МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

МЕТОДИЧНІ ВКАЗІВКИ

до проведення лабораторних робіт з дисципліни

«Моделювання транспортних процесів»

для студентів денної та заочної форм навчання спеціальностей:

275 «Транспортні технології (на залізничному транспорті)»

275 «Транспортні технології (на автомобільному транспорті)»

2024

Методичні вказівки до проведення лабораторних робіт з дисципліни «Моделювання транспортних процесів» для студентів денної та заочної форм навчання спеціальностей: 275 «Транспортні технології (на залізничному транспорті)», 275 «Транспортні технології (на автомобільному транспорті)» / Укл.: С.М. Турпак, О.О. Острогляд. – Запоріжжя : НУ «Запорізька політехніка», 2024. – 87 с.

Укладачі: С.М. Турпак, проф., д-р техн. наук,
О.О. Острогляд, канд. техн. наук,

Рецензент: О.Ф. Кузькін, проф., д-р техн. наук,

Відповідальний
за випуск: Т.В. Кальченко, зав. навч. лаб.

Затверджено на засіданні
кафедри «Транспортні технології»
протокол № 2
від 08 серпня 2024 р.

Рекомендовано до видання
НМК Транспортного факультету
протокол № 2
від 22 серпня 2024 р.

ЗМІСТ

1 ПЕРЕЛІК ЛАБОРАТОРНИХ РОБІТ

1. Загальна характеристика програми Anylogic.
   Елементи інтерфейсу користувача. – 2 год.
2. Створення моделі розвантаження автомобілів – 2 год.
3. Створення моделі сортувальної гірки з використанням – 4 год.
   Залізничної бібліотеки середовища Anylogic
4. Моделювання руху автомобілів на перехресті – 2 год.

Усього – 14 год.

2 МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

2.1 Лабораторна робота № 1

Загальна характеристика програми Anylogic. Елементи інтерфейсу користувача.

Мета роботи – ознайомитися з програмним комплексом Anylogic, вивчити інтерфейс, основні елементи та принципи роботи з програмним середовищем.

Стисла теоретична довідка

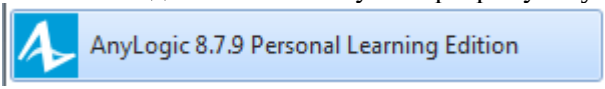
Компанія AnyLogic - багатонаціональна команда з Росії, Європи та США з глобальною мережею партнерів по всьому світу. Вона розробляє програмне забезпечення для вирішення широкого кола бізнес-завдань. Гнучкість програмних продуктів дозволяє користувачам відображати у моделях системи будь-якої складності з необхідним рівнем деталізації. Продукти AnyLogic використовуються у тисячах комерційних організацій та навчальних закладів у всьому світі [1].

Порядок виконання роботи:

- Ознайомитися з інтерфейсом та бібліотеками програмного середовища Anylogic.
- Оформити звіт.

Методичні вказівки до виконання лабораторної роботи

Знаходимо в меню «Пуск» програму Anylogic:



виконуємо її запуск.

Після запуску відкривається початкове вікно, на якому представлена корисна інформація (посилання на приклади моделей, ресурси). Детальне ознайомлення з цими матеріалами здійснимо пізніше. Закриваємо це вікно (рисунок 2.1).

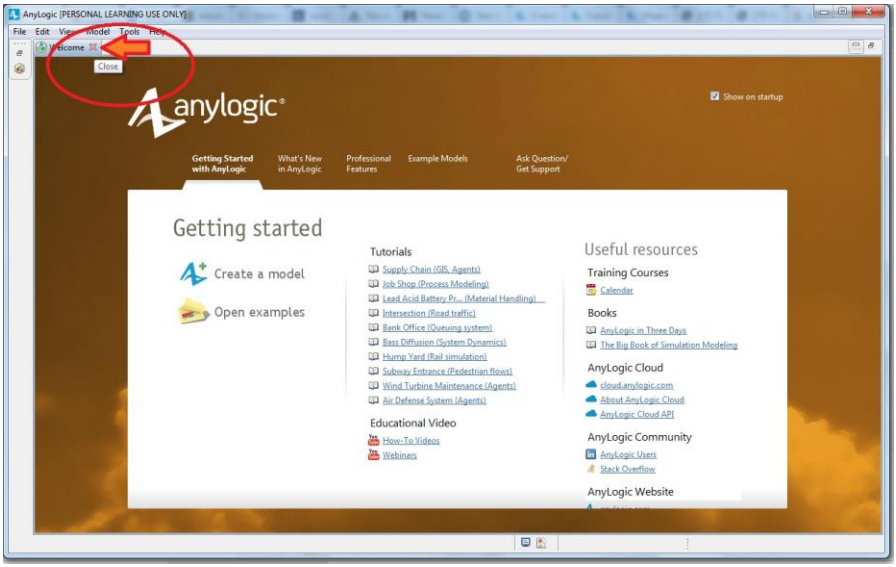


Рисунок 2.1 – Початкове вікно програми Anylogic

Створюємо власну програму у середовищі Anylogic, з якою будемо працювати надалі. Для цього обираємо на верхній панелі інструментів «File» - «New» - «Model» (рисунок 2.2). Можливе використання скороченого варіанту «Ctrl + N».

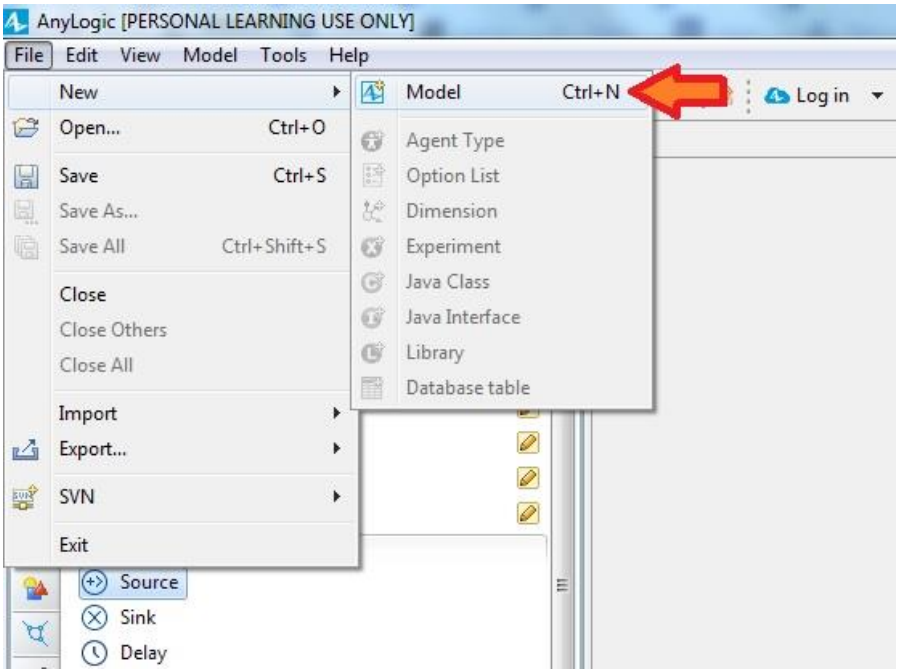


Рисунок 2.2 – Створення моделі в середовищі Anylogic

З’являється діалогове вікно (рисунок 2.3). У верхньому полі «Model name» зазначаємо ім’я програми у вигляді «Model_1», де замість цифри «1» вказуємо номер особистого варіанту.

Змінюємо ще поле «модельний час», обираючи зі списку одиниці – «хвилини» (див. рисунок 2.3).

Натискаємо кнопку «Finish».

В середній вкладці, яка з’явилась на вікні програми «Main» будемо розміщувати елементи програмного середовища Anylogic. Ознайомлення з цими елементами розпочнемо з бібліотеки моделювання процесів, яка підтримує парадигму моделювання дискретних подій.

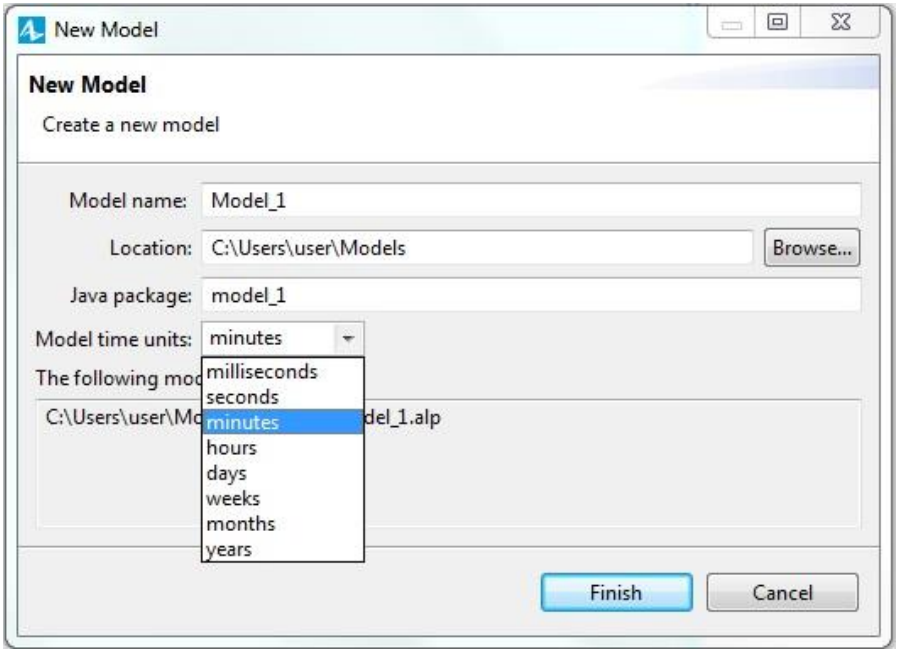


Рисунок 2.3 – Визначення ім’я та одиниць модельного часу програми

Використовуючи об’єкти бібліотеки моделювання процесів, ми можемо моделювати реальні системи з точки зору агентів (транспортні засоби), процесів (послідовності операцій, які зазвичай включають черги, затримки, використання ресурсів) та ресурсів. Процеси вказуються у вигляді блок-схем. Можна скласти великі складні системи на будь-якому рівні деталізації, навіть, враховуючи обмеження для програми, яка нами використовується (безкоштовна навчальна версія). Іншою важливою особливістю бібліотеки моделювання процесів є можливість створювати складні анімації моделей процесів.

На верхній панелі інструментів в бібліотеці моделювання процесів (після створення нами програми, саме вона є активною) підводимо курсор до елементу «Source». З’являється вікно підказки, де є можливість розгляду прикладу використання даного елементу (рисунок 2.4). Обираємо «demo»-підказку.

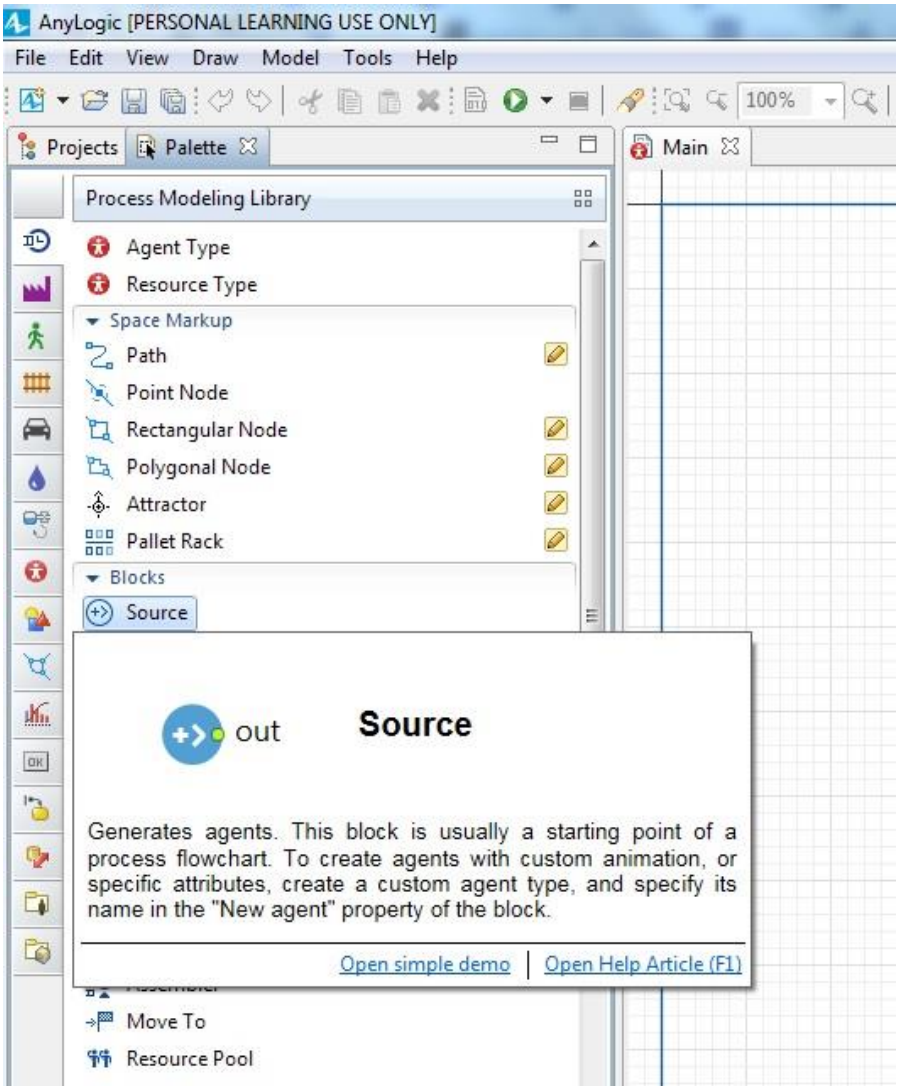


Рисунок 2.4 – Елемент «Source»

Перед тим, як розглянути приклад (рисунок 2.5), який з'явився на екрані, ознайомимось із теоретичними матеріалами.

Елемент «Source» генерує агентів. Зазвичай є відправною точкою моделі процесу.

Агенти можуть бути стандартними або будь-якого типу агента, визначеного користувачем. Ви можете налаштувати згенеровані агенти, вказавши тип агента в полі «Новий агент», а потім вказавши дію, яку слід виконати до того, як агент вийде з блоку «Джерело» в полі «Дія при виході».

Існує кілька способів визначити, коли і скільки агентів має бути створено.

Ми можемо встановлювати інтенсивність надходження агентів (і змінювати її динамічно, використовуючи функцію set_rate()), час між надходженнями, розклад часу прибуття та кількості, а також можемо програмно викликати функцію inject().

Наприклад, пуассонівський потік надходжень можна реалізувати, обираючи прибуття з відповідною інтенсивністю (нормальний розподіл) або вказуючи експоненціально розподілений час між прибуттями. Ми також можемо встановити кількість агентів у кожному прибутті та обмежити загальну кількість прибуття.

У деяких випадках має сенс використовувати два або більше вихідних блоків для реалізації складних шаблонів прибуття.

За замовчуванням «Source» не дозволяє агентам чекати на виході, доки їх не буде використано наступним блоком. Можна змінити це налаштування, знявши вибір із додаткового параметра «Примусове витискання» та обравши необхідну поведінку зі списку «Агенти, які не можуть вийти».

Крім «Source», існують інші способи створення агентів у моделях бібліотеки моделювання процесів. Наприклад, можна використовувати блок «Enter» як відправну точку потоку і викликати його функцію take() для введення агентів. Остання функція має сенс, коли агенти генеруються в іншому місці (наприклад, діаграмою стану або подією) і їх просто потрібно ввести в процес.

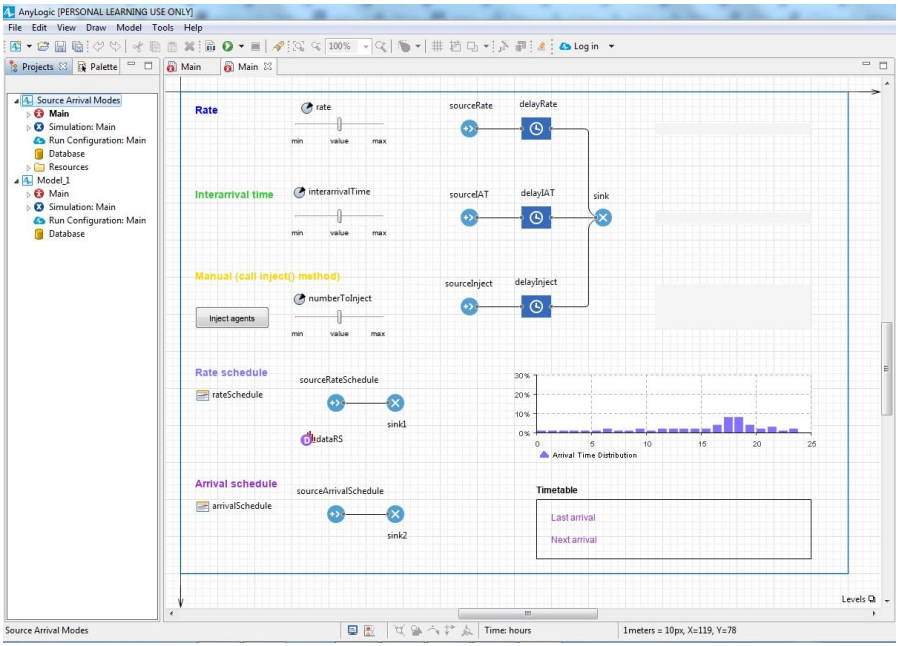


Рисунок 2.5 – Приклад використання елементу «Source»

Знайомство з наведеним прикладом буде цікавішим, якщо здійснити його запуск. Для цього виконаємо процедуру, показану на рисунку 2.6.

Після запуску самостійно виконується зміна параметрів роботи моделі, візуальне відображення результатів фіксується у звіті.

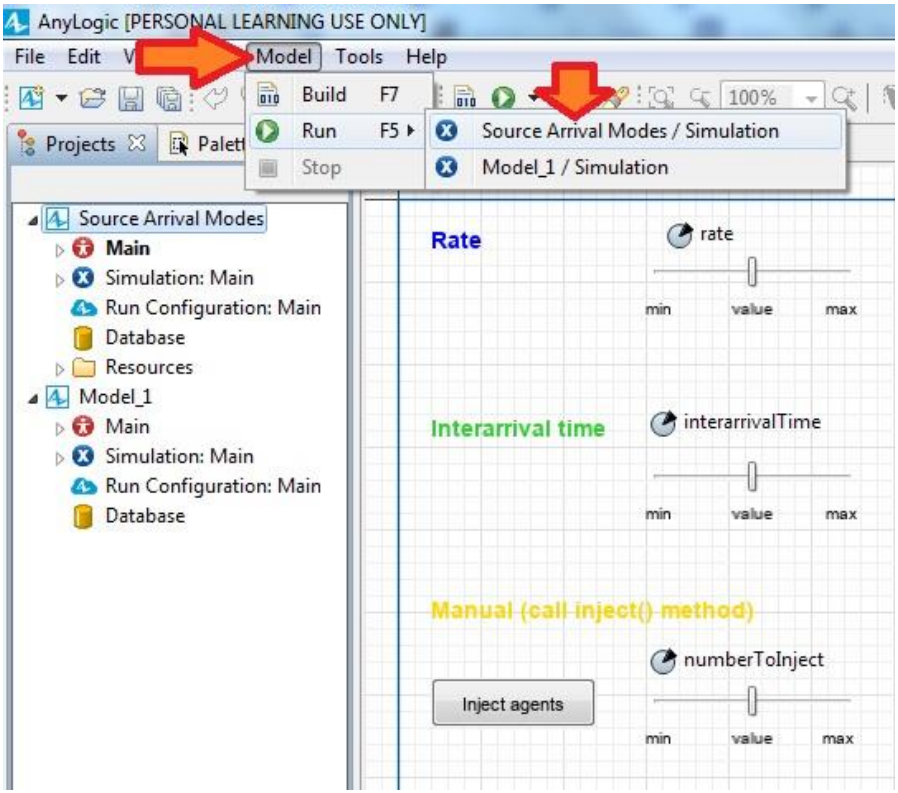


Рисунок 2.6 – Запуск

2.2 Лабораторна робота № 2

Створення моделі розвантаження автомобілів.

Мета роботи – набуття практичних навичок створення найпростішої моделі обслуговування автомобілів на вантажних фронтах шляхом використання дискретно-подійного підходу моделювання.

Стисла теоретична довідка

Бібліотека ***Process Modeling Library*** програмного комплексу AnyLogic підтримує дискретно-подійний, або "процесний" підхід моделювання. За допомогою блоків ***Process Modeling Library*** можна моделювати системи реального світу, динаміка яких представляється як послідовність операцій (прибуття, затримка, захоплення ресурсу, розподіл тощо) над агентами, що представляють клієнтів, документи, дзвінки, пакети даних, транспортні засоби тощо. Ці агенти самі не контролюють свою динаміку, але можуть мати певні атрибути, що впливають на процес їх обробки або накопичують статистику (загальний час очікування, вартість).

Діаграми процесів AnyLogic ієрархічні, масштабовані, розширювані та об'єктно-орієнтовані, що дозволяє користувачеві моделювати складні системи будь-якого рівня детальності. Іншою важливою особливістю ***Process Modeling Library*** є можливість створення складних анімацій процесних моделей. [2].

Порядок виконання роботи

У ході лабораторної роботи необхідно модель простої системи обслуговування, а саме модель вивантаження автомобілів на вантажних фронтах. Вивантаження автомобілів може виконуватися на складі вручну вантажниками, або за допомогою навантажувачів. Робота виконується у такій послідовності:

- 1.Створити просту модель та побудувати її діаграму процесу.
- 2. Створити анімацію моделі.
- 3. Задати модельні ресурси.
- 4. Задати збір статистики результатів моделювання.
- 5. Оформити звіт.

Методичні вказівки до виконання лабораторної роботи

Створення простої моделі

Спочатку ми створимо найпростішу модель, у якій розглядатимемо вивантаження автомобілів на вантажних фронтах.

Створюємо нову модель. У полі Ім'я моделі вводимо її назву ***Unloading***. Встановлюємо одиниці модельного часу - години.

У створеній моделі вже є один тип агента ***Main*** та експеримент ***Simulation***. Агенти. У нашому випадку агент ***Main*** послужить місцем, де ми задамо всю логіку моделі: тут ми розташуємо схему розташування фронтів вивантаження і побудуємо діаграму процесу потоку автомобілів.

Створення діаграми процесу

Тепер ми задамо динаміку процесу, створивши діаграму з блоків бібліотеки ***Process Modeling Library***. Кожен блок задає певну операцію, яка буде проводитися над агентами, що проходять по діаграмі процесу.

Діаграма процесу AnyLogic створюється шляхом додавання блоків бібліотеки з палітри на діаграму агента, з'єднання їх портів і зміни значень властивостей блоків відповідно до вимог вашої моделі.

За умовчанням при створенні нової моделі панелі ***Pallete*** відкривається бібліотека ***Process Modeling Library***. Додайте блоки бібліотеки на діаграму та з'єднайте їх, як показано на рисунку 2.7. Щоб додати блок на діаграму, перетягніть потрібний елемент із панелі до графічного редактора.

Дана схема моделює найпростішу систему черги, що складається з джерела агентів, затримки (черги перед затримкою) і фінального знищення агентів.

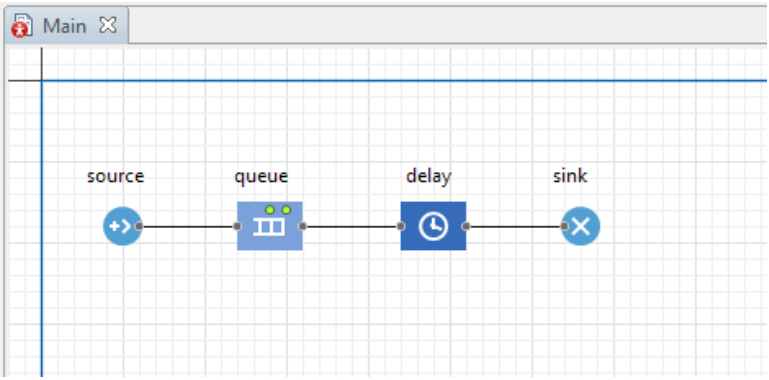


Рисунок 2.7 – Діаграма процесу моделі

Блок ***Source*** генерує агентів певного типу. Зазвичай він використовується як початкова точка діаграми процесу, формалізує потік агентів. У нашому прикладі агентами будуть автомобілі, а блок ***Source*** моделюватиме їх надходження до фронтів вивантаження.

Блок ***Queue*** моделює чергу агентів, які очікують прийому блоками, що йдуть за даними в діаграмі процесу. У нашому випадку він моделюватиме чергу автомобілів, які чекають на звільнення фронтів вивантаження.

Блок ***Delay*** затримує агентів на заданий період, представляючи в нашій моделі банкомат, у якого відвідувач банківського відділення витрачає свій час на проведення необхідної йому операції.

Блок ***Sink*** знищує агентів, що надійшли. Зазвичай він використовується як кінцева точка потоку агентів (і діаграми процесу відповідно).

Налаштування блоків діаграми

✓ Щоб змінити властивості елемента, виділіть елемент у графічному редакторі або в панелі ***Projects*** (проєкти), клацнувши мишею. Властивості елемента відкриються на панелі ***Properties*** (властивості).

✓ Перейдіть до блоку джерела. На панелі ***Properties*** вкажіть, як часто повинні прибувати автомобілі. Введіть 0.5 та виберіть години в полі ***Arrival rate*** (інтенсивність прибуття).

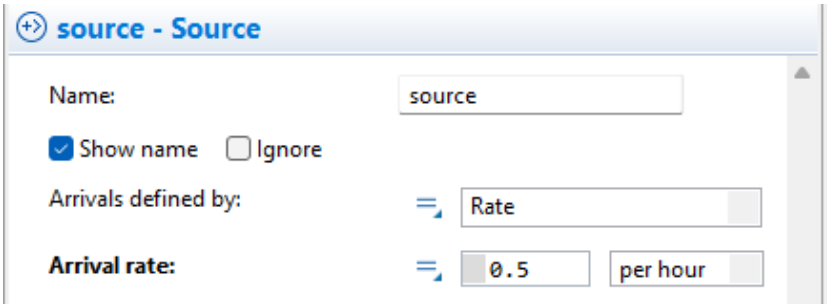


Рисунок 2.8 – Встановлення властивостей Source

✓ Змініть властивості блоку ***queue***. Введіть у поле ***Capacity*** (місткість) значення 10, у черзі зможе знаходитися не більше 10 автомобілів.

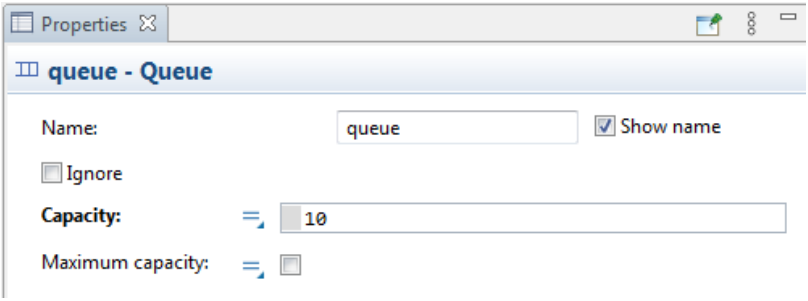


Рисунок 2.9 – Встановлення властивостей Queue

✓ Змініть властивості блоку ***delay*.** Назвіть блок *Front*. Задайте час обслуговування в полі ***Delay time*** (час затримки), розподілений за трикутним законом із середнім значенням, рівним 1.5, мінімальним – рівним 0.8 та максимальним – 3.5 години.

Функція ***triangular()*** є стандартною функцією генератора випадкових чисел AnyLogic. AnyLogic надає функції та інших випадкових розподілів, таких як нормальне, рівномірне, трикутне, і т.д.

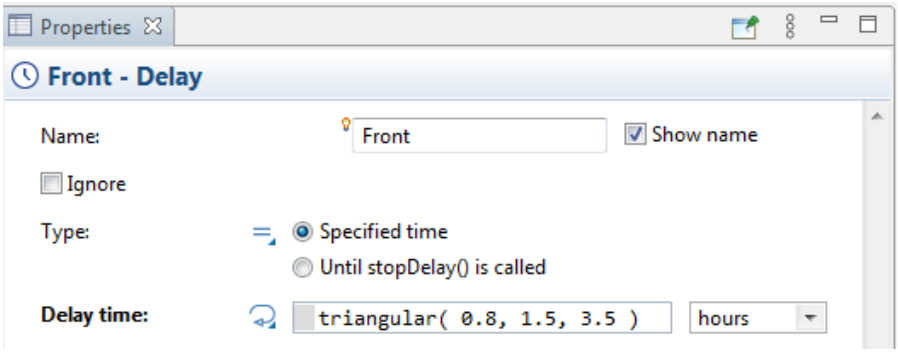


Рисунок 2.10 – Встановлення властивостей Delay

Запуск моделі

Спочатку побудуйте вашу модель за допомогою кнопки панелі інструментів ***Build model*** (Побудувати модель). Якщо у моделі є якісь помилки, то побудова не буде завершена, і в панель ***Problems*** (помилки) буде виведена інформація про помилки, виявлені в моделі. Подвійним клацанням миші помилково в цьому списку ви можете перейти до місця помилки, щоб виправити її. Після того, як ви виправите всі помилки та успішно побудуєте вашу модель, ви можете її запустити.

Запустивши модель, ви побачите вікно моделі (рис. 2.11). У ньому буде відображено презентацію агента верхнього рівня моделі. За замовчуванням це тип агента ***Main***.

Ви можете змінити швидкість виконання моделі за допомогою кнопок панелі інструментів ***Slow down*** (уповільнити) та ***Spead up*** (прискорити). Ви можете стежити за станом будь-якого блоку діаграми процесу під час виконання моделі за допомогою вікна інспекту цього блоку. Щоб відкрити вікно інспекту, клацніть на піктограму блоку. У вікні інспекту буде відображено базову інформацію щодо виділеного блоку: наприклад, для блоку ***Queue*** буде відображено місткість черги, кількість агентів, що пройшли через кожен порт блоку, тощо.

Рядок ***Capacity*** (місткість) відображає кількість агентів, що знаходяться в блоці разом з ID цих агентів.

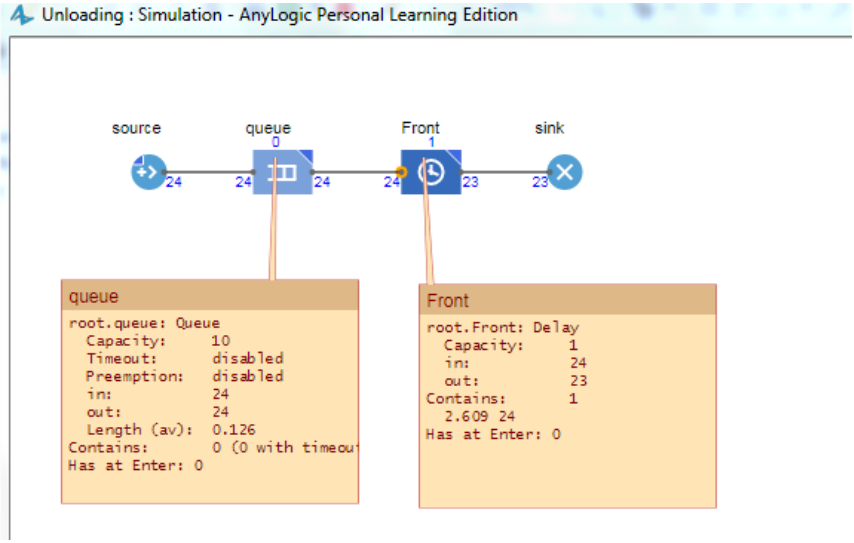


Рисунок 2.11 – Вікно запуску моделі

Створення анімації моделі

Хоча ми й могли аналізувати роботу запущеної нами щойно моделі за допомогою діаграми процесу, але куди зручніше було б мати наочнішу анімацію модельованого нами процесу.

Оскільки в нашому випадку нас не цікавить конкретне розташування об'єктів у просторі, то ми можемо просто додати схематичну анімацію об'єктів, що цікавлять нас - у нашому випадку ми хочемо бачити на анімації фронт вивантаження і чергу автомобілів.

Анімація моделі малюється в тій же діаграмі (у графічному редакторі), в якій задається і діаграма процесу, що моделюється.

✓ Намалюємо точковий вузол, що означає фронт вивантаження. Спочатку відкрийте панель ***Space Markup*** (розмітка простору) панелі ***Pallete.***

✓ Перетягніть елемент ***Point Node*** (Точковий вузол) з панелі ***Space Markup*** (розмітка простору) до графічного редактора та помістіть його під блок-схемою процесу.

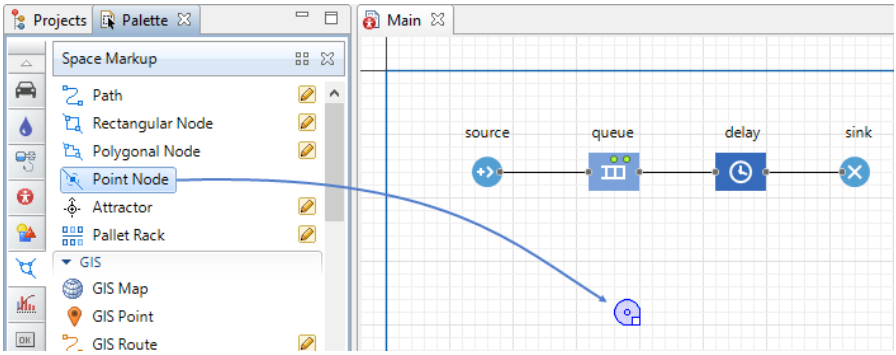


Рисунок 2.12 – Створення анімації моделі

✓ Натисніть елемент ***Point Node*** у графічному редакторі, щоб відкрити панель ***Properties*** (властивості). Ми з вами хочемо, щоб під час моделювання змінювався колір нашої фігури, тому введіть вираз, який постійно обчислюватиметься заново при виконанні моделі, у полі ***Color***:

Front.size() > 0? red : green

Тут ***Front*** це ім'я нашого блоку ***Delay***. Функція ***size()*** визначає кількість автомобілів, що обслуговуються на даний момент часу. Якщо фронт зайнятий, то колір круга буде червоним, в іншому випадку - зеленим.

✓ У властивостях блоку ***delay***, у діаграмі процесу, в параметрі ***Agent location*** (місце агентів) виберіть точковий вузол пункту - ***node***, який ми тільки намалювали. Ви можете вибрати його зі списку відповідних об'єктів, клацнувши стрілку "вниз", або вибрати фігуру з графічного редактора, попередньо клацнувши кнопку праворуч від параметра.

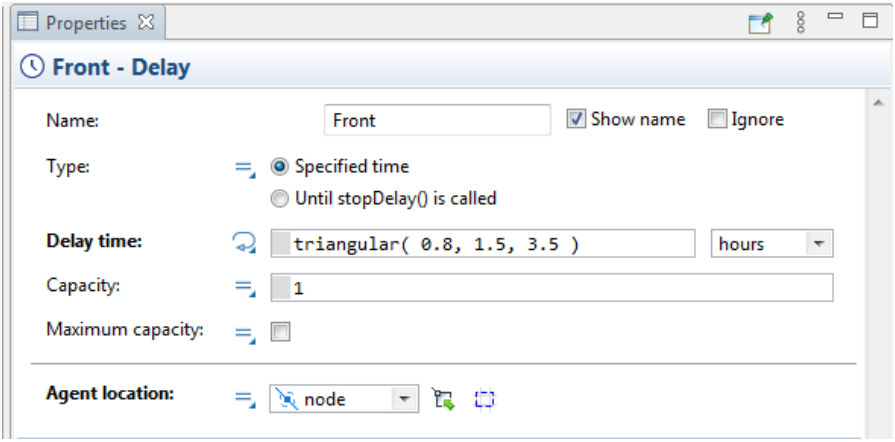


Рисунок 2.13 – Встановлення місце розташування агентів, що обслуговуються

Задайте фігуру анімації черги автомобілів, що очікують вивантаження.

- Намалюємо місце стоянки, що означає чергу до фронту вивантаження. Спочатку відкрийте панель Space Markup (розмітка) простору панелі Pallete (панелі).
- Подвійним клацанням виберіть пункт Rectangular Node палітри розмітки, щоб перейти в режим малювання.
- Тепер намалюйте прямокутник в тому місці діаграми, де будуть розташовуватися автомобілі, що очікують черги вивантаження, задайте йому назву waiting.
- Далі у властивостях об’єкту queue у діаграмі процесу вибираємо місце розташування агентів, що знаходяться у черзі. Для цього в пункті Agent Location вибираємо waiting (рис

Ви можете вибрати його зі списку відповідних об'єктів, клацнувши стрілку "вниз", або вибрати фігуру з графічного редактора, попередньо клацнувши кнопку праворуч від параметра.

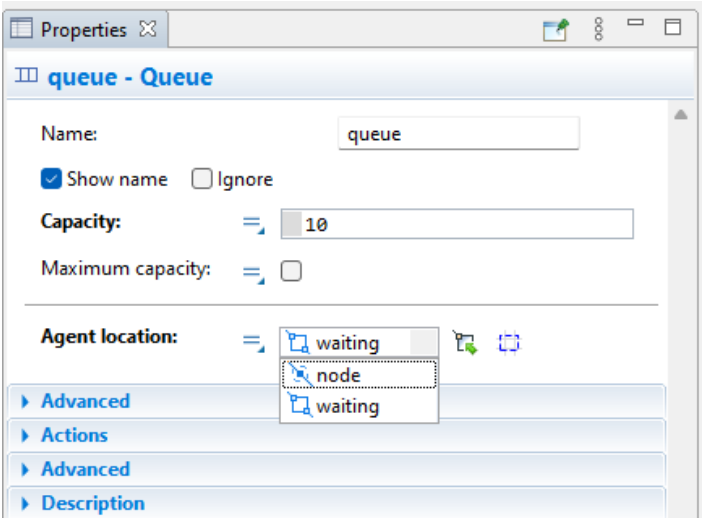


Рисунок 2.14 – Встановлення місце розташування агентів, що очікують черги

Запустіть модель. Ви побачите, що наша модель тепер має найпростішу анімацію - склад і чергу автомобілів до нього. Колір фігури складу змінюватиметься залежно від того, чи обслуговується автомобіль на даний момент часу.

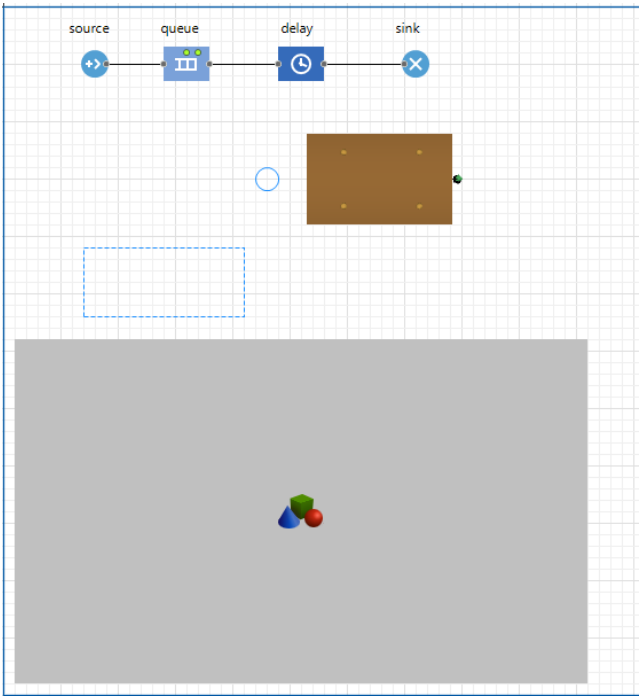
Додавання 3D анімації

Насамперед нам потрібно буде додати на діаграму типу агента 3D Вікно. 3D Вікно використовується для завдання на діаграмі агента області, де під час запуску моделі буде відображатися тривимірна анімація цієї моделі.

Щоб додати 3D вікно перетягніть елемент ***3D Window*** з розділу 3D панелі ***Presentation*** (презентація) до графічного редактора. Ви побачите у графічному редакторі зафарбовану сірим область. Помістіть її туди, де ви хочете бачити 3D анімацію під час запуску моделі.

Тепер ви можете запустити модель та спостерігати просту 3D анімацію. При створенні 3D вікна AnyLogic додає область перегляду, яка дозволяє легко перемикатися до сцени 3D анімації під час виконання моделі. Щоб перейти до 3D анімації в запущеній моделі, відк-

рийте панель розробника, клацнувши кнопку Показати/приховати панель розробника в правому куті панелі керування. У панелі розробника розкрийте список Вибрати область і показати і виберіть опцію ***[window3d].***



Додавання 3D об'єктів

За замовчуванням автомобілі в нашій моделі позначалися кольоровими точками та відображалися кольоровими циліндрами у 3D анімації. Щоб задати для нього відповідну фігуру анімації, нам потрібно створити новий тип агента.

✓ Відкрийте бібліотеку ***Process Modeling Library*** на панелі ***Palette***.

✓ Перетягніть елемент ***Agent type*** (тип агента) до графічного редактора (рис 2.15).

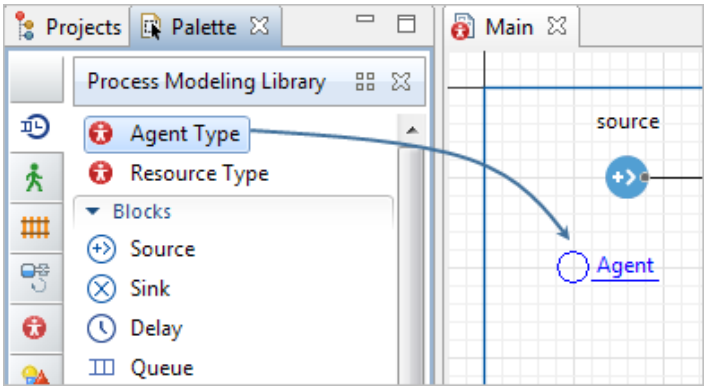


Рисунок 2.15 – Створення нового типу агента

✓ Відкриється діалогове вікно Майстра створення агентів на етапі ***Creating new agent type*** (створення нового типу агента). Введіть ***Truck*** у полі ***Agent type name*** (ім'я нового типу), залиште опцію ***Create the agent type "from scratch"*** (створити новий тип агента "з нуля") вибраною.

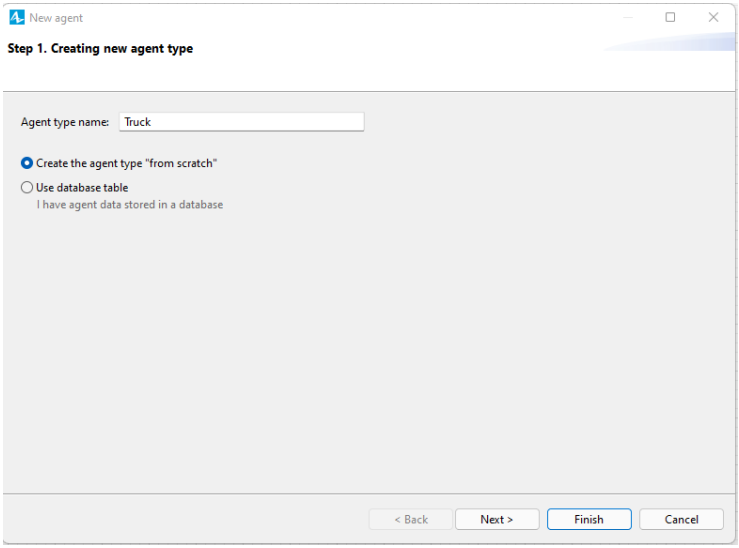


Рисунок 2.16 – Майстер створення агентів

✓ Виберіть опцію 3D для типу анімації та фігуру анімації Автомобіль зі списку 3D фігур (рис. 2. 17).

✓ Натисніть кнопку ***Finish***. Діаграма нового агента ***Truck*** відкриється автоматично. Ви можете знайти 3D фігуру автомобіля на початку координат.

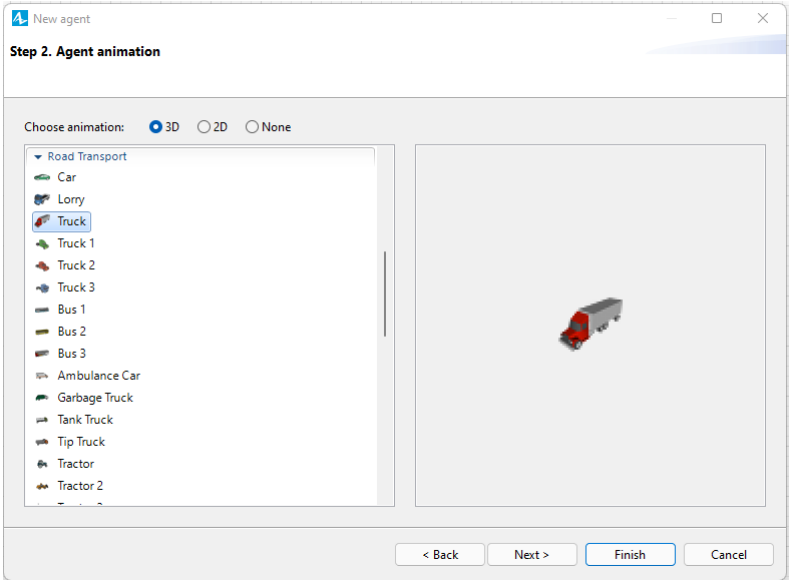


Рисунок 2.17 – Вибір анімації агента

Налаштуйте використання нового типу агентів у блок-схемі. На діаграмі ***Main*** виділіть блок ***source*** у графічному редакторі. Виберіть тип агента ***Truck*** у списку ***New agent***.

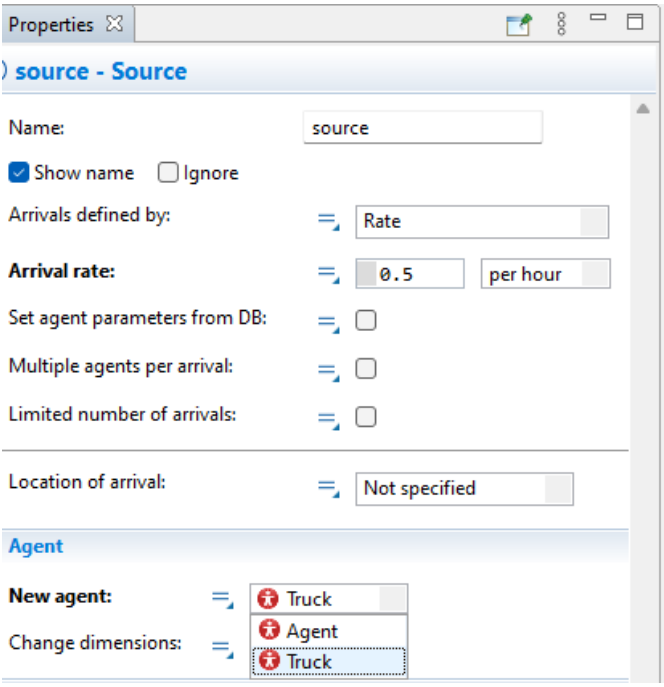


Рисунок 2.18 – Вибір типу агенту об’єкту ***source***

Відкрийте панель властивостей точкового вузла та переключіть елемент керування Видимість на значення немає. Таким чином, розмітка простору не буде видно на анімації під час виконання моделі.

Додати об'єкт складу:

- Відкрийте панель 3D Об'єкти на панелі Палітра.
- Перетягніть 3D фігуру Warehouse (склад) із секції Building (будівлі) палітри у графічний редактор та помістіть її на точковий вузол.
- Запустіть модель, щоб переконатися, що фігура складу розташована в потрібному ракурсі.

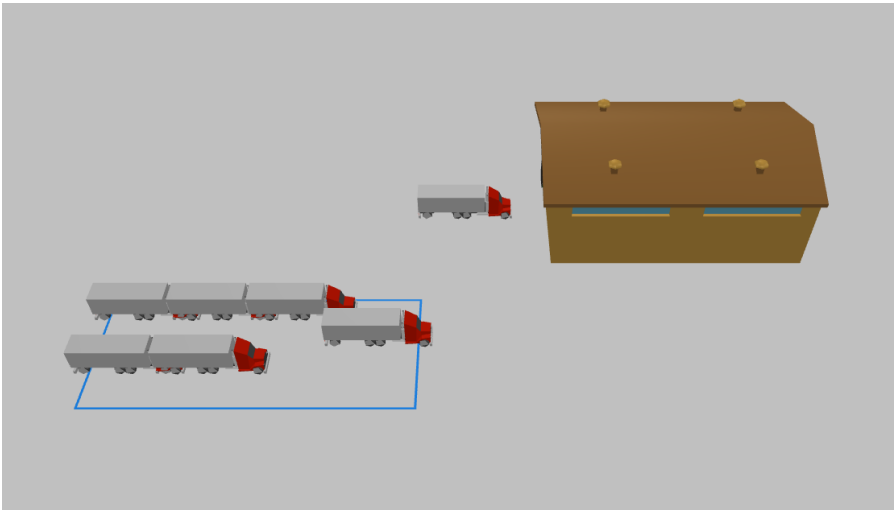


Рисунок 2.19 – Вікно запуску моделі в 3D форматі

Додавання ресурсів моделі

Тепер ми ускладнимо нашу модель, додавши до неї ресурси – навантажувачі. Ресурс – це спеціальний блок бібліотеки ***Process Modeling Library***, який може бути потрібним агенту для виконання якогось завдання. У кожний момент часу ресурс може бути зайнятий лише одним агентом. У нашому випадку завантажені автомобілі (агенти) потребують обслуговування навантажувачами (ресурсами).

- Відкрийте бібліотеку ***Process Modeling Library*** та перетягніть на діаграму ***Main*** блок ***Service***. Блок ***Service*** захоплює для агента задану кількість ресурсів, затримує агента, а потім звільняє захоплені ним ресурси.
- У властивостях ***Properties*** блоку ***service*** змініть параметри блоку таким чином:
  - до всіх фронтів з навантажувачами вестиме одна загальна черга. Задайте максимальну кількість автомобілів у цій черзі в поле ***Queue capacity*** (місткість черги): 20.
  - введіть у поле ***Delay time*** (час затримки): 1 година

Змоделюйте розподіл автомобілів на фронти вивантаження.

➢ З бібліотеки ***Process Modeling Library*** перетягніть на діаграму процесу ***Main*** блок ***SelectOutput*** у вільне місце між блоками ***source*** та ***queue***. Ви можете виділити кілька блоків діаграми процесу і перемістити їх разом або переміщати блоки по одному. ***SelectOutput*** є блоком прийняття рішення. Залежно від заданої вами умови агент, що надійшов в блок, надходитиме на один з двох вихідних портів блоку.

➢ Виділіть блок ***selectOutput*** у діаграмі процесу. У властивостях ***Properties*** цього блоку виберіть опцію ***Select True output*** (при виконанні умови) у параметрі ***if condition is true*** (вихід true вибирається). Переконайтеся, що в полі ***Condition*** (умова) стоїть ***randomTrue(0.5)***.

У цьому випадку автомобілі до фронтів з навантажувачами та до складу з ручним вивантаженням будуть надходити в рівній кількості.

➢ З'єднайте блоки ***selectOutput*** і ***service*** з іншими блоками так, як показано на рисунку нижче:

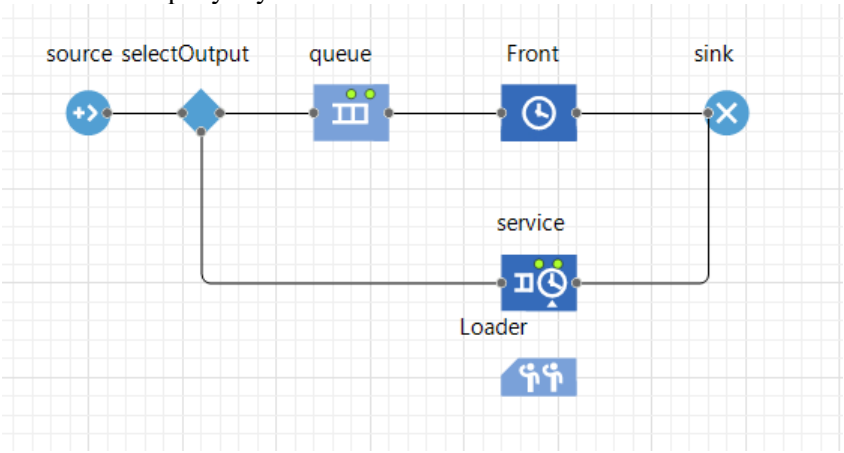


Рисунок 2.20 – Вигляд доповненої діаграми процесу

Додайте ресурси для сервісу

- З бібліотеки ***Process Modeling Library*** перетягніть блок ***ResourcePool*** на діаграму агента ***Main***. Блок ***ResourcePool*** визначає ресурси певного типу (у нашій моделі це будуть навантажувачі).
- Помістіть його, під блоком ***service*** та перейдіть до вікна властивостей ***Properties***.
- Назвіть блок ***Loader***. Задайте число навантажувачів у полі ***Capacity*** (кількість ресурсів): 4 (рис. 2.21).

- Блок ***ResourcePool*** вказується у блоках, які використовують ресурси, у нашому випадку це блок ***Service***. Тому необхідно змінити властивості блоку ***service*** діаграми процесу.

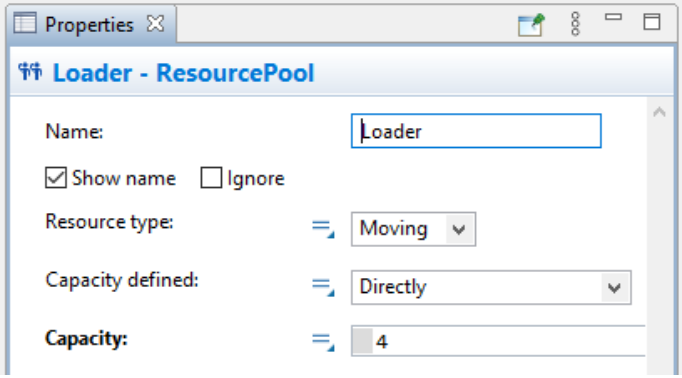


Рисунок 2.21 – Встановлення параметрів ресурсів

➢ У властивостях блоку ***Service*** виберіть ***Units of the same pool*** (ресурси одного типу) у параметрі ***Seize*** (захопити ресурси). Потім вкажіть блок ***Loader***, у параметрі ***ResourcePool*** (рис. 2.22). Ви можете вибрати його зі списку відповідних блоків, клацнувши стрілку "вниз", або вибрати фігуру з графічного редактора, попередньо клацнувши кнопку праворуч від параметра.

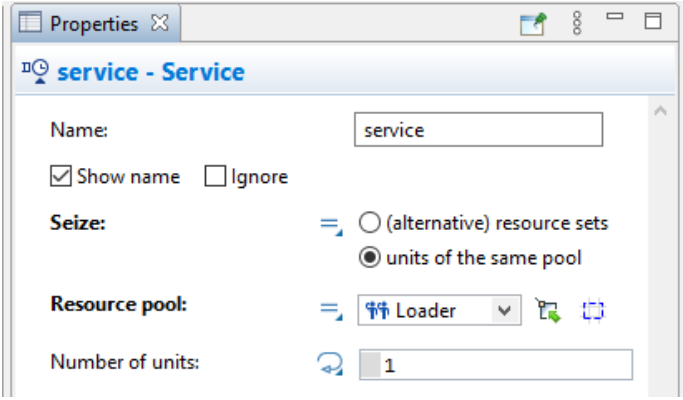


Рисунок 2.22 – Встановлення параметрів блоку ***Service***

➢ Також треба вказати місце розташування автомобілів, що очікують обслуговування навантажувачами. Будемо вважати що всі автомобілі, незалежно від способу вивантаження очікують своєї черги на одній території. Тому у властивостях блоку ***service*** у параметрі ***Agent location queue*** (місце агентів черги) обираємо прямокутний вузол ***waiting***.

Оскільки наша модель змінилася, ми повинні змінити її анімацію.

Додавання фігур розмітки простору

Задайте фігуру розмітки місця обслуговування автомобілів.

- Необхідно задати місце потрібно, де будуть розташовуватися автомобілі під час обслуговування навантажувачами.
- Використовуючи панель ***Space markup*** (розмітка простору) панелі ***Pallete*** малюємо прямокутний вузол і називаємо цю область ***truck_place*** (виділений прямокутник на рис.2.23).
- Перемкніть елемент керування ***Visible*** (видимість) у положення ***no***. Таким чином на анімації розмітка цієї області не буде відображатися.
- Щоб задати конкретне місце розташування кожного автомобіля, які будуть обслуговуватися навантажувачами, скористаємося атракторами. Виділіть прямокутний вузол ***truck_place*** у графічному редакторі та клацніть кнопку ***Attractors…*** (атрактори) у властивостях вузла. У вікні ***Attractors…*** вкажіть число атракторів 4, та натисніть OK. Ви побачите, що чотири атрактори з'явилися у вузлі ***truck_place*** на рівній відстані один від одного.

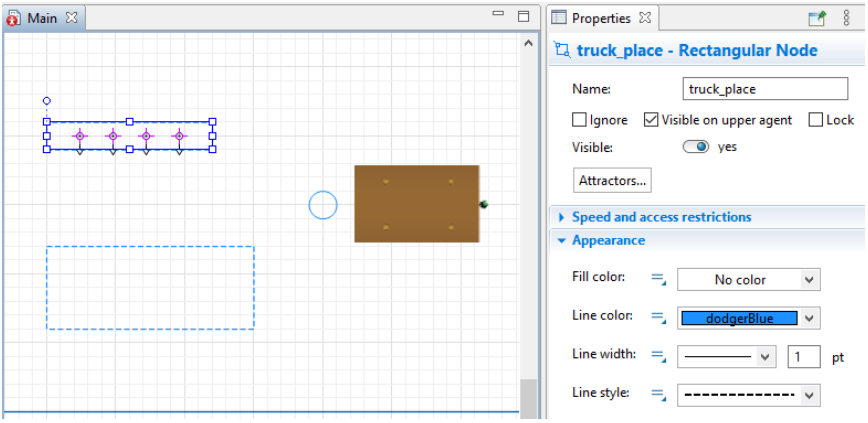


Рисунок 2.23 – Розмітка області розташування автомобілів, що обслуговуються навантажувачами

- Тепер нам необхідно послатися на цю фігуру у діаграмі процесу. Натисніть кнопку ***service*** і перейдіть до властивостей цього блоку. Виберіть намальований нами вузол ***truck_place*** у параметрі ***Agent location (delay)***.

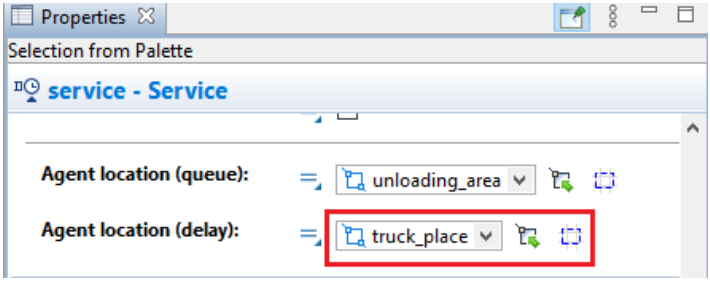


Рисунок 2.24 – Встановлення параметрів розташування автомобілів

Задайте фігуру місця розташування для навантажувачів.

– Аналогічним чином малюємо прямокутний вузол та називаємо цю область ***loader_place*** (рис. 2.25). Перемкніть елемент керування ***Visible*** (видимість) у положення ***no***.

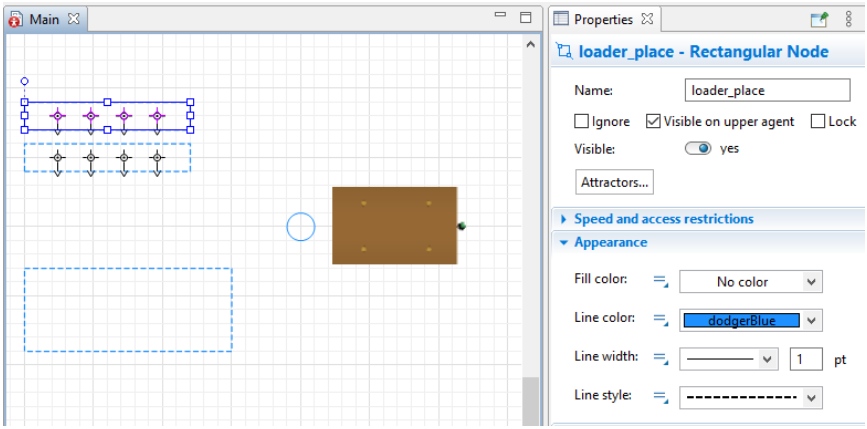


Рисунок 2.25 – Розмітка області розташування навантажувачів, що обслуговуються навантажувачами

– Розташування навантажувачів також задаємо атракторами Виділіть прямокутний вузол ***loader_place*** у графічному редакторі та клацніть кнопку ***Attractors…*** (атрактори) у властивостях вузла. У вікні ***Attractors…*** вкажіть число атракторів - 4, та натисніть OK. За необхідності ви можете змінювати направлення атракторів. Для цього затиснувши клавішу Shift та клацнувши по них мишею, у параметрі ***Orientation*** (орієнтація секції властивостей) ***Position and size*** (розташування та розмір) вибирається необхідний градус повороту фігури (рис. 2.26). В нашому випадку розташування атракторів нас влаштовує.

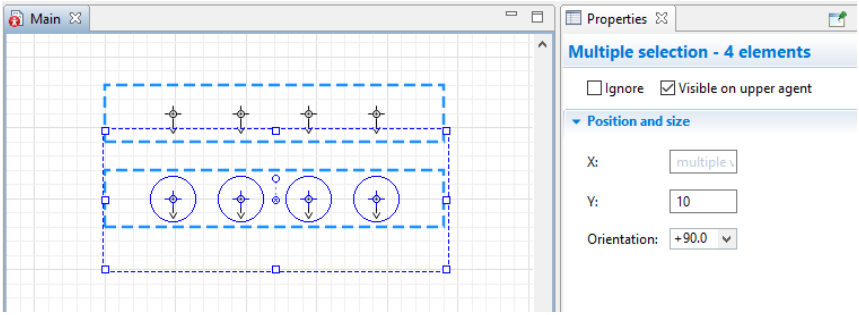


Рисунок 2.26 – Встановлення направлення атракторів

– Клацніть блок ***Loader*** у діаграмі процесу та перейдіть до його властивостей. Виберіть намальований вами вузол ***loader_place*** у параметрі ***Home location nodes*** (базове розташування).

Додавання 3D об'єктів

Додамо 3D фігури навантажувачів до нашої моделі. Для цього створимо новий тип ресурсів для їх анімації.

– Відкрийте бібліотеку ***Process Modeling Library*** на панелі ***Palette*** та перетягніть елемент ***Resource type*** (тип ресурсу) до графічного редактора (рис 3.27).

– Відкриється діалогове вікно Майстра створення агентів на етапі ***Creating new agent type*** (створення нового типу агента). Введіть ***Loaders*** у полі ***Agent type name*** (ім'я нового типу), залиште опцію ***Create the agent type "from scratch"*** (створити новий тип агента "з нуля") вибраною.

– Виберіть опцію 3D для типу анімації та фігуру анімації навантажувача зі списку 3D фігур ***Warehouse and Containers Terminal*** та натисніть кнопку ***Finish***.

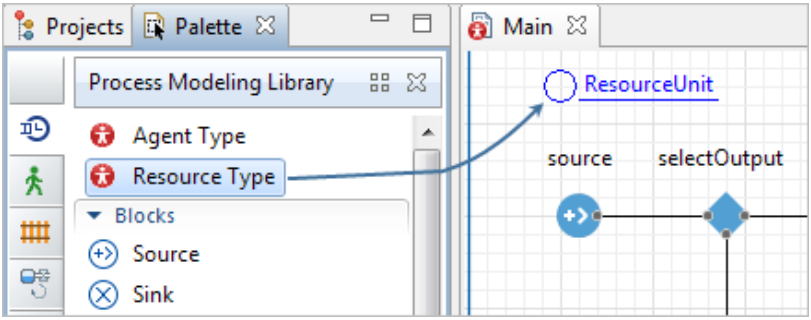


Рисунок 2.27 – Встановлення направлення атракторів

– Налаштуйте використання нового типу ресурсів у блок-схемі. У властивостях блоку ***Loader*** виберіть тип ресурсів ***Loaders*** у списку ***New resource unit*** (новий ресурс).

Для цього з панелі ***3D Objects*** зі списку ***Buildings*** виберіть фігуру ***Building 3*** та розташуйте її над вузлом розміщення навантажувачів.

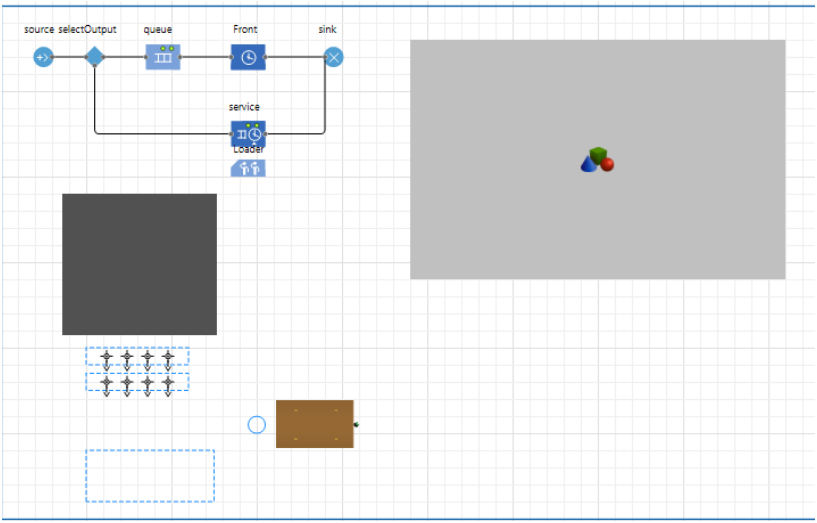


Рисунок 2.28 – Загальний вигляд моделі у графічному редакторі

Тепер ви можете запустити модель і побачити в 3D анімації, як деякі автомобілі їдуть на вивантаження до першого складу, а інші обслуговуються навантажувачами біля другого (рис. 2.29).

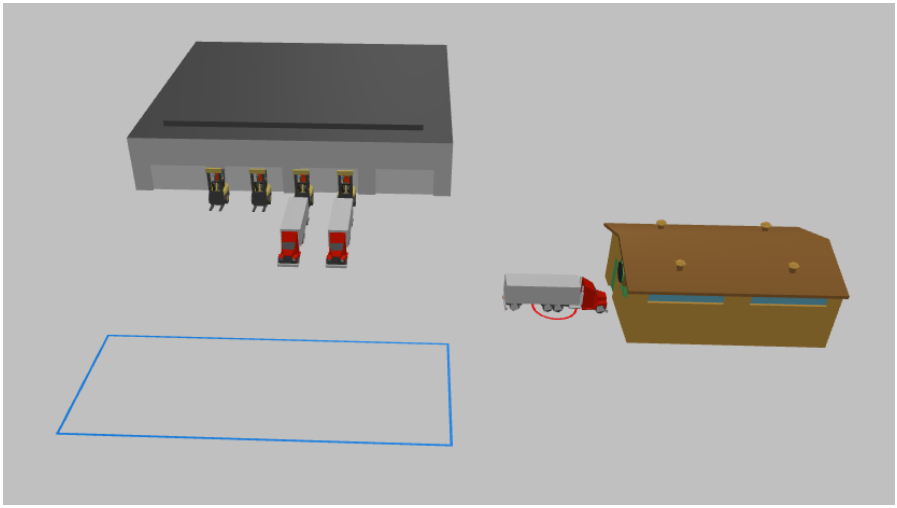


Рисунок 2.29 –3D анімація запущеної моделі

Збір статистики

AnyLogic надає користувачеві зручні засоби для збору статистики роботи блоків діаграми процесу. Ми можемо, наприклад, переглянути статистику (скажімо, статистику зайнятості складу і довжини черги), за допомогою діаграм.

Збір статистики використання ресурсів

Створимо діаграму для відображення зайнятості складу, де відбувається вивантаження вручну.

- Відкрийте панель ***Analysis*** (статистика). Перетягніть елемент ***Bar Chart*** (стовпчикова діаграма) до графічного редактору (рис. 2.30)
- Перейдіть до розділу ***Data*** властивостей стовпчикової діаграми. Змініть заголовок на Використання складу №1.

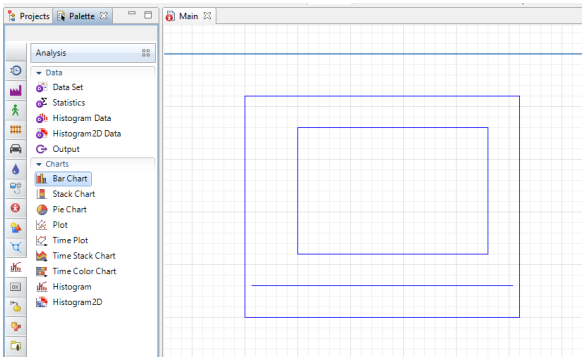


Рисунок 2.30 – Створення стовпчикової діаграми

- Введіть ***Front.statsUtilization.mean()*** у полі Значення. Тут ***Front*** – це ім'я нашого блоку ***Delay***. Кожен блок ***Delay*** має вбудований набір даних ***statsUtilization***, що займається збором статистики використання цього блоку. Функція ***mean()*** повертає середнє зі всіх виміряних цим набором даних значень. Ви можете використовувати інші методи збору статистики, такі як ***min()*** або ***max().***

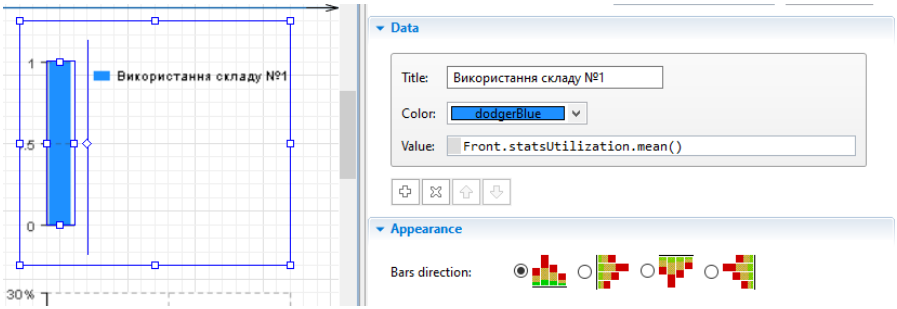


Рисунок 2.31 –Встановлення параметрів стовпчикової діаграми використання складу

Аналогічно створимо діаграму для відображення середньої довжини черги автомобілів до першого складу.

✓ Додайте ще одну стовпчикову діаграму. Змініть її розмір так, як показано на рис. 2.32.

✓ Перейдіть до розділу Зовнішній вигляд та виберіть останню опцію параметра ***Bars direction*** (напрямок стовпців), щоб стовпці стовпчикової діаграми зростали вліво.

✓ У розділі Data (дані) властивостей діаграми натисніть кнопку Додати елемент даних.

✓ Змініть Заголовок на ***Довжина черги до складу №1***. Задайте Значення: ***queue.statsSize.mean()***

Тут ***statsSize*** - це ім'я об'єкта типу "статистика" ***StatisticsContinuous***, що здійснює збір статистики розміру черги блоку ***Queue***.

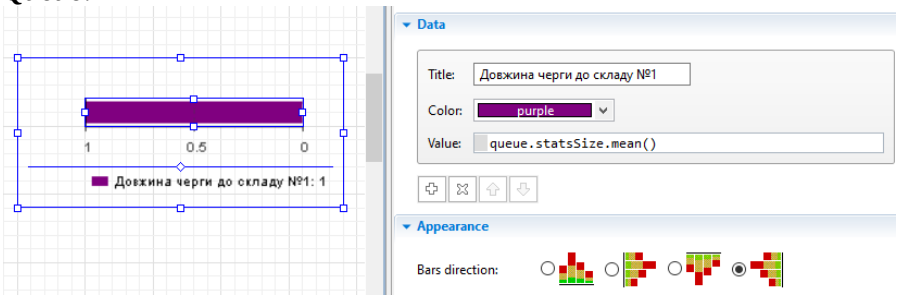


Рисунок 2.32 –Встановлення параметрів стовпчикової діаграми довжини черги

Збір статистики за часом обслуговування

Ми хочемо знати, скільки часу автомобіль проводить нашій системі (від моменту прибуття до завершення його вивантаження). Можемо зібрати цю статистику за допомогою блоків ***TimeMeasureStart*** та ***TimeMeasureEnd*** з Бібліотеки моделювання процесів та відобразимо зібрану статистику розподілу часів обслуговування клієнтів за допомогою гістограми. Щоб виміряти час, проведений агентами на певному відрізку діаграми процесу, ми повинні розмістити ці блоки відповідно на вході в відрізок, що цікавить нас, і на виході з нього. Перший блок зберігає час проходження агента через блок, а другий вимірює час, який провів агент на відрізку діаграми процесу після того, як залишив перший блок.

Додайте блоки вимірювання часу у діаграму процесу:

✓ З бібліотеки Бібліотеку моделювання процесів додайте блок ***TimeMeasureStart***, встановіть його між блоком ***source*** та блоком ***selectOutput***.

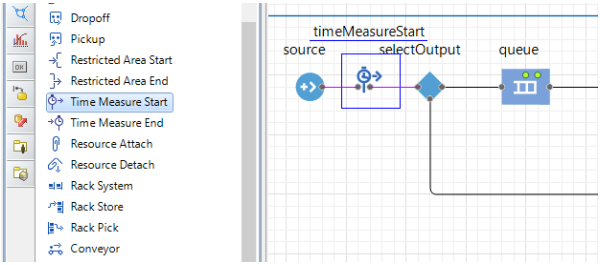


Рисунок 2.33 –Встановлення елементу фіксування часу ***TimeMeasureStart***

✓ Перетягніть блок ***TimeMeasureEnd*** з Бібліотеки моделювання процесів на графічний редактор та розмістіть його перед блоком ***sink***. Переконайтеся, що вхідний порт блоку з'єднаний із блоками ***Front*** та ***service***, а вихідний – з блоком ***sink***.

✓ Щоб розрахувати розподіл часу для агентів, у властивостях кожного блоку ***TimeMeasureEnd*** повинен бути вказаний як мінімум один блок ***TimeMeasureStart***. Відкрийте властивості вашого блоку та встановіть блок ***TimeMeasureStart*** у параметрі Блоки ***TimeMeasureStart***.

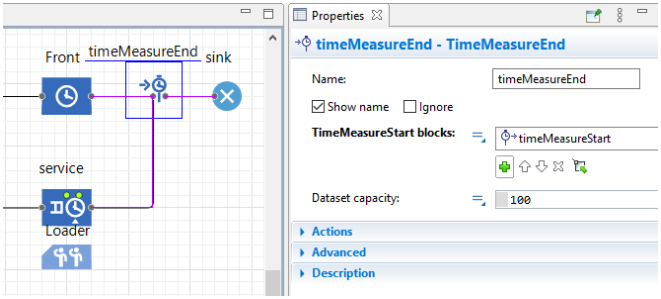


Рисунок 2.34 –Властивості елементу ***TimeMeasureEnd***

Додайте гістограму для відображення зібраної статистики:

- Перетягніть елемент ***Histogram*** (гістограма) з панелі ***Statistica*** до графічного редактора. Змініть розмір.
- Вкажіть, який елемент збору даних зберігає дані, які ви хочете відобразити на вашій гістограмі: у розділі Data (дані) властивостей гістограми натисніть ***Add histogram data*** (додати дані).
- Змініть заголовок даних ***Title***, що відображаються на ***Розподіл часу в системі***.
- Введіть ***TimeMeasureEnd.distribution*** у поле ***Histogram*** (рис.3.35).

TimeMeasureEnd - блок, який збирає розподіл часу перебування агентів у системі.

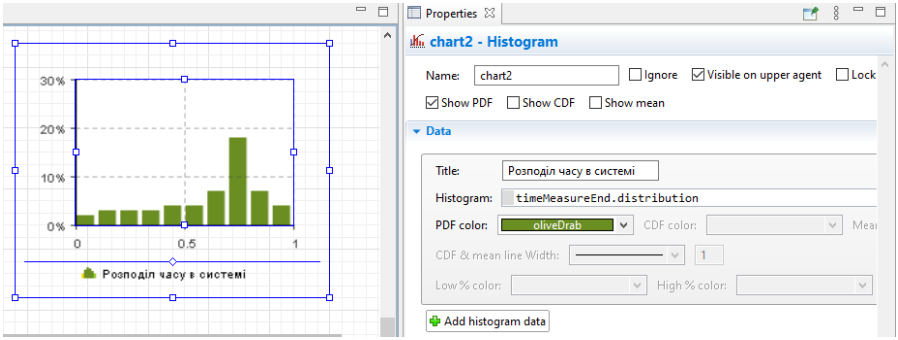


Рисунок 2.35 –Встановлення параметрів гістограми розподілу часу в системі

Запустіть модель. Увімкніть режим віртуального часу і поспостерігайте за тим, який вид набуде розподілу часу перебування клієнтів у системі. Якщо під час роботи моделі натиснути на блок ***TimeMeasureEnd***, отримаємо основні статистичні характеристики та інтервали розподілу досліджуваної випадкової величини (рис. 2.36).

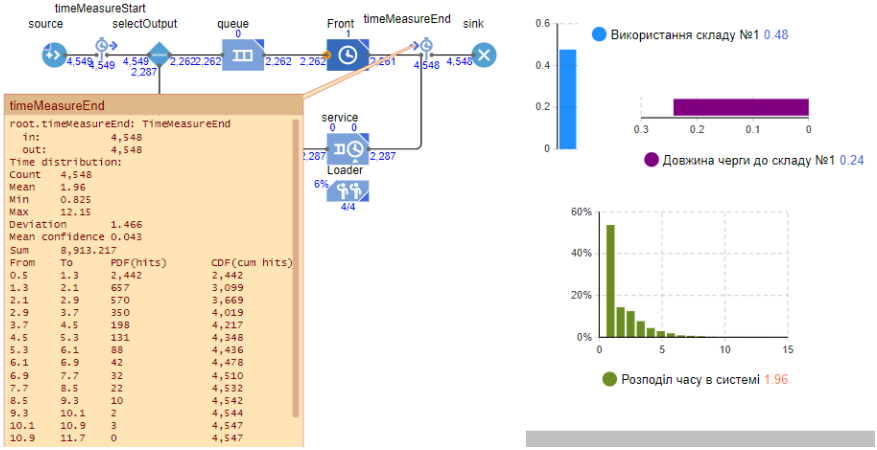


Рисунок 2.36 – Збір статистики в процесі моделювання

2.3 Лабораторна робота № 3

Створення моделі сортувальної гірки з використанням Залізничної бібліотеки середовища Anylogic

Мета роботи – ознайомитися основними елементами Залізничної бібліотеки Anylogic та особливостями її використання. Створити модель сортувальної гірки.

Стисла теоретична довідка

Залізнична бібліотека дозволяє ефективно моделювати та візуалізувати функціонування залізничних вузлів та залізничних транспортних систем будь-якого рівня складності та масштабу. Сортувальні станції, шляхи навантаження/розвантаження великих підприємств, залізничні станції та вокзали, депо, станції метрополітену, шляхи на контейнерних терміналах, рух трамваїв - всі ці задачі можуть бути легко та точно промодельовані за допомогою Залізничної бібліотеки.

Залізнична бібліотека інтегрована з іншими бібліотеками AnyLogic – Бібліотекою моделювання процесів та Пішохідною бібліотекою, що дозволяє з'єднувати залізничні моделі з моделями вантажівок, кранів, кораблів, моделями пасажиропотоків, виробничих та бізнес-процесів тощо.

Нижче наведено опис основних блоків Залізничної бібліотеки, які будуть використовуватися при створенні моделі.

Train Source – створює поїзди, виконує початкове налаштування та поміщає їх у залізничну мережу. Починає будь-яку залізничну діаграму процесу. Підтримує кілька типів розкладів прибуття.

Train Dispose – видаляє поїзди з моделі.

Train Move To – моделює рух поїздів. Може розраховувати маршрут і положення стрілок по ходу руху поїзда за маршрутом. Підтримує функції прискорення та гальмування.

Train Couple – зчіплює два потяги, які "торкаються" один одного, в один.

Train Decouple – розчіплює вагони поїзда, що надходять у блок, і створює з них новий поїзд.

Train Enter – поміщає агента-поїзд, що надходить у блок, на заданий шлях залізничної мережі.

Train Exit – витягує поїзд із залізничної мережі, що надходить в об'єкт, і передає агента-поїзд далі в звичайну діаграму процесу.

RailSettings – задає специфічні установки для залізничної мережі.

Порядок виконання роботи

- ✓ Ознайомитися з основними елементами Залізничної бібліотеки та їх функціями.
- ✓ Створити залізничний вузол.
- ✓ Задати логіку моделі сортувальної гірки.
- ✓ Створити типи вагонів.
- ✓ Реалізувати моделювання сортування вагонів.
- ✓ Виконати звіт.

Методичні вказівки до виконання лабораторної роботи

Створення залізничного вузла

За допомогою моделі необхідно відтворити відправлення вагонів на колію, яку називають стрілочною горловиною. Звідти вагони вирушають через серію стрілок, яку називають стрілочною вулицею, на колію сортування.

Спочатку треба задати топологію залізничної мережі, загальний вигляд якої наведено на рисунку 2.37:

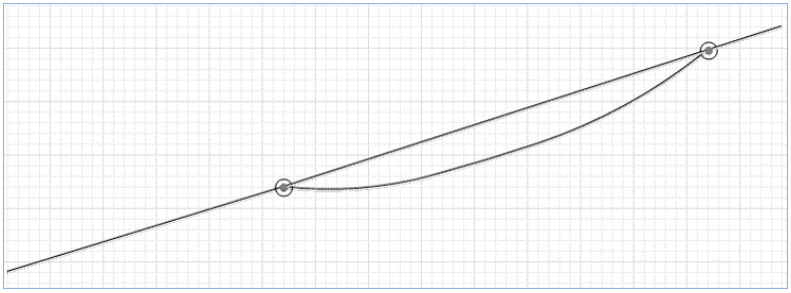


Рисунок 2.37 – Топологія залізничної мережі

Створіть нову модель. Назвіть її Hump Yard та вкажіть хвилини як одиниці модельного часу.

Для створення топології мережі спочатку намалюйте залізничну колію.

Почніть із прямої колії. На цій колії з'являтимуться поїзди, тому ми назвемо його ***trackEntry***. Спочатку виділіть подвійним клацанням елемент ***RailwayTrack*** (залізнична колія) у секції ***SpaceMarkup*** (розмітка простору) палітри ***Rail Library*** (Залізнична бібліотека). Натисніть у будь-якій точці графічного редактора, щоб почати малювати колію. Щоб намалювати прямий сегмент колії, додайте клацанням миші кінцеву точку сегмента. Завершіть малювання подвійним клацанням. Назвіть цю колію ***trackEntry***.

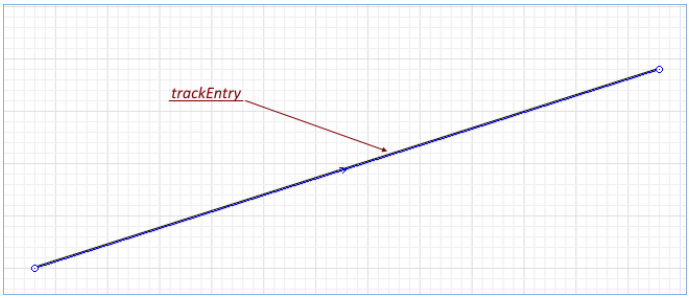


Рисунок 2.38 – Створення простої колії

Якщо це перша залізнична колія моделі, ви побачите повідомлення з пропозицією змінити масштаб моделі на: 2,0 пікселя в 1 метрі. Згоджуємося, оскільки запропонований масштаб часто використовується у залізничних моделях.

Після цього ви побачите намальований пряму залізничну колію, якою рухатиметься склад. Тепер треба намалювати об'їзну колію, якою локомотив під'їжджатиме до складу ззаду для його переміщення на сортувальну гірку. Ця колія буде не пряма, а дугової форми.

Зробіть подвійне клацання по елементу ***RailwayTrack*** (залізнична колія) на панелі Залізнична бібліотека. Потім клацніть на раніше намальовану колію, щоб почати малювати нову. Коло, яке позначає стрілку, з'явиться автоматично.

Послідовно клацаючи мишею, намалюйте кілька сегментів колії,

як на рисунку 2.39. Щоб додати дуговий елемент, не відпускайте кнопку миші після натискання, а перемістіть курсор, утримуючи кнопку миші. При цьому ви побачите, як змінюється радіус дуги. Відпустіть кнопку миші, коли вважаєте, що сегмент набув потрібної форми. Завершіть малювання залізничної колії подвійним клацанням поруч із фігурою першої колії (рис. 2.39). Відразу поставити кінцеву точку на ту саму колію, з якої почали малювати (trackEntry), не вдасться.

Рисунок 2.39 – Створення дугового елементу колії

Перетягніть кінцеву точку (2) на раніше намальовану колію (***trackEntry***), щоб створити стрілку (рис. 2.40).

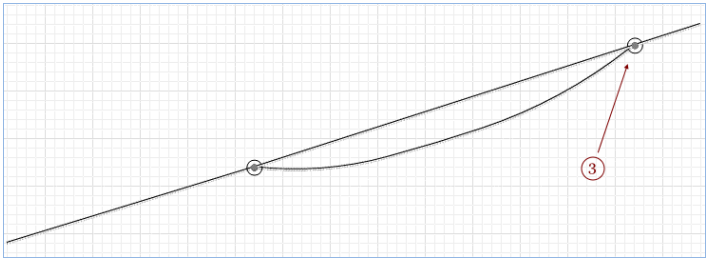


Рисунок 2.40 – Створення стрілки

За необхідності можна відредагувати колію дугової форми. Для цього виділіть шлях, який потрібно відредагувати, клацнувши по ньому. Клацніть правою кнопкою миші виділеним шляхом і виберіть Редагувати напрямні з контекстного меню. Щоб перемістити крайню точку сегмента, переміщуйте квадратну позначку-маніпулятор. Щоб

змінити радіус кривизни, переміщуйте круглу мітку-маніпулятор на кінці пунктирної напрямної лінії.

Залізнична мережа складається з колій та стрілок. Залізничні стрілки з'являються автоматично у вигляді кола у точках з'єднання залізничних гілок із існуючими шляхами. У існуючу точку з'єднання не можна додати додаткові залізничні колії. Два з трьох утворених стрілкою кутів мають бути тупими. Це необхідно для визначення можливих шляхів руху на цій стрілці.

Далі треба модифікувати залізничний вузол (рис. 2.41):

- Назвіть сегмент колії, що знаходиться між залізничними стрілками ***trackArrival***, оскільки він позначає місце прибуття поїзда.
- Перетягніть елемент ***Position on Track*** (Точка залізничної колії) з панелі Залізничної бібліотеки та розташуйте його, як показано на малюнку нижче. Назвіть його ***stopLineEntry***. Він визначатиме місце появи поїздів.
- Додайте елемент ***Position on Track*** (Точка залізничної колії): ***stopLineArrival***. У цій точці поїзд зупиниться, щоб відчепити всі вагони, що дозволить локомотиву продовжити шлях без вагонів.
- Додайте елемент ***Position on Track*** (Точка залізничної колії): ***stopLineHump***. Тут проводитиметься сортування вагонів.

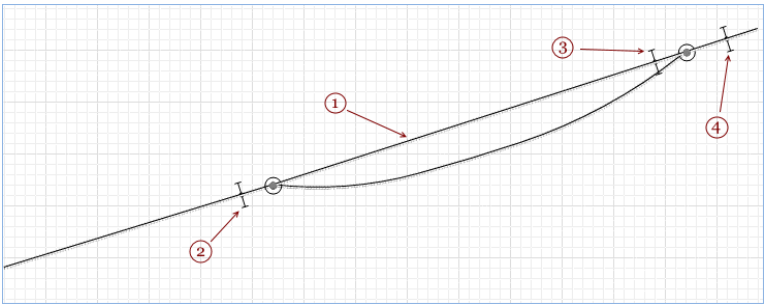


Рисунок 2.41 –Створення точок залізничної колії

Точка залізничної колії є елементом розмітки простору, який використовується, щоб задати точну позицію на залізничному шляху. Це може знадобитися, коли ви ставите:

- позицію на шляху, де з'являється поїзд;
- позицію на шляху, де поїзд має зупинитись.

Задання логіки моделі

Задайте логіку процесу за допомогою блоків Залізничної бібліотеки:

- Додайте наступні блоки на діаграму процесу: ***TrainSource***, ***TrainMoveTo***, ***TrainDecouple***, ***TrainCouple***, ***TrainDispose***. За допомогою цих блоків ми визначимо логіку, згідно з якою пересуватиметься наш поїзд.
- Почніть діаграму з блоку ***TrainSource***, який створює поїзди, потім додайте блок ***TrainMoveTo***, розташувавши його поруч із першим блоком таким чином, щоб вони могли автоматично з'єднатися. Додайте блоки, що залишилися, на діаграму процесу, як показано на рисунку 2.42.

Рисунок 2.42 – Додання основних блоків логіки моделі

- У властивостях блоків ***TrainSource*** та ***trainMoveTo*** вкажіть наступне:

Спочатку ми вкажемо, щоб поїзди з'являлися біля залізничної точки ***stopLineEntry*** кожні 15 хвилин.

TrainSource:

- ***Interarrival time*** (час між прибуттям): *15 хвилин*
- ***# of cars (including loco)*** (кількість вагонів, включаючи локомотив): *11*
- ***Position on track*** (точка залізничної колії): ***stopLineEntry***

Після створення кожен поїзд буде прямувати до залізничної точки ***stopLineArrival***, у якої він буде сповільнюватися, після чого остаточно зупиниться.

TrainMoveTo:

- **TrainMoveTo:**
  Route is: *Calculated automatically from current to target track* (Маршрут: Обчислюється автоматично від поточ-

ного шляху до шляху призначення)

- **Target is**: *A given position on track* (Мета руху: Задана точка шляху)
- **Position on track**: *stopLineArrival* (Точка залізничної колії: stopLineArrival)
- **Finish options**: *Decelerate and stop* (При закінченні руху: Загальмувати та зупинитися)

- Ми також додали блок ***TrainDecouple***, завдання якого відчеплювати задану кількість вагонів від поїзда. У властивостях блоку можна побачити, що за умовчанням він відчіплює один вагон від поїзда, а саме перший (у нашому випадку – це локомотив). Локомотив залишатиме блок через нижній порт ***outDecoupled*** (ми додамо це з'єднання на наступному кроці).
- Блок ***TrainCouple*** потрібний для зчеплення локомотива та вагонів в один склад.

Продовжимо створення діаграми процесу

➢ Додайте три блоки ***TrainMoveTo*** і з'єднайте їх, як показано на рисунку 2.43. За допомогою цих блоків ми визначимо поведінку поїзда на залізничному вузлі, яке включатиме: відчеплення, хід назад, зчеплення з вагонами з подальшим виштовхуванням їх на гірку.

➢ Почніть із приєднання блоку ***TrainMoveTo*** до порту ***outDecoupled*** (якого знаходиться внизу) блоку ***TrainDecouple***, оскільки ми будемо задавати логіку для від'єднаного локомотива.

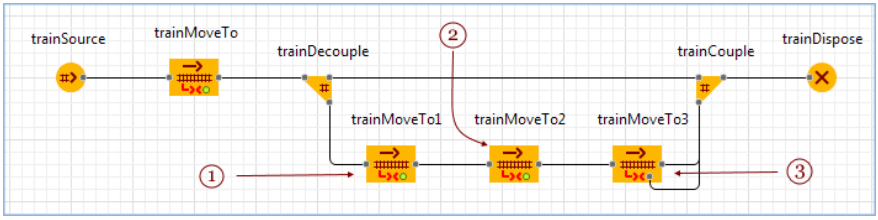


Рисунок 2.43 – Додання блоків руху поїзда

➢ Додайте ще один блок ***TrainMoveTo*** на діаграму процесу (***trainMoveTo1***). Задайте параметри нового блоку, згідно з якими ло-

комотив попрямує до ***stopLineHump*** і зупиниться там.

- **Route is**: *Calculated automatically from current to target track* (Маршрут: Обчислюється автоматично від поточного шляху до шляху призначення
  Target is: *A given position on track* (Мета руху: Задана точка шляху)
  Position on track *: stopLineHump* (Точка залізничної колії: stopLineHump)
  Finish options: *Decelerate and stop* (При закінченні руху: Загальмувати та зупинитися)

➢ Створіть ще два блоки ***TrainMoveTo*** (***trainMoveTo2***, ***trainMoveTo3***) шляхом Ctrl + перетягування (Mac OS: Cmd + перетягування) блоку, який ви щойно редагували.

➢ Змініть властивості блоку ***trainMoveTo2***. В даний момент локомотив відчеплений і нам потрібно, щоб він заштовхав вагони на гору, тобто. йому потрібно повернутись до ***stopLineEntry*** і зробити зчіпку з кінцем поїзда. Так як ми не можемо його розвернути на рейках, локомотив поїде назад повз дорогу ***trackArrival***, на якому стоять вагони:

- **Direction**: *Backward* (Напрямок руху: Назад)
  Route should not contain: *trackArrival* (Маршрут не повинен містити: trackArrival)
- **Position on track**: stopLineEntry (Точка залізничної колії: stopLineEntry)

➢ Зверніть увагу на кількість з'єднань, що ведуть від блоку ***trainMoveTo3*** до блоку ***trainCouple***. Нижній порт блоку ***TrainMoveTo***, на відміну від порту ***out***, що знаходиться праворуч, використовується поїздами, які стикаються з іншими поїздами. У нашому випадку зіштовхування є зчепленням локомотива, що прибуває до точки залізничної колії ***stopLineArrival***, з вагонами, які ми попередньо відчепили від цього локомотива.

➢ Тепер внесіть зміни у властивості блоку ***trainMoveTo3***, щоб цей блок направляв поїзд з його поточного розташування до точки ***stopLineArrival***:

- **Position on track**: *stopLineArrival* (Точка залізничної колії:

stopLineArrival)

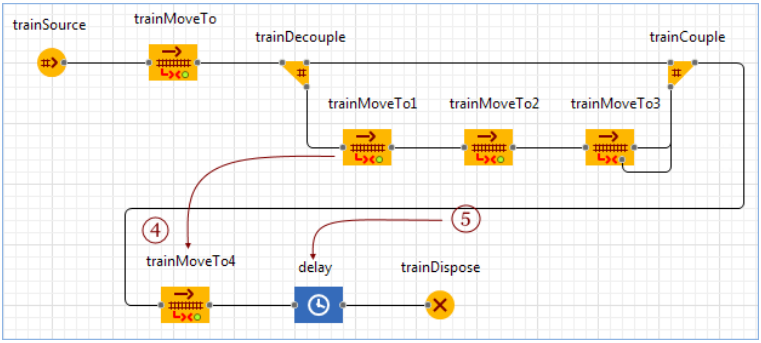


Рисунок 2.44 – Загальний вигляд логіки моделі

Тепер, коли ми змоделювали рух локомотив до кінця складу, потрібно додати ще один блок ***TrainMoveTo*** шляхом Ctrl + перетягування блоку ***trainMoveTo1***. Він моделюватиме процес штовхання вагонів локомотивом до гірки, представленої в нашій моделі елементом Точка залізничного шляху stopLineHump.

- Змініть властивості блоку ***trainMoveTo4***:
  - **Position on track:** *stopLineHump* (Точка залізничної колії: stopLineHump)
  - **Cruise speed (0 for no change):** *5* **meters per second (**Крейсерська швидкість: 5 м/с)

- Додайте блок ***Delay*** з Бібліотеки моделювання процесів, щоб змоделювати затримку. Змініть властивості блоку, задавши час цієї затримки:
  - **Delay time** (Час затримки)**:** 15 секунд.

Можете запустити модель. Якщо ви побачите помилку "The car being created must fully be on one track", значить по дорозі недостатньо місця для розміщення поїзда перед елементом Точка залізничного шляху ***stopLineEntry***. У цьому випадку вам потрібно буде збільшити довжину шляху ***trackEntry*** і відсунути далі елемент Точка залізничної колії ***stopLineEntry*** від початкової точки залізничної колії.

Локомотив під'їжджає до складу, що складається з вагонів, щоб зчепитися із ним (рис. 2.45).

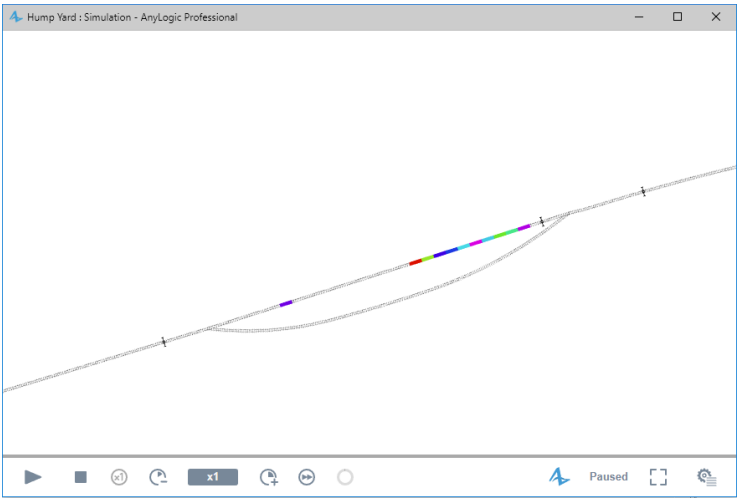


Рисунок 2.45 – Запуск моделі

Також бачимо, як локомотив штовхає вагони на гірку, після чого зупиняється, вагони відчіплюються та сортуються.

Створення типів вагонів

На даному етапі моделювання у вікні працюючої моделі, вагони відрізняються виключно кольором. Отже потрібно створити різні типи вагонів.

AnyLogic надає зручний Майстер створення типу вагона. Просто вкажіть назву нового типу вагона та виберіть фігуру анімації зі списку готових 3D об'єктів.

Задайте типи вагонів:

- Створіть новий Тип вагона, перетягнувши елемент Тип вагона із палітри Залізнична бібліотека на графічну діаграму Main (рис. 2.46).

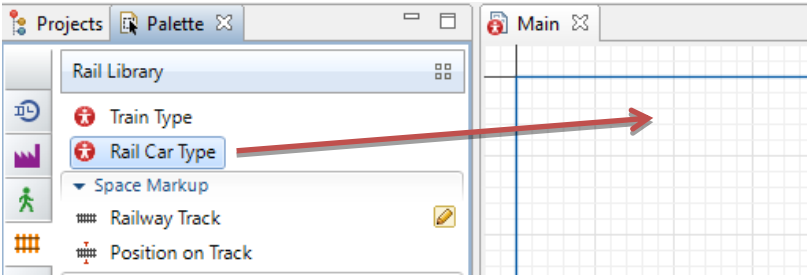


Рисунок 2.46 – Створення нового типу вагонів

- Відкриється вікно Майстра Створення агентів. Вкажіть Ім'я нового типу як Locomotive, залиште опцію Створити новий тип агента "з нуля" вибраної. Натисніть кнопку Далі.
- Виберіть Locomotive 14.1m як фігуру анімації в 3D режимі (рис. 2.47).

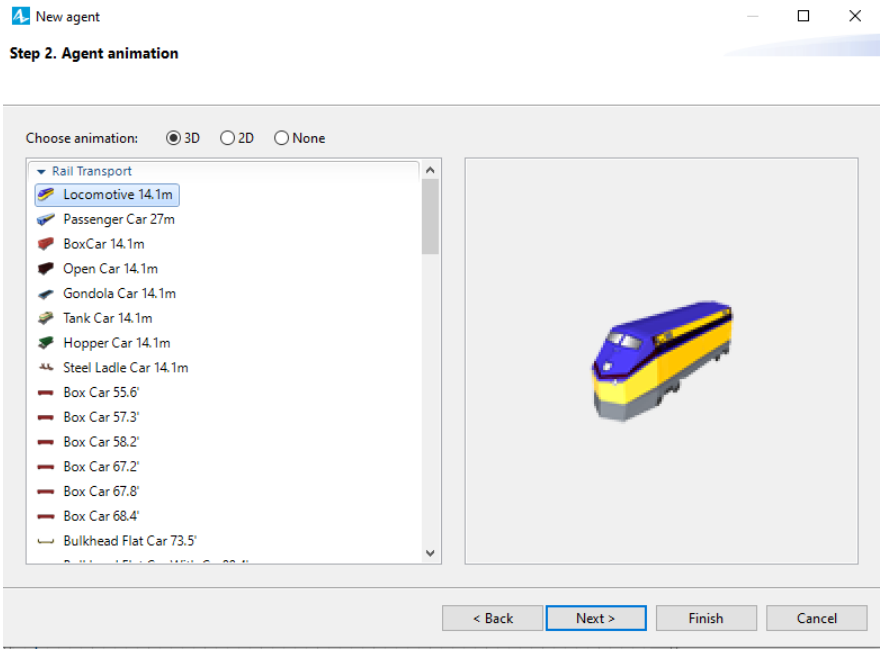


Рисунок 2.47 – Вибір анімації типу вагонів

- Додайте ще 5 типів вагонів так само. Назвіть їх ***BoxCar***, ***OpenCar***, ***GondolaCar***, ***TankCar*** та ***HopperCar***. Виберіть відповідні 3D фігури анімації для кожного типу вагона: Критий вагон, Відкритий вагон, Полувагон, Цистерна, Хопер.
- Так само створіть тип поїзда (перетягнувши елемент Тип поїзда з панелі Залізничної бібліотеки). Назвіть його ***Train***.

Встановлення властивостей блоку ***TrainSource***:

– у полі ***New train*** (Новий поїзд) виберіть ***Train*** (рис 2.48);

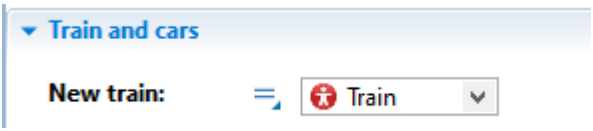


Рисунок 2.48 – Встановлення типу поїзда

– у полі ***New rail car*** (Новий вагон) перейдіть до поля для введення коду, клацнувши по іконці , яка зміниться на , дозволивши написати Java вираз;

– введіть такий вираз:

```
carindex == 0 ? new Locomotive() :
randomlyCreate( OpenCar.class,
BoxCar.class, GondolaCar.class,
HopperCar.class, TankCar.class )
```

Рисунок 2.49 – Встановлення типів вагонів

Тут використовується умовний оператор Java у тому, щоб створювати вагони різних типів. Коли блок ***TrainSource*** створює новий поїзд, він послідовно викликає вираз, заданий у полі Новий вагон, для кожного вагона поїзда, що створюється. Локальна змінна ***carindex*** зберігає індекс щойно створеного вагона. Наш вираз перевіряє виконання умови ***carindex==0***. Якщо індекс дорівнює нулю (тобто ми маємо справу з першим вагоном складу), то буде виконано вираз, написаний після знака ?

Вираз ***new Locomotive()*** створить вагон типу ***Locomotive***.

Якщо задану логічну умову поверне false, це означатиме, що за-

раз створюється не перший вагон. У цьому випадку ми скористаємося функцією ***randomlyCreate***, щоб випадково вибрати один з типів вантажних вагонів, заданих нами на минулому кроці. Список типів вагонів ми передаємо функції як її аргументи.

В результаті, у нас буде поїзд з локомотивом як перший вагон, за яким слідують десять вагонів створених нами типів.

Створіть 3D анімацію моделі. Додайте 3D вікно з панелі Презентація та помістіть його під діаграмою процесу та анімацією.

Запустіть модель. Щоб перейти до 3D анімації в запущеній моделі, відкрийте панель налагодження, натиснувши кнопку Показати/приховати панель розробника в правому кутку панелі керування. У панелі налагодження, розкрийте список Вибрати область і показати і виберіть опцію [window3d].

Тепер ви бачите поїзд, який починається з локомотива, за яким йдуть десять вагонів інших типів.

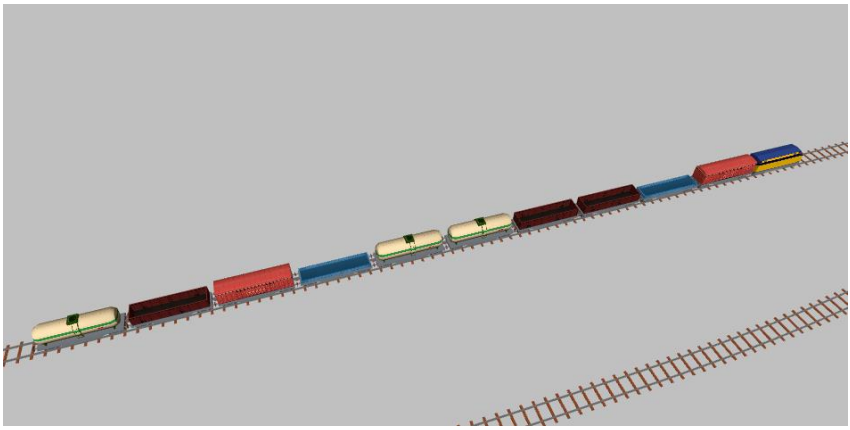


Рисунок 2.50 –3D анімація моделі

Моделювання сортування вагонів

Наступним етапом промоделюємо процес сортування вагонів та їх розподілу на різні шляхи відповідно до їх типу.

– Намалюйте додатково кілька залізничних колій, як показано на рисунку 2.51. Цими коліями відбуватиметься накопичення вагонів, на кожній з них збиратимуться вагони певного типу.

– Надайте коліям імена: trackN1 ... trackN6.

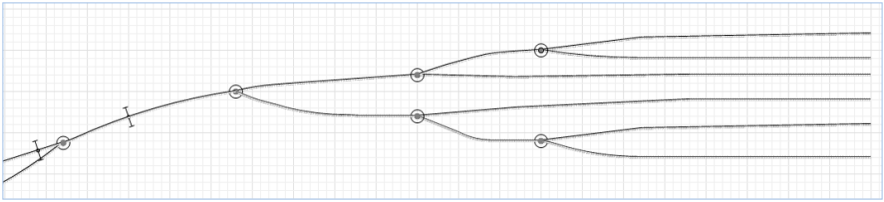


Рисунок 2.51 – Схема залізничних колій

– Додайте шість елементів ***Position on Track*** (точка залізничної колії) на кожну колію призначення. Назвіть їх s***toplineN1***, ... ***stoplinenN6*** (рис. 2.52). Таким чином, ми задаємо лінію зупинки вагонів для кожної колії.

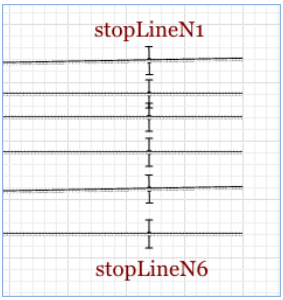


Рисунок 2.52 – Встановлення лінії зупинки вагонів

Задайте логіку діаграми процесу.

Поїзд, що прибуває на сортувальну станцію, може містити кілька вагонів одного й того самого типу, розташованих один за одним. Такі вагони можуть бути відправлені на свою колію усі разом.

- Додайте функцію ***Function***, яка підраховуватиме кількість

вагонів одного типу в поїзді, що знаходиться на гірці. Назвіть її ***carsOfSameType*** і змініть її властивості таким чином (рис. 2.53):

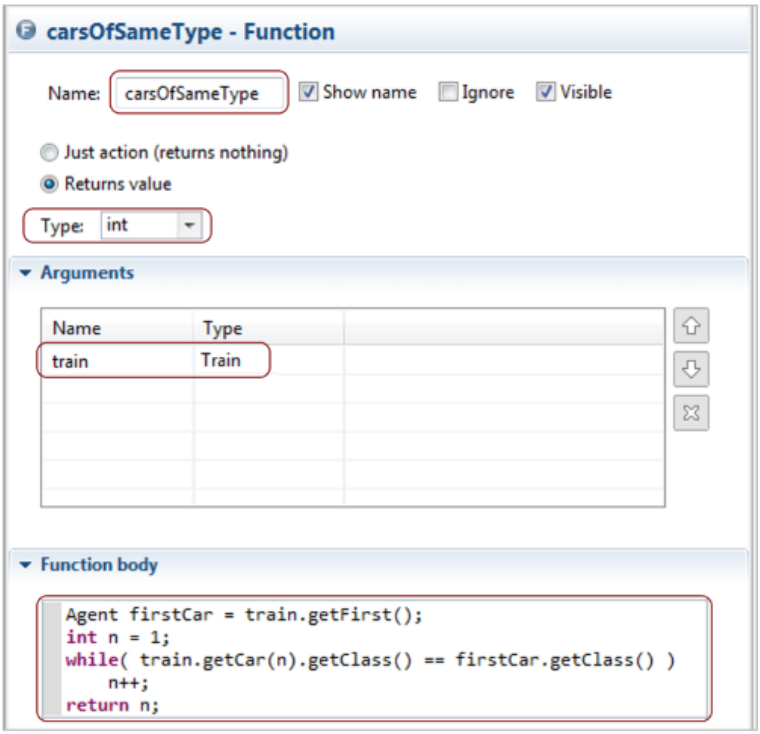


Рисунок 2.53 – Створення функції розрахунку кількості вагонів

- Створіть ще одну функцію і назвіть її ***departurePointOnTrack***. Вона визначатиме колію призначення для вагонів у сортувальному парку. Ця функція використовує поїзд як аргумент, потім перевіряє тип першого вагона поїзда. Ми знаходимо тип вагона за допомогою оператора Java ***instanceof***. Він повертає ***true*** якщо об'єкт є одиницею заданого класу. У результаті функція повертає елемент ***Position on track*** (точка залізничної колії), який залежить від типу вагона.
- Внесіть у властивості зміни, що вказані на рис. 2.54.

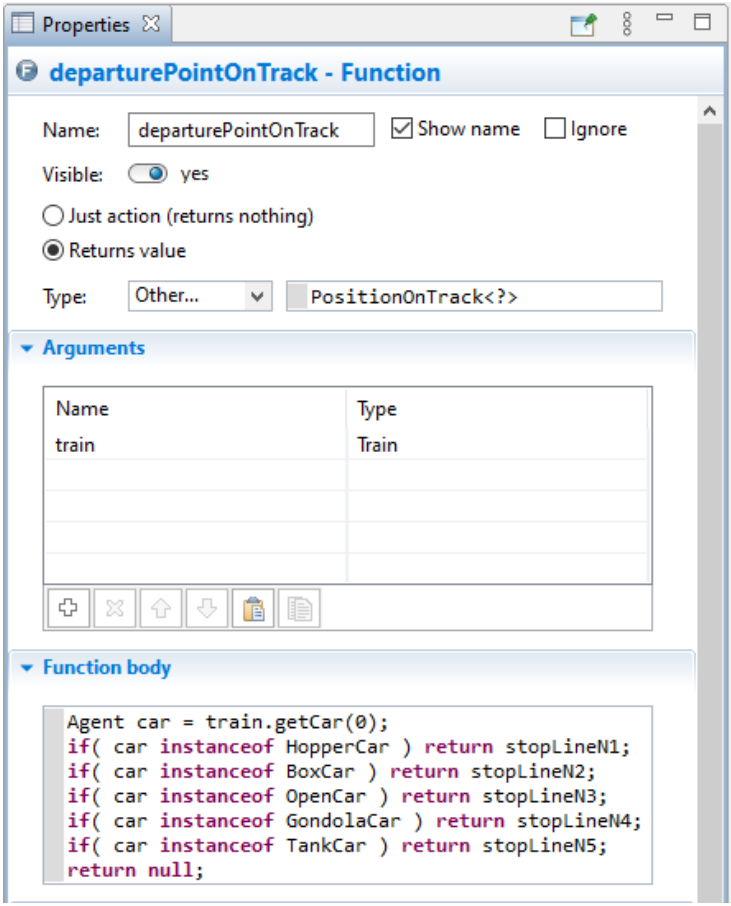


Рисунок 2.54 – Створення функції визначатиме колію призначення

- Перетягніть елемент ***Variable*** (змінна) у графічну діаграму з панелі ***Agent***. Ця змінна міститиме посилання на колію, де необхідна кількість вагонів одного типу готова залишити станцію як новий поїзд.

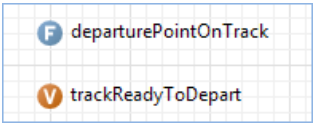


Рисунок 2.55 –Створення змінної

- Налаштуйте змінну:
  - назвіть її ***trackReadyToDepart;***
  - виберіть ***Інший*** у списку ***Тип***. У правому полі виберіть ***RailwayTrack***;
  - додайте блоки у діаграму процесу (рис. 2.56).

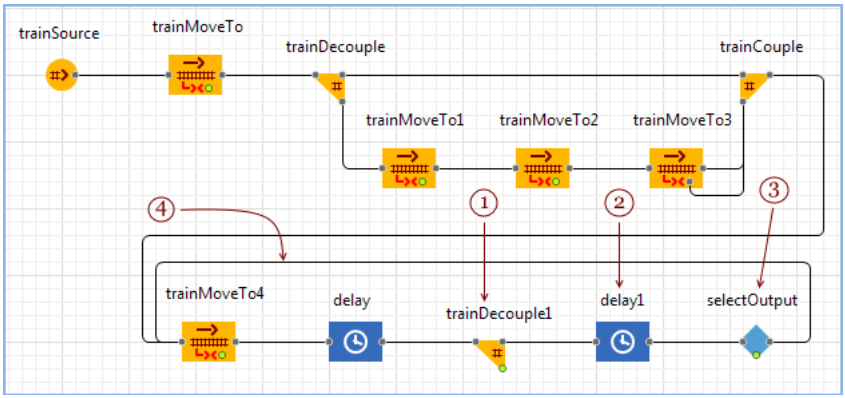


Рисунок 2.56 –Діаграма процесу

✓ Блок ***TrainDecouple*** розчіплює вагони, коли поїзд знаходиться на гірці. Потяг може містити кілька вагонів одного типу, що йдуть один за одним. Такі вагони відкочуються на свій шлях призначення одразу. Ми визначаємо, скільки вагонів потрібно відкотити, функцією ***carsOfSameType()***, створеною на попередньому кроці. Колія призначення задається створеною нами функцією ***departurePointOnTrack()***.

Вкажіть наступне у властивостях блоку ***TrainDecouple***:

- кількість вагонів для відчеплення: ***carsOfSameType(train);***
- новий поїзд: ***Train;***

✓ Є коротка затримка, перш ніж потяг продовжить розчіплюватися. Реалізуємо її за допомогою блоку ***Delay1***. У його властивостях вкажіть час затримки: 5 секунд.

✓ Тепер ми повинні визначити, чи потяг ще містить вагони, які можна відчепити. Цим займається блок ***SelectOutput***. Ми перевіряємо, чи потяг містить ще хоч один вагон (пам'ятайте, що і сам локомотив вважається вагоном). Якщо це так, то поїзд знову надходить до

блоку ***TrainDecouple***. Цикл повторюється доти, доки поїзд не містить лише локомотив, готовий покинути станцію.

Вкажіть наступне у властивостях блоку ***SelectOutput***:

Select True output: If condition is true (вихід true вибирається: у разі виконання умови

Condition *(умова)*: agent.size() > 1

✓ Не забудьте приєднати правий порт блоку ***SelectOutput*** до вхідного порту ***trainMoveTo4***, щоб повернути поїзд на гірку.

Промоделюйте, як локомотив, що привіз різнорідні вагони, залишає станцію (рис. 2.57).

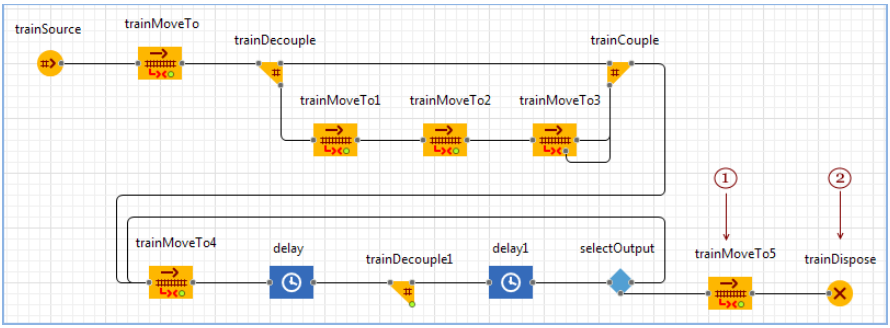


Рисунок 2.57 –Моделювання руху локомотиву

- Ми знову використовуватимемо блок ***TrainMoveTo***, щоб промоделювати рух локомотива. Як пункт призначення вказуємо нижній колію (***trackN6***). Вкажіть наступне у властивостях блоку ***TrainMoveTo***:
  - ***Route is:*** Calculated automatically… (Маршрут: Обчислюється автоматично)
  - ***Target is:*** A given offset on a track (Ціль руху: Задане зміщення на колії)
  - ***Railway track:*** trackN6 (Колія: trackN6)

- Потім локомотив видаляється з моделі блоком ***TrainDispose***.

Задайте логіку скочування вагонів на колії сортувального парку

(рис. 2.58) Промоделюйте скочування вагонів на колії призначення в сортувальному парку під дією сили земного тяжіння (колії, де вагони сортуються).

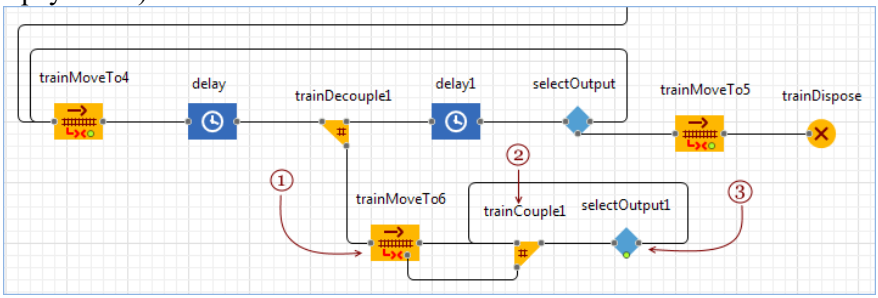


Рисунок 2.58 – Моделювання скочування вагонів

➢ Рух, як завжди, моделюється блоком ***TrainMoveTo6***. Потяг містить окремі вагони чи зчеплені вагони одного типу. Призначення визначається функцією ***departurePointOnTrack()***, яку ми вказали раніше. Вкажіть наступне у властивостях блоку ***TrainMoveTo6***:

- ***Route is:*** *Calculated automatically…* (Маршрут: Обчислюється автоматично...)
- ***Target is:*** *A given position on track* (Ціль руху: Задана точка колії)
- ***Position on track*** (Точка залізничної колії)*: departurePointOnTrack(train)*
- ***Finish options****: Decelerate and stop* (При закінченні руху: Загальмувати та зупинитися)

• Якщо на шляху призначення вже є якісь вагони, то вони комбінуються разом з вагонами, які щойно прибули в блоці ***TrainCouple***.

• Вагони продовжують з'єднуватися, доки не буде досягнуто певної кількості вагонів на колії (тоді локомотив приїжджає туди і забирає вагони зі станції). Блок ***SelectOutput*** перевіряє, коли збирання вагонів можна припинити. Ми вказуємо, що кількість зібраних вагонів має бути меншою за 8. Вкажіть наступне у властивостях блоку ***SelectOutput***:

- ***Select True output***: *If condition is true* (Вихід true вибирається: під час виконання умови)
- ***Condition*** (Умова): agent.size() < 8

Додайте логіку відправлення готового поїзда, зібраного з вагонів одного типу, зі станції (рис. 2.59)

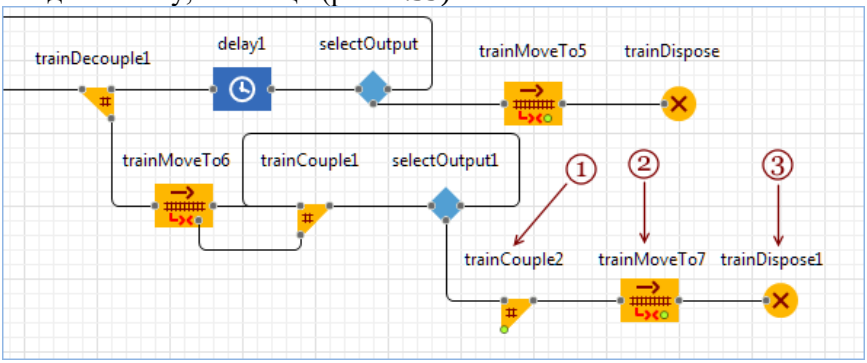


Рисунок 2.59 –Моделювання відправлення готового поїзда

- Спочатку ми зчіплюємо вагони з локомотивом за допомогою блоку ***TrainCouple*** (логіку поведінки локомотива ми поставимо на наступному кроці).
- Потім поїзд залишає станцію (це моделюється блок ***TrainMoveTo7***). Вкажіть наступне у його властивостях:
  - ***Route is***: *not specified (train will follow switches)* (Маршрут: Не заданий (поїзд слідуватиме згідно зі стрілками))
  - ***Target is:*** *not specified* (Ціль руху: Не задана)

- Зрештою ми видаляємо поїзд із моделі блоком ***TrainDispose***.

Додайте логіку локомотива, що забирає готові до відправки вагони. Треба змоделювати, як локомотиви забирають зі станції готові поїзди, що складаються з вагонів одного типу (рис. 2.60).

✓ Спочатку давайте додамо блок ***TrainSource***, що моделює прибуття локомотива. Потяг складається лише з одного вагона (сам локомотив). Виберіть ручний режим створення поїздів – викликами функції ***inject()***, оскільки цей блок буде створювати поїзди тільки тоді, коли новий поїзд готовий відправитися зі станції. Локомотив повинен з'явитися на шляху, заданому змінною ***trackReadyToDepart***.

Вкажіть наступне у властивостях блоку ***TrainSource***:

– ***Name*** (Ім'я): *newLoco*

– ***Arrivals defined by***: *Call of inject() function* Поїзди прибувають згідно: Викликам методу inject()

– ***# of cars (including loco)*** Кількість вагонів (включаючи локомотив): *1*

– ***Entry point defined as:*** offset on the track Точка входу задається як: Зміщення по дорозі

– ***Railway track*** (: *trackReadyToDepart* (Switch to the code field by clicking the icon) Шлях: trackReadyToDepart

– ***Offset from***: *Beginning of the track* Зсув від: початку шляху

– ***Offset of 1st car***: *tracklength - 15* **meters** Зміщення першого вагона: tracklength – 15 метрів

– ***New train*** Новий поїзд: *Train*

– ***New rail car*** Новий вагон: Locomotive

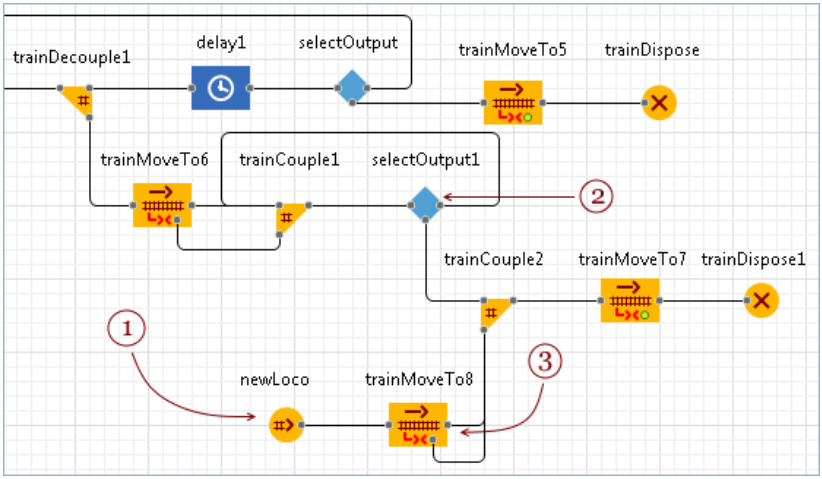


Рисунок 2.60 – Моделювання відправлення готового поїзда

✓ Цей блок перевіряє, чи є достатня кількість вагонів одного типу по дорозі чи ще немає. Вкажіть наступне у властивості При виході (***false***) секції ***Дії*** блоку ***SelectOutput***:

– *trackReadyToDepart = agent.getTrack(true);*

– *newLoco.inject();*

При виході (***false***) виконується, коли з'являється необхідна кількість вагонів. Рядок ***trackReadyToDepart = agent.getTrack(true)***; отримує поточну колію, на якій зібралися готові до відправки вагони, і записує його в локальну змінну ***trackReadyToDepart***. Рядок ***newLoco.inject()***; генерує новий локомотив у блоці ***newLoco***. Коли локомотив створено, він йде до блоку, який створюємо далі.

✓ Блок ***TrainMoveTo8*** моделює рух локомотива до готових до відправки вагонів. Зверніть увагу на напрямок його руху – *Назад* (локомотив рухається у напрямку, зворотному напрямку шляху, праворуч наліво). Потім локомотив зчіпляється з вагонами і поїзд, що вийшов, проходить ту частину діаграми процесу, яка моделює відправлення поїзда з сортувальної гірки. Вкажіть наступне у властивостях блоку ***TrainMoveTo8***:

– ***Direction***: *Backward* (Напрямок руху: Назад)
– ***Route is***: *not specified (train will follow switches)* (Маршрут: Не заданий (поїзд слідуватиме згідно зі стрілками))
– ***Target is***: *not specified* (Ціль руху: Не задана)
– ***Cruise speed:*** *10* meters per second (Крейсерська швидкість: 10 м/с)

У підсумку, ваша діаграма має виглядати так (рис. 2.61):

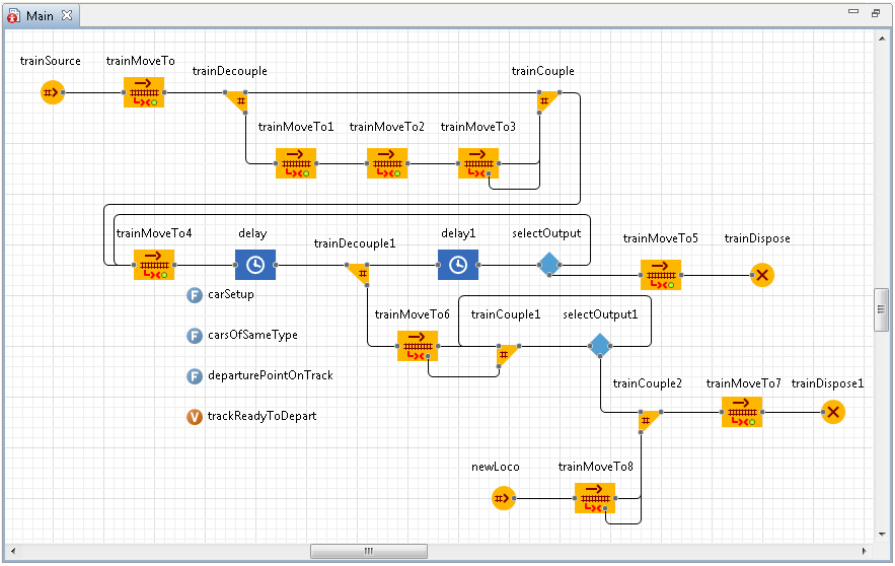


Рисунок 2.61 – Загальний вигляд діаграми процесу

Запустіть модель. Ви побачите, як вагони сортуються за типом. Коли набирається достатня кількість вагонів одного типу, локомотив забирає їх із сортувальної станції (рис. 2.62).

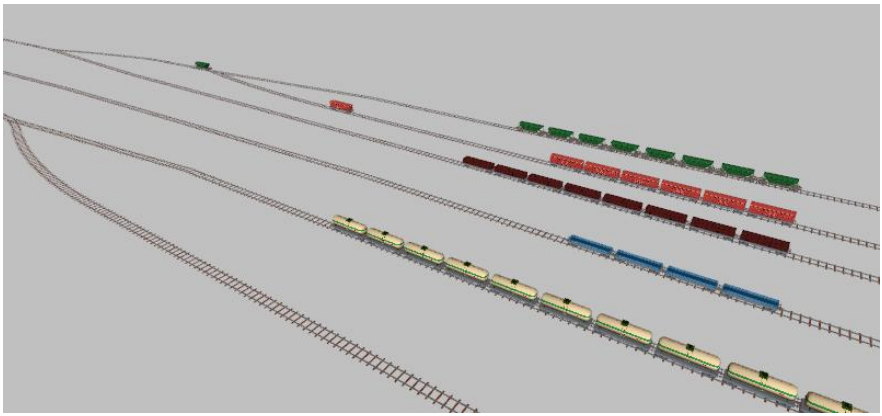


Рисунок 2.62 –3D анімація моделі

2.4 Лабораторна робота № 4

Моделювання руху автомобілів на перехресті

Мета роботи – ознайомитися основними елементами Бібліотеки дорожнього руху Anylogic та особливостями її використання. Навчитися моделювати рух автомобілів на перехресті.

Стисла теоретична довідка

Бібліотека дорожнього руху дозволяє моделювати та візуалізувати рух потоків машин. Бібліотека підтримує деталізоване, але водночас високоефективне моделювання руху машин фізично. З її допомогою ви можете промоделювати як рух машин на автомагістралі, так і вуличний трафік машин, транспортування на виробництві, паркування та будь-які інші системи з машинами, дорогами та дорожніми смугами.

Бібліотека дорожнього руху сумісна з іншими бібліотеками AnyLogic – Бібліотекою моделювання процесів, Пішохідною бібліотекою та Залізничною бібліотекою. Ви можете легко поєднувати моделі руху автомобілів з моделями вантажівок, кранів, кораблів, поїздів, пасажиропотоків, виробничих та бізнес-процесів тощо.

Бібліотека дорожнього руху включає сім блоків, за допомогою яких ви можете задати сценарії руху потоків машин.

CarSource - створює автомобілі та намагається помістити їх у вказане місце дорожньої мережі (на вказану дорогу чи паркування).

CarDispose - видаляє машини з моделі. Видаляти автомобілі потрібно саме за допомогою блоку ***CarDispose***, а не блоків ***Sink*** чи ***Exit***.

CarMoveTo - блок, який керує рухом автомобіля. Автомобіль може їхати лише тоді, коли він знаходиться в блоці ***CarMoveTo***. Автомобіль намагається розрахувати шлях від свого поточного місця до вказаного призначення, коли надходить у блок ***CarMoveTo***. Як місце призначення можуть виступати: дорога, парковка, автобусна зупинка або стоп-лінія.

CarEnter - приймає агента-машину та намагається помістити його як автомобіль у вказане місце дорожньої мережі (на вказану дорогу чи паркування). Блок ***CarEnter*** використовується разом із блоком

CarExit для моделювання частини руху автомобіля на вищому рівні абстракції, а не на детальному, фізичному рівні.

CarExit - вилучає автомобіль, що надходить в об'єкт, з дорожньої мережі і передає його як агента далі в звичайну діаграму процесу, яка може бути складена з блоків Бібліотеки моделювання процесів. Зазвичай використовується у зв'язці з блоком ***CarEnter*** для моделювання якихось процесів (наприклад, рух машини на певній ділянці) на вищому рівні абстракції, а не на детальному, фізичному рівні.

TrafficLight - моделює світлофор, що управляє рухом машин на перехресті або біля якоїсь стоп-лінії.

RoadNetworkDescriptor - опціональний блок. За допомогою цього блоку розробники отримують доступ до керування всіма транспортними засобами, що знаходяться в одній дорожній мережі. Блок дозволяє задавати дії, які будуть виконуватись при додаванні автомобіля в дорожню мережу, в'їзді на дорогу, зупинці автомобіля, зміні смуги тощо. Крім того, за допомогою даного блоку можна ввімкнути відображення пробок на дорогах.

Порядок виконання роботи

- Ознайомитися з основними елементами Бібліотеки дорожнього руху та їх функціями.
- Побудувати дорогу користуючись супутниковим знімком місцевості та створити початкову діаграму процесу.
- Додати 3D анімацію та створити тип автомобілів.
- Змоделювати паркування.
- Додати на модель рух автобусів.
- Додати світлофори для регулювання руху на перехресті.
- Виконати звіт.

Методичні вказівки до виконання лабораторної роботи

Ми створюватимемо модель, взявши за основу супутниковий знімок місцевості. Перехрестя, що розглядається у прикладі наведене на рис. 2.63. На ньому чітко видно всі аспекти цієї ділянки дорожньої мережі. Обидві дороги - з двостороннім рухом і мають по одній смузі для руху в кожному напрямку.

Рисунок 2.63 – Супутниковий вигляд місцевості

Створення дороги

Створимо найпростішу модель, яка моделюватиме рух автомобілів у північному напрямку. Для цього натисніть кнопку панелі інструментів ***Створити***. Дайте назву новій моделі ***Road Traffic***. Виберіть секунди як одиниці модельного часу.

Треба додати супутниковий знімок місцевості, в якій знаходиться досліджуване перехрестя, на графічну діаграму моделі. Для цього відкрийте панель ***Presentation*** (презентація) на панелі ***Pallete*** (палітра). Та перетягніть елемент ***Image*** (зображення) з панелі ***Presentation*** на графічну діаграму (рис. 2.64).

Тепер потрібно вибрати зображення, яке ми будемо використовувати. Діалог для вибору файлу з'явиться автоматично. Відкрийте папку, в яку ви зберегли файл зображення, виберіть його та натисніть ***Відкрити***.

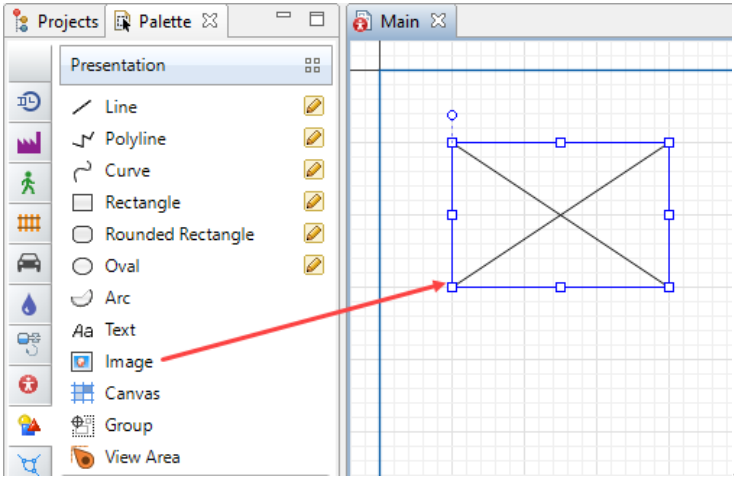


Рисунок 2.64 – Додання супутникового знімку місцевості

Створення дороги

Намалюємо дорогу, розмістивши її чітко поверх тієї, що на знімку. Дорожні мережі в AnyLogic задаються таким чином: ви малюєте дороги, перехрестя і при необхідності додаєте інші елементи розмітки простору (автобусну зупинку, стоп-лінію та паркування).

Створіть дорогу:

- перш ніж намалювати дорогу, слід відключити сітку у графічному редакторі. Натисніть кнопку панелі інструментів ***Увімкнути/Вимкнути сітку***;
- на панелі Палітра виберіть панель ***Road Traffic Library*** (Бібліотека дорожнього руху) (рис 2.65);
- зробіть подвійне клацання по елементу ***Road*** (дорога) в розділі ***Space Markup*** (розмітка простору) панелі Бібліотека дорожнього руху. Значок елемента зміниться на .;
- натисніть мишею у графічному редакторі, щоб поставити першу точку дороги. Щоб намалювати прямий сегмент, клацніть мишею там, де хочете розмістити кінцеву точку сегмента. Щоб намалювати дуговий сегмент дороги, затисніть ліву кнопку миші у точці закінчення дугового сегмента та початку наступного прямого сегмента та переміщуйте курсор з натиснутою лівою кнопкою миші доти, доки сегмент не набуде необхідної форми (рис. 2.66);

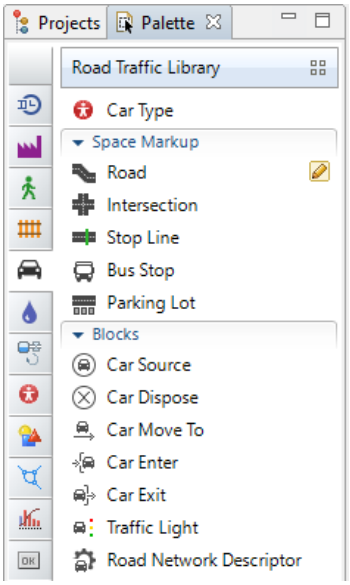


Рисунок 2.65 – Панель Road Traffic Library (Бібліотека дорожнього руху)

Рисунок 2.66 – Побудова головної дороги

- щоб завершити, додайте останню точку дороги подвійним клацанням миші;
- якщо це перша дорога моделі, ви побачите повідомлення з пропозицією змінити масштаб моделі на: 4 пікселі в 1 метрі. Ми радимо погодитись, оскільки пропонований масштаб часто використовується в моделях дорожнього руху;
- налаштуйте атрибути дороги на панелі ***Properties*** (властивості). За замовчуванням дороги створюються з двостороннім рухом, при цьому кожен напрямок дороги (основний та зустрічний) містить по дві смуги руху. Наша дорога містить одну смугу основного руху та одну смугу зустрічного руху, тому необхідно задати значення параметрів (рис 2.67);

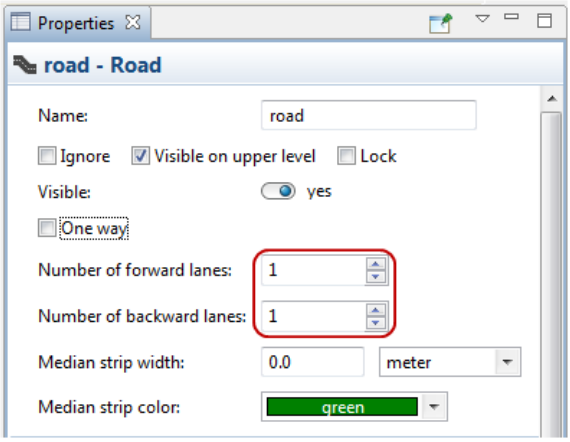


Рисунок 2.67 – Встановлення властивостей дороги

- Якщо створена дорога ширша або вужча чим треба, потрібно налаштувати масштаб моделі. При необхідності можете перетягнути полотно трохи нижче, щоб над віссю X з'явився елемент ***Масштаб***, у властивостях якого можна буде вказати масштаб дороги. Встановимо, щоб лінійка масштабу відповідала 35 метрам. Ширина дороги зміниться відповідно до нового значення масштабу моделі. У результаті у вас повинна вийти дорога, подібна до тієї, що на рис 2.68.

Рисунок 2.68 – Загальний вигляд головної дороги

Створення діаграми процесу

Тепер треба задати логіку процес руху транспорту, створивши діаграму з блоків Бібліотеки дорожнього руху:

– відкрийте палітру Бібліотека дорожнього руху та додайте блоки з цієї палітри на діаграму, з'єднавши їх, як показано на рис. 2.69;

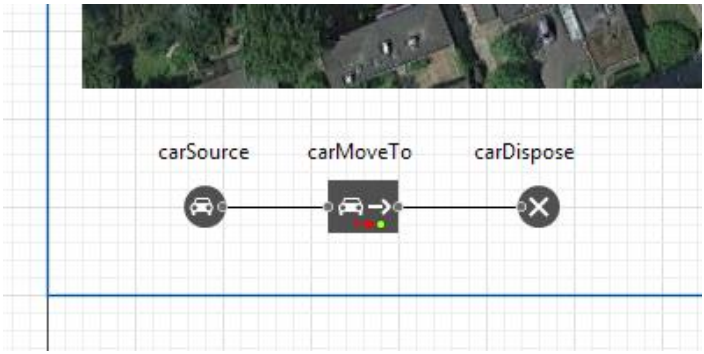


Рисунок 2.69 – Створення діаграми процесу

– у властивостях блоку ***carSource*** вкажіть, як часто повинні прибувати автомобілі. У списку ***Road*** (дорога), що розкривається, виберіть ***road***, це дорога, яку ми щойно створили. Таким чином ми вказуємо блоку ***carSource*** дорогу, на яку варто помістити створювані автомобілі (рис. 2.70);

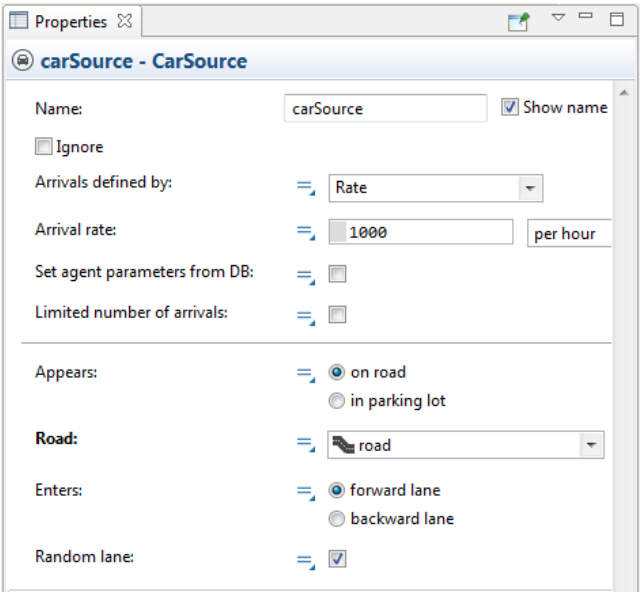


Рисунок 2.70 – Властивості блоку ***carSource***

- змініть властивості блоку ***carMoveTo*** (рис. 2.71). Залишіть задану за замовчуванням опцію ***Moves to: road*** (мета руху: дорога) та виберіть ім'я нашої дороги ***road*** у списку ***Road*** (дорога). Цей блок моделюватиме рух автомобілів вказаною дорогою до кінця смуги основного руху (задається параметром ***Destination***).

Створення 3D-анімації

Перетягніть елемент ***3D Вікно*** з розділу 3D панелі ***Презентація*** до графічного редактора. У графічному редакторі з'явиться зафарбована сірим кольором область. Помістіть цю область туди, де хочете бачити 3D анімацію під час запуску моделі. У нашій моделі супутниковий знімок займає більшу частину простору вікна, тому ми розташуємо 3D вікно під знімком.

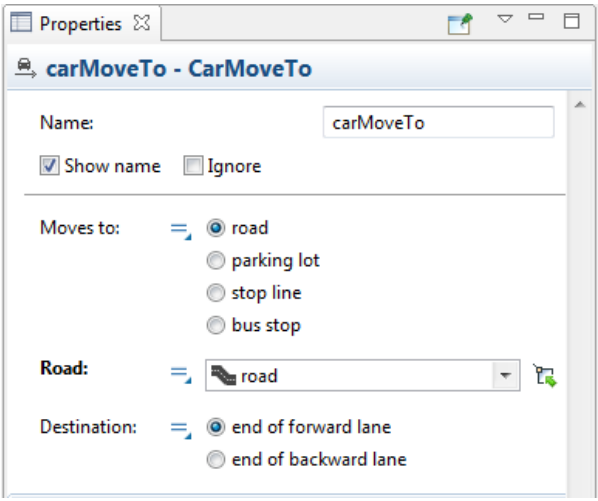


Рисунок 2.71 – Властивості блоку ***carMoveTo***

Створіть новий тип агента:

- відкрийте панель бібліотеки дорожнього руху;
- перетягніть елемент ***Car type*** (тип автомобіля) до графічного редактора;
- відкриється діалогове вікно майстра створення агентів. Залишіть задане за замовчуванням ім'я нового типу ***Car***;
- на наступній сторінці майстра створення типу агента, оберіть анімацію агенту – автомобіль.

Налаштуйте діаграму процесу, щоб використати новий тип автомобіля:

- на діаграмі ***Main***, виділіть блок ***carSource*** у графічному редакторі;
- розкрийте розділ ***Car*** (автомобіль) на панелі ***Properties*** (властивості) та виберіть ***Car*** зі списку ***New car*** (новий автомобіль);
- запустіть модель і перейдіть в режим 3D, щоб побачити 3D анімацію автомобілів, що рухаються дорогою (рис 2.72).

Рисунок 2.72 – 3D анімація моделі

Моделювання перехрестя

Зараз машини рухаються в моделі головною дорогою, причому тільки в одному напрямку, з півдня на північ.

Тепер створимо перехрестя, намалювавши вулицю, що примикає до головної зі сходу. Після цього треба додати нові блоки в діаграму процесу, які моделюватимуть рух машин по обох дорогах у всіх напрямках.

Намалюйте перехрестя. Перехрестя AnyLogic створюються автоматично при з'єднанні доріг. Намалюйте нову вулицю, що примикає до головної дороги зі сходу.

Додайте початкову точку дороги, клацнувши по розділовій смузі у правій частині рисунка.

Додайте кінцеву точку дороги. Щоб правильно з'єднати дороги до Т-подібного перехрестя, зробіть подвійне клацання миші, коли побачите зелену точку на розділовій смузі вулиці, як показано на рисунку 2.73. Буде створено тристороннє перехрестя.

У властивостях щойно намальованої дороги, встановіть такі налаштування: ***Number of forward lanes*** (кількість смуг основного руху) і ***Number of backward lanes*** (кількість смуг зустрічного руху) цієї дороги рівним 1.

Щоб надалі нам було легше посилатися на цю дорогу у параметрах блоків діаграми процесу, змініть ім'я цієї дороги на ***roadEast***.

Рисунок 2.73 – Створення перехрестя

Якщо ви зараз запустите модель, то побачите, що машини їдуть не до кінця вулиці, а лише доїжджають до перехрестя і зникають. Це викликано тим, що при приєднанні нової дороги до вже існуючої дороги створюється новий елемент розмітки простору AnyLogic - перехрестя, що ділить дорогу на дві окремі дороги.

У нашій діаграмі процесу блок ***carSource*** створює машини на початку дороги ***road***, після чого блок ***carMoveTo*** моделює рух машини до кінця тієї ж самої дороги, яка тепер закінчується біля перехрестя (після перехрестя слідує вже інша дорога, названа ***road2***).

Перейменуйте обидві ці дороги на ***roadNorth***, ***roadSouth*** (рис 2.74), щоб надалі було простіше посилатися на них у блоках діаграми процесу.

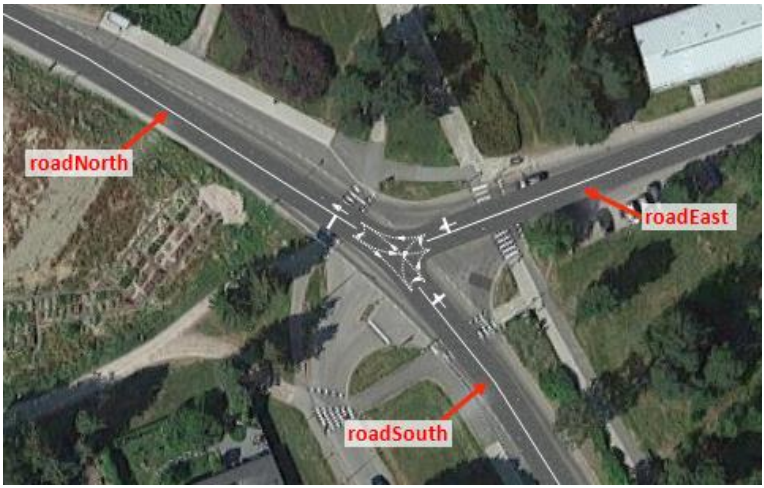


Рисунок 2.74 – Зміна назв доріг

Оскільки блоки діаграми процесу ***carSource*** і ***carMoveTo*** все ще посилаються на стару назву дороги, потрібно внести відповідні зміни до параметрів цих блоків.

✓ Виділіть блок ***carSource***. Змініть його ім'я на ***carSourceS*** (щоб підкреслити, що цей блок генерує машини, що з'являються з півдня).

✓ У полі Дорога цього блоку ви побачите стару назву дороги. Виберіть нове ім'я дороги ***roadSouth*** з списку.

✓ Аналогічно змініть властивості блоку ***carMoveTo***. Назвіть його ***carMoveToN*** (він буде моделювати рух машин до кінця дороги, що йде на північ) і виберіть в полі ***Road*** (дорога): ***roadNorth***.

Наступним кроком буде додавання нових блоків у діаграму процесу, щоб автомобілі почали їздити всіма дорогами, причому в кожному дозволеному напрямку.

Додайте блоки, що моделюють рух тією самою дорогою, але з півночі на південь. Додайте ще один блок ***CarSource*** та ***CarMoveTo***. З'єднайте їх, як показано на рис 2.75.

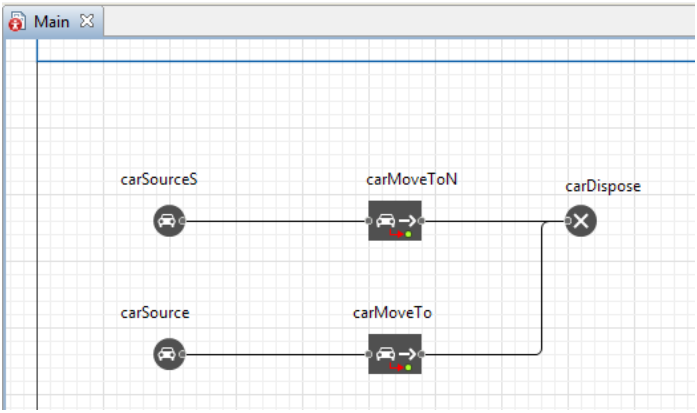


Рисунок 2.75 – Додання нових блоків до діаграми

Перейменуйте блок ***carSource*** на ***carSourceN*** (цей блок генерує машини, що з'являються з півночі).

У полі ***Road*** (дорога) виберіть дорогу, на якій будуть з'являтися автомобілі, що створюються цим блоком: ***roadNorth***.

Оскільки нам потрібно, щоб автомобілі з'являлися на початку смуги зустрічного руху цієї дороги (біля її північного кінця), виберіть у параметрі ***Enters*** (поміщається на смугу): ***backward lane*** (зустрічного руху).

Встановіть у полі ***New car*** (новий автомобіль): ***Car***.

Аналогічно змініть властивості блоку ***carMoveTo***. Назвіть його ***carMoveToS*** (він моделюватиме рух машин на південь), виберіть у полі Дорога: ***roadSouth***, а також виберіть опцію ***Destination: end of backward lane*** (доїхати до кінця смуги: зустрічного руху).

Якщо ви запустите модель, то побачите, що тепер машини рухаються по дорозі в обох напрямках.

Тепер давайте промоделюємо рух машин, що їдуть по зі Сходу. На перехресті частина цих машин (приблизно половина) прямуватиме на південь, а частина, що залишилася, - на північ.

Додайте нові блоки у діаграму процесу:

✓ додайте ще один блок ***CarSource***, назвіть його ***carSourceE***, а в полі Дорога виберіть ***roadEast***. Встановіть у властивості ***New car*** (новий автомобіль): ***Car***;

✓ тепер нам потрібно додати блок, який спрямовуватиме частину машин по східній дорозі на північ, а частина - на південь. Для цього перетягніть блок ***SelectOutput*** з Бібліотеки моделювання процесів на діаграму та з'єднайте його з щойно створеним блоком ***carSourceE***. Назвіть цей блок ***selectNS***;

✓ з'єднайте два вихідні порти блоку ***SelectOutput*** із вхідними портами блоків ***carMoveToN*** та ***carMoveToS*** (рис. 2.76).

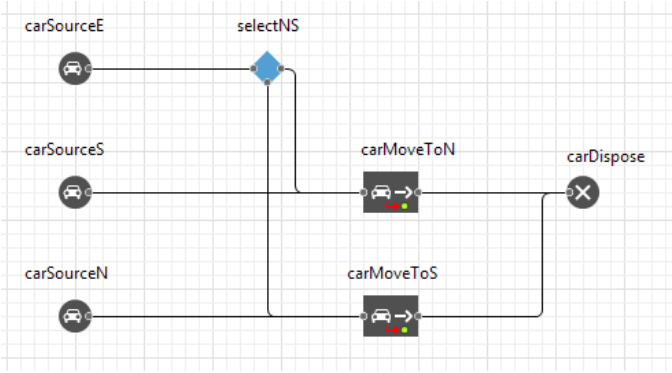


Рисунок 2.76 – Додання блоку ***SelectOutput***

Якщо ви запустите модель, то побачите, що тепер машини рухаються в обох напрямках головної дороги, а також приїжджають зі східної дороги, після чого прямують або на південь, або на північ. Єдине, що залишилося неврахованим у поточній моделі – це поворот машин з головної на східну дорогу.

Треба додати ще кілька блоків у діаграму процесу, які дозволять промоделювати цей останній неврахований сценарій руху машин.

Додайте ще один блок ***SelectOutput***. Помістіть його між блоками ***carSourceS*** та ***carMoveToN*** (рис. 2.77). Цей блок потрібний нам, щоб розподілити потік автомобілів з півдня між східним та північним напрямками. Назвіть цей блок ***selectNE***.

У властивостях блоку налаштуйте маршрутизацію автомобілів. У полі ***Probability*** (ймовірність) введіть 0,7. Тепер у середньому 70 відсотків агентів направлятимуться у верхній порт, а 30 - у нижній (ми з'єднаємо нижній порт із відповідним блоком пізніше).

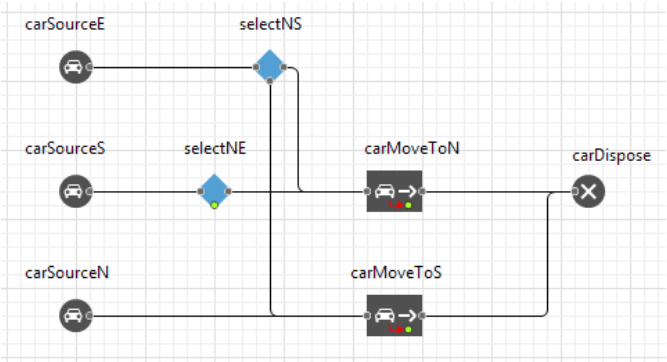


Рисунок 2.77 – Розподілення потоку автомобілів з півдня між східним та північним напрямками

Додайте ще один блок ***SelectOutput***. Назвіть цей блок ***selectSE***. Помістіть його між блоками ***carSourceN*** та ***carMoveToS*** (рис. 2.78). Цей блок потрібний для вибору дороги автомобілів, що їдуть із півночі. Аналогічно змініть частку автомобілів, що їдуть на перехресті прямо, на 70 відсотків, ввівши в поле ***Probability*** 0,7.

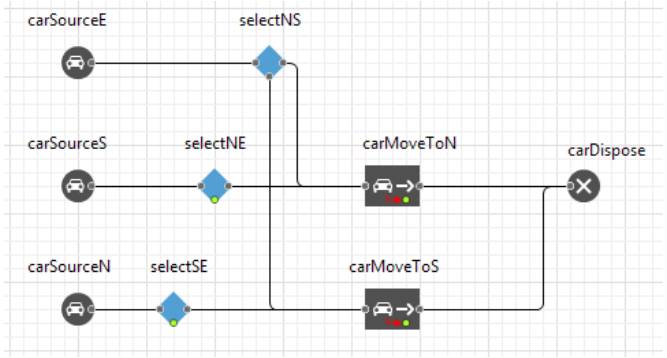


Рисунок 2.78 – Реалізація вибору дороги автомобілів, що їдуть із півночі.

Додайте ще один блок ***CarMoveTo***. Назвіть його ***carMoveToE***. З'єднайте вхідний порт із нижніми портами двох блоків, ***selectNE*** і ***selectSE***. Вихідний порт з'єднайте з блоком ***carDispose*** (рис. 2.79)

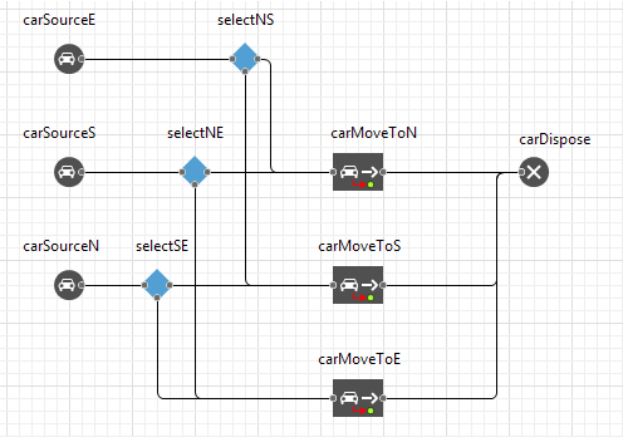


Рисунок 2.79 – Підсумкова діаграма процесу

У властивостях блоку ***carMoveToE*** виберіть поле ***Road*** (дорога): ***roadEast***, а також виберіть опцію ***Destination: end of backward lane*** (доїхати до кінця смуги: зустрічного руху).

Ми закінчили завдання логіки руху машин на перехресті. Запустіть модель і поспостерігайте за тим, як машини рухаються дорогами.

Додавання паркування

Наступним етапом треба додати паркування на узбіччі дороги. Але спочатку трохи змінимо перехрестя в моделі, щоб досягти більш точної відповідності супутниковому знімку.

Щоб збільшити форму та довжину в'їздів на перехресті, потрібно збільшити площу самого перехрестя. Це можна зробити, віддаливши кінцеві точки доріг, що примикають до перехрестя. Для цього виділіть у графічному редакторі дорогу. Акуратно перетягніть кінцеву точку вбік від перехрестя, що примикає до перехрестя (рис. 2.80). При цьому відпустити кнопку миші потрібно лише в тому випадку, якщо точка все ще підсвічується зеленим кольором – це означає наявність з'єднання з перехрестям. Якщо ж точка стане білою, це означає, що ви перетягнули кінець дороги занадто далеко від перехрестя. У такому разі скасуйте останню дію, натиснувши кнопку панелі інструментів Скасувати, а потім повторіть дію.

Рисунок 2.80 – Зміна перехрестя моделі

Аналогічно перетягніть і кінцеві точки інших двох доріг. В результаті в'їзди на головну дорогу повинні будуть набути тієї ж форми, що й дороги на супутниковому знімку (рис. 2.81).

Рисунок 2.81 – Остаточний вигляд перехрестя

Перетягніть елемент ***Parking Lot*** (паркування) з розділу ***Space Markup*** (розмітка простору) бібліотеки дорожнього руху на графічну діаграму та розташуйте його біля східної дороги. У властивостях змініть тип паркування ***Type*** з паралельного ***Parallel*** на перпендикулярне ***Perpendicular***. У результаті паркування має виглядати таким чином (рис. 2.82):

Рисунок 2.82 – Додання елементу паркування

За замовчуванням паркування створюється з п'ятьма місцями для паркування. Кількість місць для паркування можна змінити у властивостях елемента, але в нашому випадку нам потрібне паркування саме на п'ять машин.

Тепер потрібно додати в нашу діаграму процесу блоки, які моделюватимуть заїзд автомобілів на паркування та перебування там протягом певного часу.

Насамперед потрібно видалити з'єднувач, що веде з блоку ***selectNE*** в блок ***carMoveToE***, оскільки в цьому місці діаграми нам потрібно буде зробити ще один поділ потоку машин - частина машин поїде на паркування, тоді як інші - до кінця дороги.

Щоб зробити це поділ потоку машин, додайте ще один блок ***SelectOutput***. Назвіть цей блок ***selectParking***. Вкажіть у його властивостях ймовірність 0,1 (10 відсотків авто буде спрямовано на паркування)

Додайте ще один блок ***CarMoveTo*** (у гілку процесу, що веде з верхнього порту блоку ***selectParking***, рис. 3.80). Назвіть його ***carMoveToParking***. Цей блок моделюватиме рух автомобілів до паркування. У властивостях блоку виберіть Ціль руху: паркування і потім виберіть в полі Паркування ім'я створеного нами раніше паркування: ***parkingLot***.

Для відтворення затримки автомобілів на парковці додайте блок ***Delay***. Назвіть цей блок ***parking*** і з'єднайте його так, як показано на рис. 2.83. Цей блок моделює затримку, пов'язану з виконанням аген-

том певної операції (у нашому випадку – очікування на парковці). Вкажіть у властивостях час затримки ***triangular(5, 15, 45) хвилин***. Оскільки на парковці можуть одночасно перебувати до 5 машин, введіть у поле Місткість: 5.

У блоках типу ***CarMoveTo*** крім двох стандартних портів на лівій і на правій межі блоку є і ще один, поміщений на нижню межу значка. До нього веде червона стрілочка. Цей порт називається ***outWayNotFound*** і є вихідним портом, у який перенаправляються ті автомобілі, для яких не вдалося побудувати маршрут до заданої в цьому блоці мети руху.

З'єднайте нижній вихідний порт блоку ***carMoveToParking*** із портом блоку ***carMoveToE***. У нашому випадку автомобілі будуть прямувати в цей порт, якщо на момент надходження автомобіля в блок ***carMoveToParking*** на парковці не виявиться вільних місць. Таким чином, якщо парковка буде повністю зайнята, то автомобілі просто продовжуватимуть рух далі по дорозі.

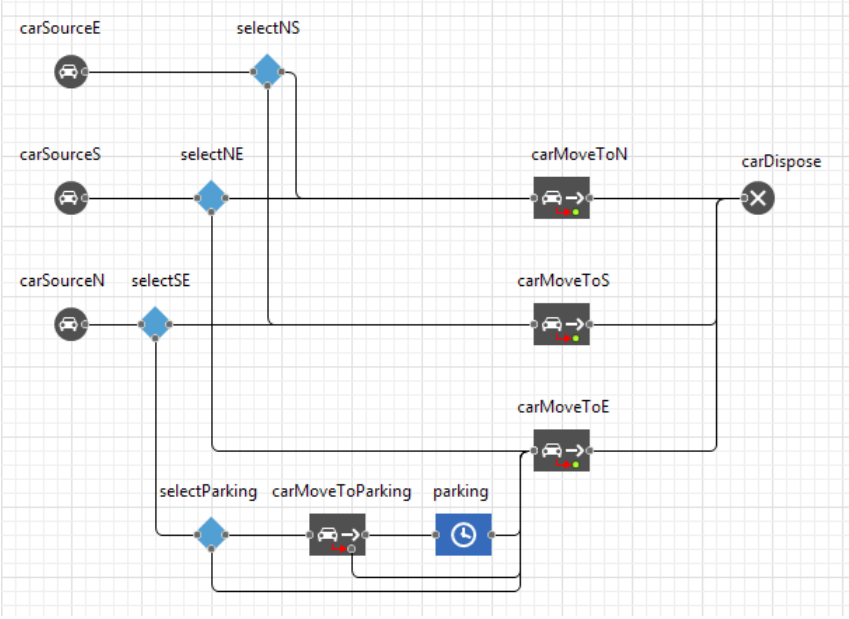


Рисунок 2.83 – Моделювання паркування автомобілів

Додавання автобусів

Зараз у нашій моделі є тільки легкові автомобілі. Наступним етапом додамо до неї рейсові автобуси. Автобуси проїжджатимуть з півдня на північ і зупинятимуться на зупинці.

✓ Додайте ще один блок бібліотеки дорожнього руху ***CarSource***. Цей блок моделюватиме появу автобусів. Назвіть блок ***busSource*** і з'єднайте його з ***carMoveToN***.

✓ Змініть властивості щойно створеного блоку. Встановіть ***Arrival rate*** (інтенсивність появи) рівної *20 на годину*.

✓ У якості ***Road*** (дорога) виберіть ім'я дороги ***roadSouth***.

✓ В полі ***Length*** (довжина) задайте довжину створюваного автобуса: *10 метрів*.

✓ У полі ***Initial speed*** (початкова швидкість) введіть *40 км/год*.

✓ Ми хочемо, щоб цей блок створював не легкові автомобілі, як це роблять створені раніше блоки ***CarSource***, а автобуси. Для цього створіть ще один тип агента – автобус ***Bus*** з фігурою анімації автобуса (за аналогією створення типу агента ***Car***).

Намалюйте автобусну зупинку

➢ Перетягніть елемент ***Bus Stop*** (автобусна зупинка) з розділу ***Space Markup*** (розмітка простору) панелі Бібліотека дорожнього руху на графічну діаграму. Розташуйте зупинку на узбіччі, як це зроблено на рис. 2.84.

Рисунок 2.84 – Додання автобусної зупинки

➢ Подовжіть зупинку. Для цього виділіть її клацанням миші і потім перетягніть квадратний маркер на краю.

Тепер додайте нові блоки у діаграму процесу (рис. 2.85).

✓ Додайте ще один блок ***CarMoveTo*** (у гілку процесу, що веде з блоку ***busSource***, див. малюнок нижче). Назвіть блок ***busMoveToStop***. Він моделюватиме рух автобусів до зупинки.

✓ У властивостях блоку ***busMoveToStop*** виберіть ***Moves to*** (ціль руху): *автобусна зупинка*, а потім виберіть в полі ***Bus stop*** (автобусна зупинка) ім'я створеної нами раніше зупинки: ***busStop***.

✓ Тепер нам потрібно промоделювати, як автобуси, що приїхали на зупинку, перебувають на ній протягом певного часу. Для цього додайте блок Бібліотеки моделювання процесів ***Delay***. Назвіть цей блок ***stop*** і з'єднайте його так, як показано на рис. 3.82.

✓ Відкрийте властивості цього блоку та задайте там Час затримки: triangular(15, 30, 90) секунд.

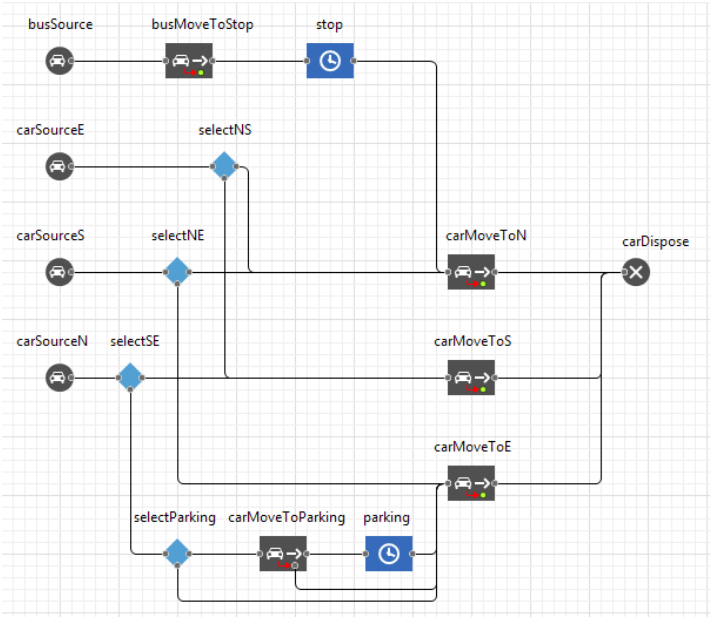


Рисунок 2.85 – Моделювання руху автобусів

Додавання світлофорів

На останньому етапі треба зробити перехрестя регульованим, додавши світлофори. Для цього нам знадобиться блок бібліотеки дорожнього руху ***Traffic Light***. Блок ***Traffic Light*** визначає логіку роботи одного або декількох світлофорів, що регулюють рух машин на перехресті або на пішохідному переході.

Задайте логіку роботи світлофорів

- Додайте на діаграму блок ***Traffic Light*** з панелі Бібліотека дорожнього руху. У цього блоку немає портів, і його не потрібно з'єднувати з жодними іншими блоками діаграми процесу. Його можна додати в будь-яке місце графічного полотна.
- Відкрийте властивості блоку. Зверніть увагу на ключовий параметр блоку: ***Defines the mode for*** (*задає режим роботи*). Цей параметр має три альтернативні опції.

Intersection's stop lines (*стоп-ліній перехрестя*), вибрано за замовчуванням. У нашому випадку ми залишимо обрану за замовчуванням опцію оскільки світлофори, що задаються цим блоком, регулюватимуть рух на перехресті, причому кожна фаза світлофора буде дозволяти / зупиняти рух відразу на всіх смугах дороги, що примикає до перехрестя.

Intersection's lane connectors (з'єднувачів смуг перехрестя), якщо на регульованому перехресті дозволені напрямки руху машин під час однієї і тієї ж фази світлофора можуть відрізнятися для різних смуг однієї дороги.

Specified stop lines (заданих стоп-ліній), якщо світлофори, що моделюються цим блоком, регулюють рух машин на одному або декількох пішохідних переходах, заданих стоп-лініями AnyLogic.

- У властивості ***Intersection*** (перехрестя) виберіть ім'я намальованого нами перехрестя: ***intersection*** (рис. 2.86).
- Після того, як ви вкажете перехрестя у властивостях блоку ***Traffic Light***, всі стоп-лінії цього перехрестя з'являться нижче у таблиці ***Phases*** (фази). Тут ви можете задати режим роботи світлофора - як змінюватимуться червона, зелена (а опціонально і жовта) фази для стоп-ліній, що входять у обране перехрестя.

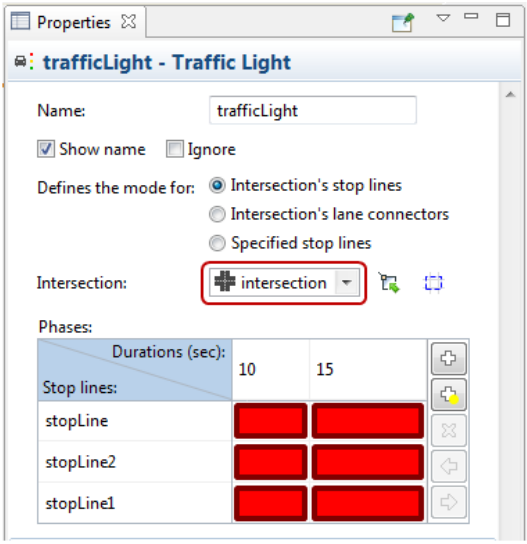


Рисунок 2.86 – Властивості елементу світлофора

- Налаштуйте фази світлофора. Нехай 30-секундна зелена фаза світлофора для руху головною дорогою (на час якої в'їзд на перехрестя зі східної дороги припиняється) змінюється 20-секундною червоною фазою (зеленою для в'їзду зі східної дороги), і ці фази нескінченно змінюють одна одну. За замовчуванням таблиця містить два стовпці, один для кожної фази. Необхідно налаштувати зелену та червону фази для кожної стоп-лінії. Почнемо із першої фази. Клацніть на заголовку стовпця першої фази та вкажіть тривалість фази, що дорівнює 30 секундам.
- Оскільки ми виділили комірку таблиці ***Phases*** , що відповідає першій фазі світлофора, ви побачите, що у графічному редакторі всі фігури будуть тимчасово приховані, а стоп-лінії виділені червоним кольором. Це означає, що ми перебуваємо в режимі редагування фази, який дозволяє змінювати стан кожної стоп-лінії у цій фазі прямо у графічному редакторі.
- Натисніть на стоп-лінії в графічному редакторі, як показано на рис. 2.87. Ви побачите, що колір стоп-лінії зміниться із червоного на зелений. Аналогічним чином зміниться і колір відповідного осередку таблиці ***Phases***.

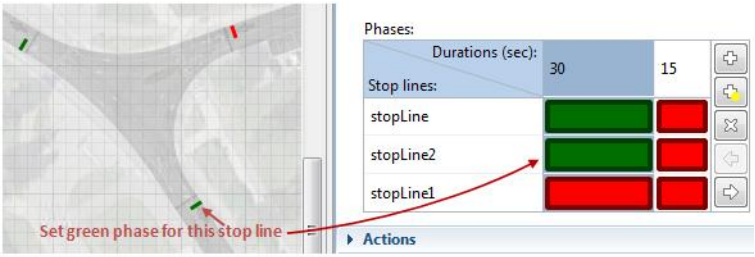


Рисунок 2.87 – Налаштування першої фази світлофора

- Таким же чином зробіть першу фазу світлофора зеленою та для іншої стоп-лінії, розташованої на тій же дорозі.
- Тепер налаштуємо другу фазу. Клацніть по заголовку стовпця другої фази та вкажіть її тривалість, що дорівнює 20 секундам, після чого клацніть по стоп-лінії, розташованій на в'їзді зі східної дороги. У результаті таблиця ***Phases*** має виглядати як показано на рис. 2.88.

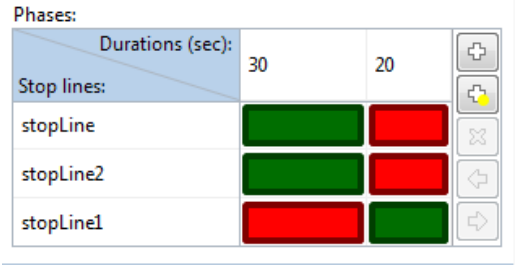


Рисунок 2.88 – Налаштування другої фази світлофора

Запустіть модель. Ви зможете побачити, як по черзі змінюються фази світлофорів, що дозволяють рух то головною дорогою, то на в'їзді на перехрестя зі східної дороги.

Щоб краще розуміти, як завантажені дороги за заданого режиму роботи світлофорів, можна ввімкнути відображення пробок.

Для цього додайте на діаграму блок ***Road Network Descriptor*** із панелі Бібліотека дорожнього руху. Цей блок також не потрібно з'єднувати з іншими блоками діаграми процесу (рис. 2.89).

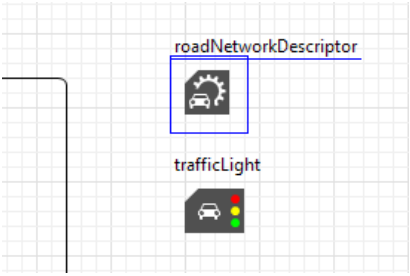


Рисунок 2.89 – Додання блоку відображення пробок

У властивостях блоку виберіть мережу доріг ***roadNetwork***.

Відкрийте розділ властивостей ***Density map*** (карта пробок) на дорозі та встановіть прапорець ***Enable density map*** (показувати пробки.

Запустіть модель та вивчіть ситуацію на перехресті (рис. 2.90).

Рисунок 2.90 – Анімація роботи моделі

Тепер ви можете спробувати змінити режим роботи світлофорів (наприклад, задати фази тривалістю 30/20 або 30/30 секунд) та вивчити, за якого режиму досягається найкращий для водіїв результат.

ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Лащених О. А., Кузькін О. Ф., Грицай С. В. Імовірнісні і статистико-експериментальні методи аналізу транспортних систем: навч. посібник.– Запоріжжя, ЗНТУ, 2011.– 419 с.

2. Турпак С. М. Логістичні системи управління залізничним транспортом металургійних підприємств : монографія. Херсон: Грінь Д. С., 2015. 264 с.

3. Б.П. Середа, С.М. Турпак, І.В. Кругляк, О.О. Острогляд, Д.Я. Муковська, Д.Б. Середа, Д.О. Кругляк. Підвищення експлуатаційної стійкості та ефективності роботи промислового транспорту в умовах металургійного підприємства: монографія. Кам'янське : ДДТУ, 2021. 272 с.

4. Турпак, С.М. Формування комплексного підходу до досліджень впливу транспортно-промислових процесів на сіті-логістичні системи / Турпак С.М., Кузькін О.Ф., Трушевський В.Е., Острогляд О.О. // Вісник машинобудування та транспорту. №2(18). 2023. С. 168-174. doi: 10.31649/2413-4503-2023-18-2-168-174.