобільного парку України в період 2000–2014 рр., у %

doi: 10.15802/stp2021/

В Україні інтенсивний ріст парку транспортних засобів розпочався в 1990-х роках і стабільно продовжується. Цей процес прямий і безпосередньо пов'язаний із набуттям економічної свободи громадянами України, свободи щодо вибору місця проживання і прикладення праці. Це в свою чергу призвело до швидкого росту автомобілізації, що підтверджує бажання українців до підвищення мобільності та якості життя. Автомобіль став не лише засобом переміщення, а й підтвердженням статусу життя, символом благополуччя і успішності в житті[2].

Але збільшення кількості транспорту у світі залежить не тільки від індивідуального транспорту. Через збільшення населення і потреб, модернізації транспорту і підвищення рівня життя, з'явилася потреба в комерційних перевезеннях.

Основні фактори, що впливають на ціну перевезення, тим саме на ціну самого товару – обсяг і вага вантажу (питома вартість перевезення великих партій завжди нижче, як і в будь-яких інших випадках з оптом). Деякі товари потребують особливих умов перевезення і зберігання. Чим довше товар буде в дорозі тим гірше для перевізника, і в купі ці всі фактори будуть впливати на кінцеву ціну товару.

Має значення і вид використовуваного авто, а також регіон: в великих логістичних та промислових вузлах перевезення для вантажовідправників традиційно дорожче. Це в першу чергу Київ, Дніпро, Одеса, Львів і Рівне.

Лише у 2020 році обсяг перевезення вантажів автомобільним транспортом склав близько 242,7 мільйонів тон. Так, як Україна територіально розташована майже у самому центрі автомобільних перевезень і основні дороги між державами проходять через Україну, доцільно організувати максимально швидкі шляхи для подолання відстані між точкою А і Б. А через збільшення транспорту, з'явилася дужа важлива проблема – це затори.

Чим більше місто і його населення, з тим більшою кількістю проблем йому доводиться зустрічатися щодня. А бажання кожного жителя мати особистий транспорт створює одну величезну проблему – трафік, з яким не кожні дороги можуть впоратися.

Відсутність достатньої кількості паркувальних місць, збільшення часу, проведеного в дорозі, висока ймовірність ДТП, зниження безпеки для пішоходів і, звичайно ж, пробки, які збільшують витрату палива і рівень забруднення навколишнього середовища.

Основні проблеми функціонування транспортних систем у містах:

– зростання рівня автомобілізації населення;

– збільшення інтенсивності використання індивідуально транспорту;

– зниження ефективності міського пасажирського транспорту;

– збільшення потреби жителів міста в переміщеннях;

– диспропорція між рівнем автомобілізації і темпами дорожнього будівництва;

– містобудівні проблеми розвитку міської території.

Особливо проблематичним є високі прямі і непрямі економічні витрати, в тому числі високі витрати, пов'язані з логістикою, аваріями, заторами та забрудненням повітря. Від цього страждають конкурентоспроможність України та її привабливість для інвесторів. Тому головним завданням є модернізація концепцій мобільності в українських містах.

Затори не тільки порушують наші плани. Вони – основне джерело забруднення повітря, що негативно відбивається на здоров'ї кожного. За даними Центру аналізу ризиків Гарвардського університету, пробки в 83 найбільших містах Сполучених Штатів стали причиною понад 2200 випадків передчасної смерті в 2010 році і збільшили витрати на практику охорони здоров'я на 18 млрд доларів.

Але ж є ще економічні витрати, пов'язані з втраченими в пробках годинами (робочими і неробочими) і з затримкою доставок. Водії в 10 американських містах з самим перевантаженим рухом щорічно проводять в пробках близько 42 годин, несучи збиток на більш ніж 121 млрд доларів.

У період автомобільного буму 1960-х містобудівники бачили тільки одне, здавалося б, очевидне рішення: будувати більше доріг і робити їх більш широкими. Але це не спрацювало. Чим більше доріг будувалося, тим більше

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)
Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2021, № 5 (95)

автомобілів на них з'являлося. Наприклад, дослідження, проведене в Каліфорнії в 1997 році, показало, що як би не збільшувалася пропускна здатність доріг, знадобиться всього п'ять років, щоб рівень їх завантаження досяг 90%.

Проблема заторів не вирішується шляхом будівництва нових доріг і тунелів. Необхідно визначити пріоритети. Це означає, що громадський транспорт повинен отримати пріоритет завдяки окремим рейсовими шляхами і виділеним автобусним смугах, а також на світлофорах.

Шляхів вирішення проблем заторів дуже багато, від соціальних, психологічних, до розробки моделей поведінки і розуміння основної причини появи заторів.

При першому аналізі проблеми, можливо зробити перші висновки, що є головною причиною заторів. Це, здавалося б, очевидно - вузькі місця на дорогах і скупчення автомобілів. Тим не менш, деякі пробки, на перший погляд виникають спонтанно, можуть бути викликані так званим ефектом «метелика», коли всього один водій раптом робить нічим не виправданий маневр - наприклад, різко перебудовується в інший ряд. Автомобілям, які їдуть за ним, доводиться різко гальмувати, що викликає ланцюгову реакцію, миттєво приводить до утворення пробки на трасі.

Дослідження, проведене Технологічному університеті штату Джорджія, бачить проблему в поєднанні агресивних водіїв (їдуть занадто швидко і занадто близько до машини попереду) і боязких водіїв, які дотримуються занадто велику дистанцію. І перші, і другі змушують інших водіїв гальмувати, в результаті чого рух сповільнюється ще більше, аж до повної зупинки.

Вчені намагалися змоделювати транспортні потоки, шукаючи аналогії в тому, як течуть рідина або газ, переміщаються птиці або лижники. Однак, вважає Габор Орос з Університету штату Мічиган, «хоча подібні аналогії допомагають зрозуміти деякі закономірності, стає все більш і більш очевидним, що транспортні потоки не схожі ні на які інші потоки в ньютонівському всесвіті».

На даний час покладаються великі надії на смарт технології і те, як саме вони можуть вирішити і вже вирішують проблеми завантаженого трафіку. Оскільки велика частина машин на дорозі – це ті, чиї водії шукають місця для паркування, в деяких містах намагаються управляти дорожнім рухом, використовуючи датчики для визначення того, зайнято конкретне місце на вулиці або на автостоянці або вільно.

Якщо зв'язати ці розумні датчики з системою, яка швидко і ефективно направляє водіїв на вільні паркувальні місця, є надія, що це знизить рівень завантаженості доріг. Першим такі датчики випробував Сан-Франциско. Не відстає від сусіда і Лос-Анджелес. Сьогодні обидва міста обслуговує ACS, дочірня компанія корпорації Хвтох.

Ще одним способом позбутися від пробок, як вважають деякі, може стати автоматизація самого процесу водіння. Нещодавно компанія Google представила свій власний прототип повністю автономного самоврядного автомобіля, якому не потрібен водій.

Мета

Метою вирішення поставленої задачі є розробка безпечної моделі дорожнього руху методом імітаційного моделювання координованих транспортних потоків в міський дорожньої мережі та розробка необхідної для досягнення поставленої мети системи комп'ютерного моделювання.

Методика

Вивчення сутності проблем транспортної Вивчення сутності проблем транспортної перевантаженості сучасних великих міст і мегаполісів спиралося на два методологічні принципи: принцип історизму та принцип системності. Згідно з принципом історизму основна увага в аналізі було приділено формуванню, розвитку і динаміці досліджуваних об'єктів. Аналіз спирається на принцип історизму як методологічний вираз саморозвитку дійсності, включаючи вивчення сучасного стану, минулого і процесів генезису. Принцип системності дозволяє розглядати міський транспорт як складну систему. Причому тут можливі різні підходи. Напри-

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)

Наука та прогрес транспорту. Вісник Дніпропетровської національної академії транспорту, 2021, № 5 (95)

клад, можна розглядати міський транспорт як складну систему з точки зору [3]:

1) інтенсивності руху по міських магістралях;

2) пропускної здатності перехресть;

3) оптимального регулювання і розподілу транспортних потоків за допомогою заборонних та обмежують знаків.

В цьому випадку елементами складної системи будуть:

1) міські магістралі та перехрестя;

2) засоби сигналізації та управління;

3) транспортні засоби.

Можливо вивчати міський пасажирський транспорт як складну систему з транспортних засобів (тролейбусів, автобусів, трамваїв, метрополітену, таксі та ін.), маршрутів руху, перехресть і світлофорів з урахуванням їх завантаження іншими видами транспорту, пасажиропотоками, що формуються в різних пунктах міста в залежності від часу доби, диспетчерських пунктів, засобів зв'язку і збору інформації, органів планування і управління, засобів ремонту і заправки автомобілів і т. д. При такому підході міський пасажирський транспорт як складна система розглядається з точки зору якості обслуговування пасажирів, планування маршрутів, розподілу рухомого складу на маршрутах і визначення оптимальних режимів руху (розкладів), планування поточного та капітального ремонту транспортних засобів [3]. Аналогічно як складну систему можна розглядати міський вантажний транспорт. Замість пасажиропотоків розглядаються вантажопотоки від постійних і тимчасових джерел, що вимагають доставки в постійні і тимчасові пункти призначення. При дослідженні такої складної системи виникають питання попереднього і оперативного планування перевезень, що забезпечує своєчасну і економічну доставку вантажів, а також проблеми, пов'язані з нормальною експлуатацією транспортних засобів [3]. Для проектування й розробка інструментального забезпечення для дослідження автотранспортних потоків використовується середовище AnyLogic. Це єдиний інструмент імітаційного моделювання (ІМ), який підтримує всі підходи до створення імітаційних моделей: процесно-орієнтований (дискретно-подієвий), системно динамічний і агентний, а також будь-яку їх комбінацію[4]. Унікальність, гнучкість і потужність мови моделювання AnyLogic дозволяє врахувати будь-який аспект моделювання з будь–яким рівнем деталізації. Графічний Інтерфейс AnyLogic, інструменти та бібліотеки дозволяють швидко створювати моделі для широкого спектру завдань від моделювання виробництва, логістики, бізнес-процесів до стратегічних моделей розвитку компанії і ринків. AnyLogic — це імітаційна платформа для повного бізнес-циклу [4].

Агент — елемент моделі, який може мати поведінку, пам'ять (історія), контакти тощо. Агенти можна моделювати у вигляді людей, компаній, проектів, автомобілей, міст, тварин, кораблів, товарів тощо.

Можна створювати всередині об'єкта змінні, діаграми станів, задавати події, потокові діаграми системної динаміки, а також додавати всередину агента об'єкти бібліотек AnyLogic. Можна створити в одній моделі стільки типів агентів, скільки агентів потрібно промоделювати[4].

Створення агента зазвичай починається з визначення його інтерфейсу для зв'язку із зовнішнім світом[4].

Початковий стан і поведінка агента може бути реалізована різними способами. Стан (накопичена історія) агента може бути представлена за допомогою змінної, або стану діаграми станів. Поведінка може бути або пасивною (агенти реагують тільки на прибуття повідомлень або на виклик методів і не мають власних подій, запланованих на майбутнє) або активною, коли внутрішня динаміка агента(події, заплановані через заданий таймаут або процеси системної динаміки) є причиною дій, що здійснюються агентом. В останньому випадку всередині агентів швидше за все повинні бути задані події і/або діаграми станів[5].

AnyLogic включає в себе графічну мову моделювання, а також дозволяє користувачеві розширювати створені моделі за допомогою мови Java. Інтеграція компілятора Java в AnyLogic надає більш широкі можливості при створенні моделей, а також створення Java апплетів, які можуть бути відкриті будь-яким браузером. Ці аплети дозволяють легко розміщува-

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)

Наука та прогрес транспорту. Вісник Дніпропетровської національної академії залізничного транспорту, 2021, № 5 (95)

ти моделі AnyLogic на веб-сайтах. На додаток до Java-аплетів, AnyLogic Professional підтримує створення Java-додатків, в цьому випадку користувач може запустити модель без інсталяції AnyLogic [6].

Графічне середовище моделювання AnyLogic включає в себе наступні елементи:

– Stock & Flow Diagrams (діаграма потоків і накопичувачів) застосовується при розробці моделей, використовуючи метод системної динаміки [6].

– Statecharts (карти станів) в основному використовуються в агентних моделях для визначення поведінки агентів. Вони також часто використовуються в дискретно-подієвому моделюванні, наприклад для симуляції машинних збоїв[6].

– Action charts (блок-схеми) використовуються для побудови алгоритмів. Застосовується в дискретно-подієвому моделюванні (маршрутизація дзвінків) і Агентно моделюванні (для логіки рішень агента) [6].

– Process flowcharts (процесні діаграми) - основна конструкція, яка використовується для визначення процесів в дискретно-подієвому моделюванні[6].

Середовище моделювання також включає в себе: низькорівневі конструкції моделювання (змінні, рівняння, параметри, події тощо), форми подання (лінії, квадрати, овали і т.д), елементи аналізу (бази даних, гістограми, графіки), стандартні картинки і форми експериментів[6].

При розробці моделі було використано елементи з таких бібліотек, як:

– бібліотека дорожнього руху (елементи бібліотеки показані на рис. 2);

– бібліотека моделювання процесів (елементи бібліотеки показані на рис. 3);

– пішохідна бібліотека(елементи бібліотеки показані на рис. 4).

Бібліотека дорожнього руху дозволяє детально імітувати фізичне переміщення машин по дорожній мережі. Крім того, вона дає можливість моделювати:

– рух з урахуванням правил дорожнього руху;

– світлофори і пріоритети проїзду на перехрестях;

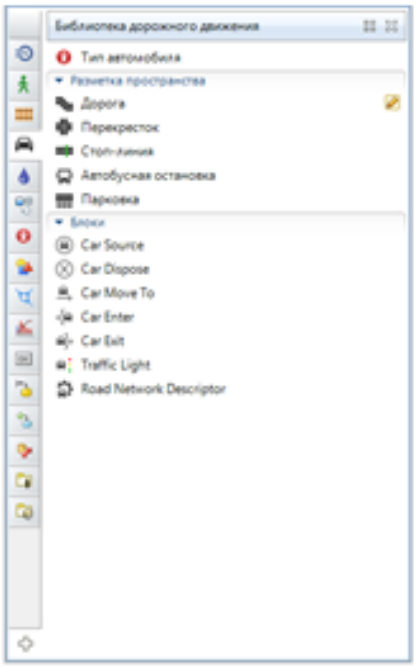


Рис. 2 – Дорожня бібліотека

– місця для паркування;

– рух і зупинки громадського транспорту.

Бібліотека моделювання процесів AnyLogic підтримує дискретно-подієвий, або, якщо бути більш точним, «процесний» підхід моделювання. За допомогою об'єктів бібліотеки моделювання процесів можна моделювати системи реального світу, динаміка яких представляється як послідовність операцій (прибуття, затримка, захоплення ресурсу, поділ) над агентами, що представляють клієнтів, документи, дзвінки, пакети даних, транспортні засоби, тощо. Ці агенти можуть володіти певними атрибутами, що впливають на процес їх обробки (наприклад, тип дзвінка, складність роботи) або накопичують статистику (загальний час очікування, вартість).

Процес задається у формі потокових діаграм (блок-схем) — графічне представлення, прийняті в багатьох областях: виробництво, бізнес-процеси, центри обробки дзвінків, логістика, охорони здоров'я, тощо[7].

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)

Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2021, № 5 (95)

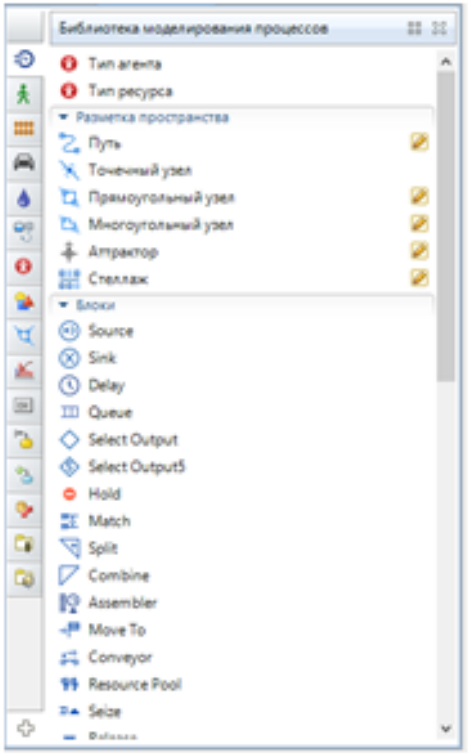


Рис. 3 – Бібліотека моделювання процесів

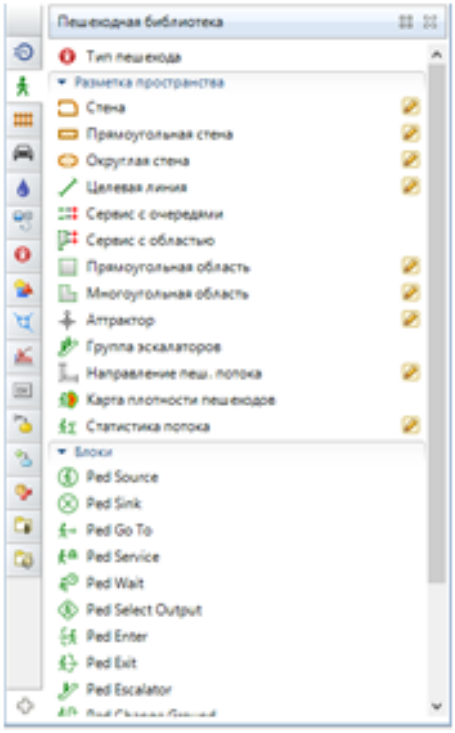


Рис. 4 – Пішохідна бібліотека

Пішохідна бібліотека AnyLogic є високорівневою бібліотекою моделювання руху пішоходів у фізичному просторі. Вона дозволяє моделювати будівлі, в яких рухаються пішоходи (станції метро, стадіони, музеї), а також вулиці і інші місця великого скупчення людей. З нею можна збирати статистику [7].

Моделі руху пішоходів складаються з двох складових — середовища і поведінки. Під середовищем мається на увазі об'єкти фізичного середовища — стіни, різні області, сервіси, черги, тощо.

Основний об'єкт бібліотеки є пішохід. Пішохід задається за допомогою об'єкта типу Ped. Пішохід «живе» в заданому фізичному просторі (середовищі) і пересувається згідно із заданими правилами. З іншого боку, тип пішохода успадкований від типу агента Agent, тому пішоходи пересуваються по блок-схемі так само, як агенти.

Блок-схема пішохідної моделі будується за допомогою об'єктів, що міститься в пішохідній бібліотеці. Тип агента Ped є базовим типом для моделювання пішоходів. В бібліотеці є об'єкти для створення пішоходів і управління потоком пішоходів[7]. Правила завдання потоку пішоходів аналогічні правилам завдання потоку агентів в бібліотеці моделювання процесів.

Опис агентів. Main — головний агент моделі, який об'єднує всі інші агенти. Цей агент описує повну роботу всієї моделі

Bus113A — агент, який описує властивості автобуса, що перевозить людей по системі. Він має свій параметр «ВремяПоявления113», який фіксує час появи агента в системі. Агент має свою модель з 2D/3D текстурою. Цей агент показаний на рис. 5.

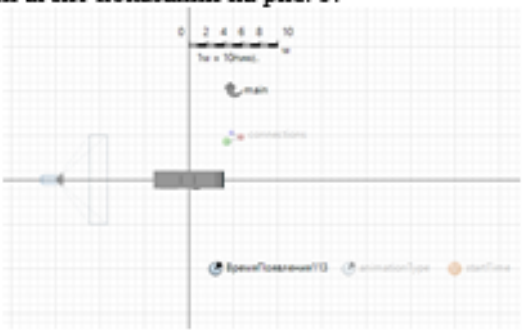


Рис. 5 – Модель агента Bus113A

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)
Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2021, № 5 (95)

Bus146A— агент, який описує властивості автобуса, що перевозить людей по системі. Він схожий с агентом Bus113A і має такий же параметр — «ВремяПоявления146».

Car — агент, який описує властивості авто, що рухаються по системі. Цей агент схожий з агентами Bus146A та Bus113A. Має властивість «ВремяПоявления». Цей агент має п'ять текстур різного кольору. Агент представлений на рис. 6.

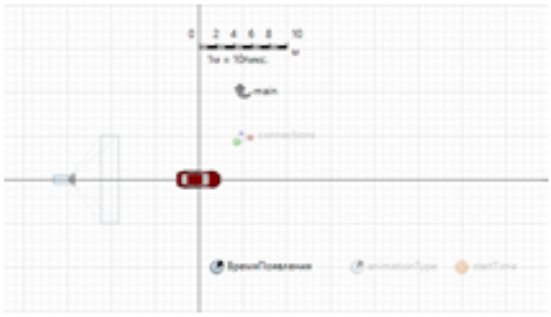


Рис. 6 – Модель агента Car

Vehicle — це агент без текстури, який слугує для того, щоб доповнювати агенти Car, Bus146A та Bus113A та розширювати їх функціонал. Цей агент має камеру camera, параметр animationType та змінну startTime.

Пешеход — агент, який описує властивості пішохода. Він схожий з агентом Car крім того, що він не доповнюється агентом Vehicle і має свою текстуру.

В імітаційній моделі використовуються такі елементи:

1 Елемент «Дорога »є графічним елементом розмітки простору, це безперервна дорога (тобто такої дороги, яка не містять перехрестя). Використовуючи дороги, перехрестя і інші елементи розмітки простору, можна створювати дорожні мережі для моделей бібліотеки дорожнього руху.

2 За допомогою спеціального елемента« Масштаб» можна задати масштаб анімації моделі, тобто, відношення розміру об'єкта на анімації моделі до натурального розміру об'єкта, що моделюється. Фактично, масштаб задається як співвідношення пікселів анімації і фізичної одиниці довжини.

3 За допомогою елемента розмітки простору «перехрестя» можна поєднувати дві або більше дороги. Використовуючи дороги, перехрестя і інші елементи розмітки простору, є можливість створювати дорожні мережі для моделей бібліотеки дорожнього руху. Відображені на перехресті білі точкові лінії - з'єднувачі смуг - задають дозволені напрямки руху транспорту на перехресті (рис. 7).

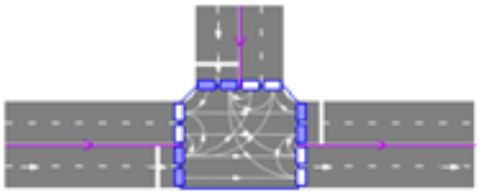


Рис. 7 – Перехрестя

4 За допомогою елемента розмітки простору «автобусна зупинка» можна намалювати автобусну зупинку на узбіччі дороги у напрямку руху(рис. 8).

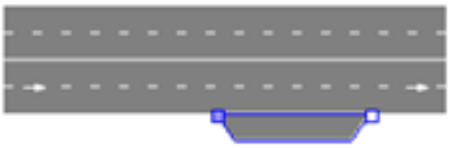


Рис. 8 – Автобусна зупинка

5 За допомогою елемента розмітки простору «парковка» можна малювати однорядні паркувальні місця, розташовані на узбіччях дороги.

Парковка може бути паралельна (машини паркуються в одну лінію з іншими припаркованими машинами), або перпендикулярна — це задається у властивості «тип парковки» (рис. 9).

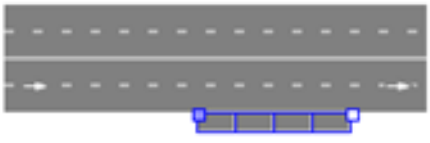


Рис. 9 – Парковка

6 CarSourse — створює автомобілі і намагається помістити їх в зазначеному місці дорожньої мережі (на обраній дорозі або парковці).

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)

Наука та прогрес транспорту. Вісник Дніпропетровської національної академії транспорту, 2021, № 5 (95)

7 CarDispose — видаляє машини з моделі.

8 CarMoveTo — блок, який керує рухом автомобіля. Автомобіль може їхати, тільки коли він знаходиться в блоці CarMoveTo. Автомобіль намагається розрахувати шлях від свого поточного місця до зазначеного місця призначення, коли надходить до блоку CarMoveTo. В якості місця призначення можуть виступати: дорога, парковка, автобусна зупинка або стоп-лінія.

9 TrafficLight — моделює світлофор, керуючий рухом машин на перехресті або на будь-якій стоп-лінії.

10 RoadNetworkDescriptor — опціональний блок. За допомогою блоку RoadNetworkDescriptor розробники отримують доступ до управління всіма транспортними засобами, що перебувають в одній дорожньої мережі. Блок дозволяє задавати дії, які будуть виконуватися при додаванні автомобіля в дорожню мережу, в'їзді на дорогу, зупинки автомобіля, зміні смуги, тощо. Крім того, за допомогою даного блоку можна включити відображення пробок на дорогах

11 PedSource — створює пішоходів. Зазвичай використовується в якості початкової точки діаграми пішохідного процесу.

12 PedSink — видаляє пішоходів. Зазвичай використовується в якості кінцевої точки діаграми пішохідного процесу.

13 PedGoTo — змушує пішоходів йти до заданої точки простору.

14 PedEnter — поміщає вже створених раніше пішоходів в модельовану середу.

15 PedExit — видаляє пішоходів з моделюючого середовища.

16 Delay — затримує агентів на заданий період часу.

17 Queue — зберігає агентів в певному порядку. Моделює чергу агентів, які чекають прийом об'єктами, заданими в потоковій діаграмі.

18 SelectOutput — направляє агентів в один з двох вихідних портів в залежності від виконання заданої умови.

19 SelectOutput5 — направляє агентів в один з п'яти вихідних портів в залежності від виконання заданих умов.

20 Pickup — додає агентів до вмісту агента-контейнера.

21 DropOff — видаляє обраних агентів з агента-контейнера і посилає їх далі.

22 3D Вікно — це елемент, що задає на діаграмі агента область, в якій під час запуску оделі буде відображатися тривимірна анімація цього об'єкта.

23 Область перегляду — За допомогою цього елемента можна виділити на діаграмі типу агента, деякі області, що містять логічно відокремлені групи елементів або ділянки діаграми. Задавши такі області, можна легко перемикатися між ними під час виконання моделі за допомогою спеціальних засобів навігації. Це дозволить швидко переходити до тієї або іншої ділянки діаграми агента.

24 Гістограма — це елемент, що відображає дані, зібрані об'єктом «дані гістограми» (на одній гістограмі можуть одночасно відображатися дані відразу декількох таких об'єктів). Ось X завжди масштабується таким чином, щоб вмістити всі дані. Масштаб по осі Y також вибирається автоматично, таким чином, щоб висота найвищої стовпця дорівнювала висоті області діаграми.

Початок руху авто починається в одному із блоків CarSourse (рис. 10).

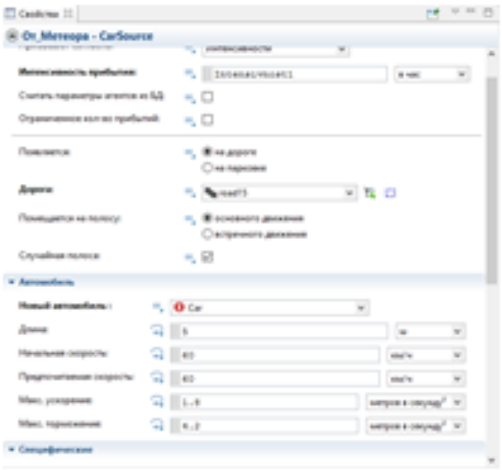


Рис. 10 – Властивості блоку CarSourse

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)
Наука та прогрес транспорту. Вісник Дніпропетровської національної академії транспорту, 2021, № 5 (95)

В кожному з блоків CarSource можна вказати початкову дорогу або парковку, на якій будуть з'являтися автомобілі. Обрати тип автомобіля (агента, який відповідає за автомобіль), вказати початкові дані транспорту (початкову швидкість, середню швидкість, тощо.), задати інтенсивність потоку.

Після цього потік потрапляє в блок selectOutput5, який розподіляє, куди буде рухатися те чи інше авто. В блоці selectOutput5 є п'ять виходів із своїми ймовірностями. Кожен із виходів має свою вірогідність використання, це показано на рис. 10

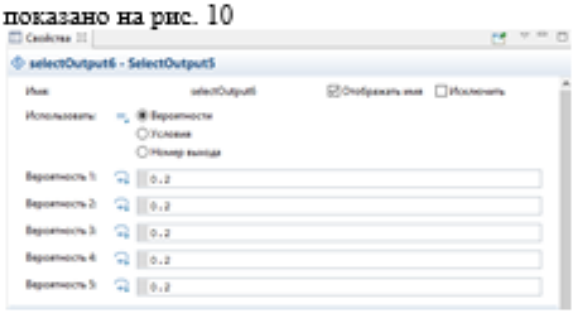


Рис. 10 – Властивості блоку selectOutput5

Після обрання шляху авто потрапляє в блок selectOutput, який по властивостям схожий з блоком selectOutput5, але має лише 2 ймовірності з виходами:

1 По першій ймовірності, авто потрапляє в блок CarMoveTo, у властивостях якого вказано, до якої кінцевої дороги потрібно їхати автомобілю. Після приїзду до кінця обраної дороги, авто потрапляє до блоку carDispose, який видаляє авто з системи.

2 По другій ймовірності, авто потрапляє до блоку selectOutput5. Після цього блоку авто потрапляє в блок CarMoveTo, в якому вказано місце знаходження парковки. Якщо на парковці не має місця, то авто їде далі по логіці. Якщо місце вільне то авто паркується і потрапляє в блок Delay, в якому вказано скільки автомобілю потрібно очікувати часу. Після потрібного очікування авто їде далі по логіці моделі

Після пункту 2, авто знов потрапляє в блок вибору шляху, де по ймовірності обирається блок CarMoveTo. Авто їде до потрібної дороги, після чого видаляється з системи.

Логіка проїзду автомобілів в системі показано на рис. 11.

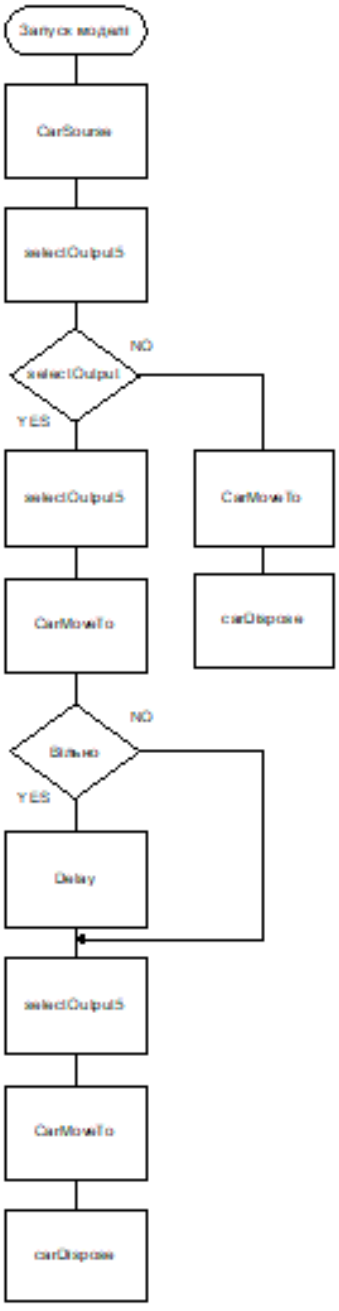


Рис. 11 – Схема логіки автомобілів

Після запуску моделі всі автомобілі рухаються за правилами дорожнього руху.

Такі блоки потрібні для того, щоб у процесі експерименту була можливість змінювати параметри (наприклад параметр появи кількості автомобілів у системі).

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)
Наука та прогрес транспорту. Вісник Дніпропетровської національної академії транспорту, 2021, № 5 (95)

У системі присутні елементи Traffic Light, які вже були описані. Налаштування таких блоків було повністю взято з реального життя для чистоти експерименту. Приклад налаштування показано на рис. 12.

Рис. 12 – властивості блоку Traffic Light

Основна карта моделі і логіки моделі показано на рис. 13-14.

Рис. 13 – Дорожня карта моделі

Результати

Експеримент повинен показати, які фактори більше за все впливають на появи пробок і збільшення навантаження на транспортну мережу. Для більш точних показників, кожен експеримент буде проводитись 5 разів. Час тестування – 1 година.

Експеримент 1. Провести експеримент у звичному режимі роботи транспортної мережі, без чинників, які впливають на систему.

Основні вхідні данні:

Таблиця 1

Кількість авто у системі

Напрямок	Кількість появи легкових авто
От Метеора От Макарова	200
От Кирова От Дафи	300
От Суворова	30
От Янгеля	70

Кількість пішоходів у системі – 300 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	210,09	364,31	449,94	601,10	50,79	1043	81,79
2	206,42	357,69	496,32	558,75	45,20	1012	71,88
3	204,65	428,41	497,30	597,55	48,99	977	75,44
4	220,92	354,39	467,93	674,08	55,88	976	73,19
5	201,57	414,41	458,30	652,60	54,79	1010	79,80
Середнє значення	208,73	383,84	473,96	616,82	51,13	1003,00	76,42

Рис. 14 – Результати експериментів

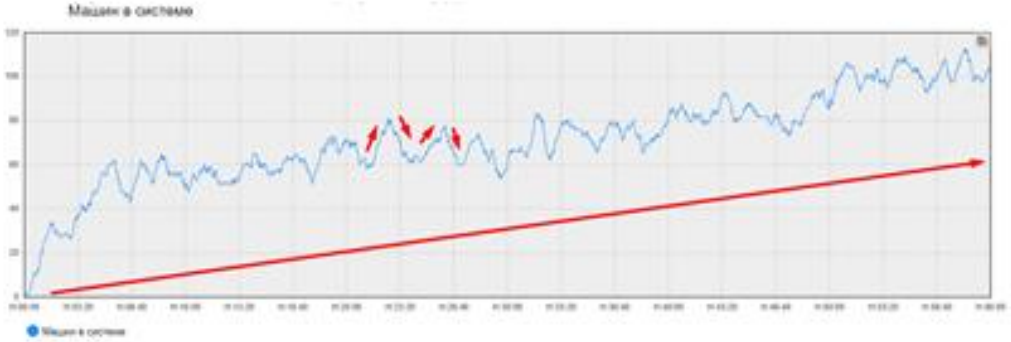


Рис. 15 – Динаміка проїзду авто

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)

Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2021, № 5 (95)

A	B	C	D	E	F	G	H
Експеримент №	Час проїзду авто	Час проїзду 146 автобуса	Час проїзду 113 автобуса	Час знаходження пішоходів у системі	Середня постійна кількість пішоходів	Кількість авто у системі	Середня постійна кількість авто
1	393,16	630,12	624,25	869,98	42,95	991	148,34
2	391,12	611,41	666,03	909,63	65,56	1228	143,27
3	440,48	668,03	681,88	912,61	64,99	1199	145,18
4	441,55	663,61	713,30	903,59	73,78	1194	154,58
5	520,41	867,42	778,30	1009,15	86,39	1035	148,84
Середнє значення:	437,34	688,52	692,75	921,02	66,734	1129,40	148,04

Рис. 16 – Результати експериментів

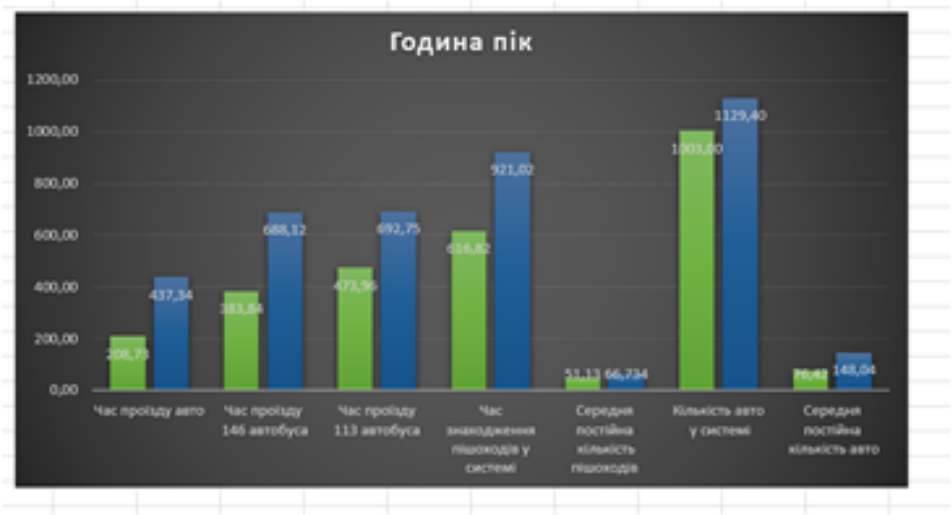


Рис. 17 – Порівняння експериментів

Маємо статистику роботи моделі, для порівняння з іншими експериментами.

По динаміці проїзду авто можна побачити, що модель працює стабільно, без просадок, стрімкого росту навантаження. Результати одного з експериментів показано на рис. 23.

Висновок: при розборі графіку проїзду можна побачити, що є зростання і падіння тренду, що є коректним. Основне зростання зумовлено тим, що у системі є місця для паркування авто. Такі авто теж рахуються, як «Авто у системі».

Ці результати будемо вважати за результати правильної роботи моделі.

Експеримент 2. Збільшити кількість авто по кожному основному напрямку на 100 авто, імітуючи «Година пік».

Кількість пішоходів у системі – 300 чоловік на годину.

Інтервал появи автобусів – кожні 10 хвилин.

Загальна кількість місць в автобусі – 45.

Таблиця 2

Кількість авто у системі

Напрямок	Кількість появи легкових авто
От. Метеора От. Макарова	300
От. Кирова От. Дафи	400
От. Суворова	30
От. Янгеля.	70

Згідно з рис. 17 можна побачити стрімкий ріст часу кожного параметру, у деяких ріст понад 45%.

Висновок: зі збільшенням кількості автомобілів у системі, збільшується і час проходження системи транспортом. Збільшується час очікування і проходження системи для пішоходів так, як автобуси стоять у заторах і не можуть вчасно приїхати до зупинки.

Під час проведення експериментів, було виявлено, що при звичайній роботі транспортної мережі, швидкість проїзду усього виду тра-

doi: 10.15802/stp2021/

нспорту задовільний. При появі чинників, які створюють велике навантаження, збільшується час проїзду усього виду транспорту. Було виявлено і виділено такі фактори, які можуть вплинути на оптимальну роботу мережі навіть під великим навантаженням, а саме:

1. Для уникнення великого скупчення людей на зупинках:

– У час, коли трафік починає збільшуватися (у ранці і у ввечері) зменшити інтервал між автобусами. Тим саме час очікування на зупинках зменшиться, а бюджет автопарку збільшиться.

– Збільшити кількість місць у автобусі (оновлення автопарку).

– Надавати можливість пішоходам використовувати альтернативний транспорт, наприклад електросамокати.

– Організація окремих полос для руху міського транспорту.

– Побудова метро.

2. Для уникнення пробок на дорогах:

– У час, коли трафік починає збільшуватися (у ранці і у ввечері) змінювати фази перемикання світлофорів таким чином, щоб навантаженні ділянки дороги проїжджалися швидше.

– Заохочувати автомобілістів пересаджуватися на міський транспорт у годину пік.

– Розширювати основні навантажені дороги/перехрестя.

Наукова новизна та практична значимість

Наукова новизна роботи полягає у створенні загальної методології при моделюванні процесів за допомогою методу поетапного моделювання та удосконаленні формалізації методу агентного моделювання;.

Всі результати, отримані в роботі, є новими і актуальними. Зокрема, запропонована модель агента-учасника дорожнього руху, що відображає основні аспекти поведінки водіїв. Також виконана програмна реалізація моделі агента, придатна для дослідження транспортних систем. Усі дослідження виконуються на основі реальної транспортної системи.

Висновки

У цій статті запропоновано методику вирішення проблем заторів засобами імітаційного моделювання. Було побудовано модель на основі реальної транспортної мережі. Проведено ряд експериментів з імітацією різних проблем, які можуть з'явитися на дорозі. Було зібрано статистику і оптимізовано рух.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Міністерство інфраструктури України. URL: https://mtu.gov.ua/timeline/Dorozhne-gospodarstvo.html (дата звернення 10.09.2021).
2. Дослідження транспортних потоків в аспекті заторових станів дорожнього руху : Монографія / В.М. Перцаков та ін. Київ : УДК, 2015. 32с.
3. Агентне моделювання в AnyLogic. URL: http://eki.tneu.edu.ua (дата звернення 18.09.2021).
4. AnyLogic. URL: https://www.anylogic.ru/ (дата звернення 01.09.2021).
5. Агентне моделювання в AnyLogic. URL: http://eki.tneu.edu.ua (дата звернення 03.09.2021).
6. Вільна енциклопедія Wikipedia. URL: https://uk.wikipedia.org (дата звернення 09.09.2021).
7. Імітаційне моделювання. . URL: http://ni.biz.ua/5/5_3/5_32577_imitatsionnoe-modelirovanie.html (дата звернення 19.09.2021).
8. Л.Ф. Вьюненко, Імітаційне моделювання: Посібник / Л.Ф. Вьюненко, Москва : 2016. 266с.
9. Система Actor Pilgrim URL: http://simulation.su/static/actor-pilgrim-full-info.html (дата звернення 29.09.2021).
10. Імітаційне моделювання. URL: http://ni.biz.ua/5/5_3/5_32577_imitatsionnoe modelirovanie.html (дата звернення 19.09.2021).
11. Дослідження транспортних потоків в аспекті заторових станів дорожнього руху : Монографія / В.М. Перцаков та ін. Київ : 2015. 32с.

doi: 10.15802/stp2021/

12. Як Ford використовує імітаційне моделювання при розробці безпілотних автомобілів. URL: https://nfp2b.ru/2020/04/02/kak-ford-ispolzuet-imitatsionnoe-modelirovanie-pri-razrabotke-bespilotnyh-avtomobilejj/ (дата звернення 29.09.2021).
13. Моделювання систем. Інструментальні засоби GPSS World: Навч. Посібник / В.Д. Боев : СПБ, 2019. 346с.
14. Аналітико-імітаційні дослідження систем та мереж масового обслуговування : Монографія / В.М. Задорожний : Омск, 2010. 280с.
15. Formation of recommendations for the selection of types of connections for different types: Monograph / A. Novikov : Krasnoyarsk, 2019. 10p.
16. Methodology for TRITIA transport model : Report / Transport Research Institute, JSC : 2019. 43p.

А. В. ГОРБОВА[1*], А.Д. МЕРЗЛИЙ

[1*]Каф. «Компьютерные информационные технологии», Днепровский национальный университет железнодорожного транспорта имени академика В. Лазаряна, ул. Лазаряна, 2, Днипро, Украина, 49010, тел. +38 (056) 373 15 35, эл. почта alexandra.gorbova@gmail.com, ORCID 0000-0002-5612-2715

[1*]Каф. «Компьютерные информационные технологии», Днепровский национальный университет железнодорожного транспорта имени академика В. Лазаряна, ул. Лазаряна, 2, Днипро, Украина, 49010, тел. +38 (056) 373 15 35, эл. почта zver343434@gmail.com, 0000-0002-5219-5028.

ИССЛЕДОВАНИЕ АВТОМОБИЛЬНЫХ ПОТОКОВ СРЕДСТВАМИ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Цель. Автомобильные перевозки стали очень важной частью процесса перевозки и взаимодействия со всеми видами транспорта, выполняя перевозку грузов и пассажиров. Из-за увеличения автотранспортных потоков в транспортной сети актуальна проблема их рациональной организации. Однако с учетом влияния различных факторов, таких как загруженность участка дороги, состояния дороги, данная задача не может быть решена с помощью аналитических моделей, основанных на графовых моделях. Целью решения поставленной задачи является разработка безопасной модели дорожного движения методом имитационного моделирования координированных транспортных потоков в городской дорожной сети и разработка необходимой для достижения поставленной цели системы компьютерного моделирования. **Методика**. Для поиска эффективных стратегий управляемой транспортными потоками в мегаполисе, оптимальных решений по проектированию улично-дорожной сети и организации дорожного движения необходимо учитывать широкий спектр характеристик транспортного потока, закономерности влияния внешних и внутренних факторов на динамические характеристики смешанного транспортного потока. Применение моделирования и создание адекватной модели транспортного потока является актуальной задачей в процессе организации и управления дорожным движением. Методика исследований позволит создать комплексный подход к решению задач приведенного типа и будет содержать симбиоз теоретических и экспериментальных исследований. **Результаты.** Во время проведения экспериментов было обнаружено, что при обычной работе транспортной сети скорость проезда всего вида транспорта удовлетворительна. При появлении факторов, создающих большую нагрузку, увеличивается время проезда всего вида транспорта. Были обнаружены и выделены такие факторы, которые могут повлиять на оптимальную работу сети даже под большой нагрузкой. **Научная новизна**. Состоит в создании общей методологии при моделировании процессов с помощью метода поэтапного моделирования и усовершенствовании формализации метода агентного моделирования. **Практическая значимость**. Результаты работы положены в основу системы имитационного моделирования транспортных потоков, что позволяет анализировать свойства существующих и проектируемых транспортных узлов. Система реализована в виде программного комплекса, который может быть использован в учреждениях государственного управления, проектных организациях и консалтинговых компаниях, занимающихся проектированием и реорганизацией схем дорожного движения. Предлагаемая модель агента может быть использована в составе более сложных имитационных моделей организационно-технических систем.

Ключевые слова: автотранспортный поток; имитационное моделирование; графовая модель; стратегия управления транспортными потоками; дорожная сеть

doi: 10.15802/stp2021/

ISSN 2307–3489 (Print), ISSN 2307–6666 (Online)

Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2021, № 5 (95)

A. V. GORBOVA[1*], A. D. MERZLIY[2*]

[1*]Dep. «Computer Information Technologies», Dnipro National University of Railway Transport named after Academician V. Lazaryan, Lazaryana St., 2, Dnipro, Ukraine, 49010, tel. +38 (056) 373 15 35, e-mail alexandra.gorbova@gmail.com, ORCID 0000-0002-5612-2715

[2*]Dep. «Computer Information Technologies», Dnipro National University of Railway Transport named after Academician V. Lazaryan, Lazaryana St., 2, Dnipro, Ukraine, 49010, tel. +38 (056) 373 15 35, e-mail zver343434@gmail.com, 0000-0002-5219-5028.

RESEARCH OF AUTOMOBILE FLOWS BY IMITATION SIMULATION

Purpose. Road transportation has become a very important part of the transportation process and interaction with all modes of transport, carrying out the transportation of goods and passengers. Due to the increase in traffic flows in the transport network, the problem of their rational organization is urgent. However, taking into account the influence of various factors, such as the congestion of the road section, the condition of the road, this problem cannot be solved using analytical models based on graph models. The purpose of solving the problem is to develop a safe traffic model by the method of simulation of coordinated traffic flows in the urban road network and to develop a computer simulation system necessary to achieve this goal. **Methodology**. To search for effective strategies for controlled traffic flows in a megalopolis, optimal solutions for the design of the road network and traffic management, it is necessary to take into account a wide range of traffic flow characteristics, the regularities of the influence of external and internal factors on the dynamic characteristics of a mixed traffic flow. The use of modeling and the creation of an adequate model of traffic flow is an urgent task in the process of organizing and managing traffic. The research methodology will make it possible to create an integrated approach to solving problems of the above type and will contain a symbiosis of theoretical and experimental research. **Findings**. During the experiments, it was found that with the normal operation of the transport network, the travel speed of the entire mode of transport is satisfactory. With the appearance of factors creating a large load, the travel time of the entire mode of transport increases. We have identified and identified such factors that can affect the optimal operation of the network, even under heavy load. Scientific novelty. Consists in the creation of a general methodology for modeling processes using the step-by-step modeling method and improving the formalization of the agent-based modeling method. **Originality**. Consists in creating a general methodology for modeling processes using the step-by-step modeling method and improving the formalization of the agent-based modeling method. **Practical value**. The results of the work are used as the basis for a system of simulation modeling of traffic flows, which makes it possible to analyze the properties of existing and projected transport hubs. The system is implemented in the form of a software package that can be used in public administration institutions, design organizations and consulting companies involved in the design and reorganization of traffic patterns. The proposed agent model can be used as part of more complex simulation models of organizational and technical systems.

Key words: traffic flow; simulation modeling; graph model; traffic management strategy; road network

REFERENCES

1. Ministry of Infrastructure of Ukraine. URL: https://mtu.gov.ua/timeline/Dorozhne-gospodarstvo.html (access date 10.09.2021).
2. Research of transport flows in the aspect of traffic jams: Monograph / V.M. Pershakov and others. Kyiv: UDC, 2015. 32p.
3. Agent modeling in AnyLogic. URL: http://eki.tneu.edu.ua (access date 18.09.2021).
4. AnyLogic. URL: https://www.anylogic.ru/ (access date 01.09.2021).
5. Agent modeling in AnyLogic. URL: http://eki.tneu.edu.ua (access date 03.09.2021).
6. Free encyclopedia Wikipedia. URL: https://uk.wikipedia.org (accessed 09.09.2021).
7. Simulation modeling. URL: http://ni.biz.ua/5/5_3/5_32577_imitatsionnoe-modelirovanie.html (access date 19.09.2021).
8. L.F. Vyunenko, Imitation modeling: Manual / L.F. Vyunenko. Moscow: 2016. 266p.
9. Actor Pilgrim system URL: http://simulation.su/static/actor-pilgrim-full-info.html (accessed 29.09.2021).

doi: 10.15802/stp2021/

10. Simulation modeling. URL: http://ni.biz.ua/5/5_3/5_32577_imitatsionnoe modelirovanie.html (access date 19.09.2021).
11. Research of transport flows in the aspect of traffic jams: Monograph / V.M. Pershakov and others. Kyiv: 2015. 32p.
12. How Ford uses simulation in the development of unmanned vehicles. URL: https://nfp2b.ru/2020/04/02/kak-ford-ispolzuet-imitatsionnoe-modelirovanie-pri-razrabotke-bespilotnyh-avtomobilejj/ (access date 29.09.2021).
13. Modeling of systems. GPSS World Tools: Textbook. Manual / VD Boev; SPB, 2019. 346p.
14. Analytical and simulation studies of systems and networks of queuing: Monograph / V.M. Zado-rozhny; Omsk, 2010. 280p.
15. Formation of recommendations for the selection of types of connections for different types: Monograph / A. Novikov: Krasnoyarsk, 2019. 10p.
16. Methodology for TRITIA transport model: Report / Transport Research Institute, JSC: 2019. 43p.

doi: 10.15802/stp2021/

Довідка
про відсутність плагіату у випускній кваліфікаційній роботі

Міністерство освіти і науки України
Український державний університет науки та технологій

Кафедра «Комп'ютерні інформаційні технології»

ДОВІДКА

За результатами перевірки випускної кваліфікаційної роботи здобувача вищої освіти *Мерзлого Олексія Дмитровича*
(прізвище, ім'я, по батькові)
на тему: *Дослідження та оптимізація автомобільних транспортних потоків засобами імітаційного моделювання*
в роботі не виявлено порушень академічної доброчесності.

Керівник ВКР _____ Олександра ГОРБОВА

РЕЦЕНЗІЯ

на дипломну роботу студента *ПЗ2021* групи факультету КТС

Мерзлого Олексія Дмитровича
(прізвище, ім'я по батькові)

1. Тема дипломної роботи *Дослідження та оптимізація автомобільних транспортних потоків засобами імітаційного моделювання*

2. Обсяг поданого на рецензію матеріалу *пояснювальна записка - 86 сторінок, технічна документація - 94 сторінок*
(пояснювальна записка, технічна документація)

3. Відповідність роботи завданню *цілком відповідає*

4. Якість поданого матеріалу: *Пояснювальна записка є технічно грамотною, всі розділи викладені повно та завершено, мають розгорнуті висновки.*

5. Обґрунтованість, доцільність застосованих рішень *всі отримані рішення є доцільними в межах задачі, що розв'язується, та є раціонально використаними при розробці імітаційної моделі автомобільних транспортних потоків*

6. Оцінка рівня підготовки дипломника *було встановлено під час особистої бесіди та прочитаної пояснювальної записки*

7. Зауваження до роботи *деталі створення імітаційної моделі та логіку її роботи можна було описати більш детально.*

8. Оцінка, що рекомендується, *відмінно*

9. Інші загальні зауваження й побажання щодо якості підготовки фахівців *відсутні*

Рецензент *Казимир Главацький, к.т.н., доцент кафедри „Прикладна механіка та матеріалознавство" ЖДТУ.*

«17» грудня 2021 р. _____

ВІДГУК

Основного керівника *к.т.н., доц. кафедри КІТ*
Горбової Олександри Вікторівни
(посада, вчене звання, прізвище, ім'я, по батькові)
на дипломну роботу студента *ПЗ2021* групи факультету
Мерзлого Олексія Дмитровича
(прізвище, ім'я по батькові)

Тема дипломної роботи *Дослідження та оптимізація автомобільних транспортних потоків засобами імітаційного моделювання*

1.Коротка характеристика об'єкта проектування *Задача збільшення автотранспортних потоків в транспортній мережі та їх раціональної організації є актуальною, а її вирішення та розробка безпечної моделі дорожнього руху методом імітаційного моделювання координованих транспортних потоків в міський дорожньої мережі надасть можливість урахування впливу різних чинників на організацію руху транспортних засобів на міських автодорогах.*

2. Відповідність теми дипломного проекту завданню
Відповідає

Обсяг матеріалу: пояснювальна записка *86* сторінок та технічна документація *94* сторінок

3. Якість поданого матеріалу: *матеріал подано якісно та повно, без зауважень.*
(якість і повнота викладу матеріалу)

4. Якість та обґрунтованість проектних рішень
виконана у повному обсязі
(у повному обсязі, не в повному)

5. Дослідницька частина і якість її опрацювання *Тема добре опрацьована, отримані якісні результати. За результатами роботи виконані дослідження, що знайшли широке практичне застосування у відповідній предметній області.*

6. Ритмічність роботи, оцінка загальної теоретичної підготовки і ступеня самостійності роботи дипломника, інші зауваження, побажання

Робота виконувалася ритмічно та самостійно, строки виконання магістерської роботи складалися вчасно, маріструант володіє високим рівнем теоретичної підготовки та заслуговує високої оцінки за виконану роботу

7. Оцінка проекту, що рекомендується, і рейтинг
відмінно
(відмінно, добре, задовільно)

«17» грудня 2021 р. _____
(підпис)