**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ МІСЬКОГО ГОСПОДАРСТВА імені О. М. БЕКЕТОВА**

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання практичних робіт та організації самостійної роботи
з навчальної дисципліни

# «АНАЛІЗ ЛОГІСТИЧНИХ ПРОЦЕСІВ»

*(для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форм навчання зі спеціальності 073 – Менеджмент, освітня програма «Логістика»)*

**Харків**
**ХНУМГ ім. О. М. Бекетова**
**2024**

Методичні рекомендації до виконання практичних робіт та організації самостійної роботи з навчальної дисципліни «Аналіз логістичних процесів» (для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форм навчання зі спеціальності 073 – Менеджмент, освітня програма «Логістика) / Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова ; уклад. А. С. Галкін. – Харків : ХНУМГ ім. О. М. Бекетова, 2024. – 119 с.

Укладач д-р техн. наук, проф. А. С. Галкін

Рецензент

**Ю. О. Давідіч,** доктор технічних наук, професор, професор кафедри транспортних систем і логістики Харківського національного університету міського господарства імені О. М. Бекетова

*Рекомендовано кафедрою транспортних систем і логістики, протокол № 1 від 19.08.2024*

**ЗМІСТ**

## Практична робота № 1

# ТЕОРЕТИЧНА ОСНОВА МОДЕЛЮВАННЯ ТА ІМІТАЦІЇ НА ОСНОВІ АГЕНТІВ У ПРОГРАМНОМУ ЗАБЕЗПЕЧЕННІ *ANYLOGIC*

**Мета:** отримати теократичні відомості з основ моделювання та імітацій на основі агентів.

## Хід виконання

Моделювання та імітаційне моделювання. Моделювання – це спосіб вирішення реальних проблем. У багатьох випадках ми не можемо дозволити собі експериментувати з реальними об’єктами, щоб знайти правильні рішення: будівництво, руйнування та внесення змін можуть бути надто дорогими, небезпечними або просто неможливими. Якщо це так, можна створити модель, яка використовує мову моделювання для представлення реальної системи. Цей процес передбачає абстракцію: ми включаємо деталі, які вважаємо важливими, і залишаємо без уваги ті, які вважаємо неважливими. Модель завжди менш складна, ніж вихідна система (рис. 1.1).

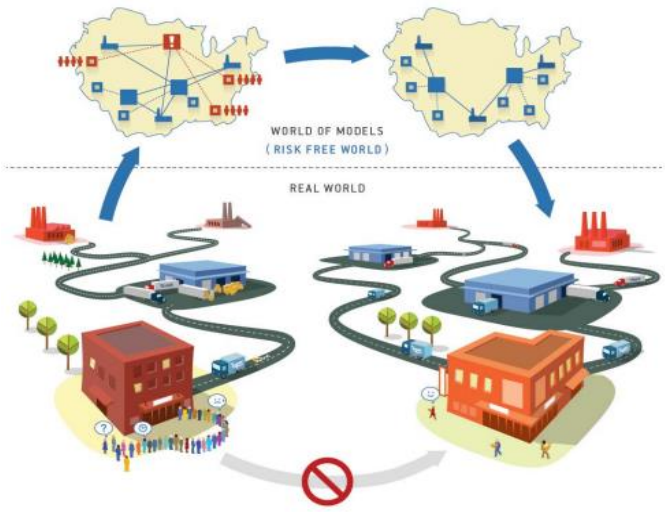



Рисунок 1.1 – Оригінальна система

Моделювання. Фази побудови моделі – це відображення реального світу у світі моделей, вибір рівня абстракції та мови моделювання – усі менш формальні,

ніж процес використання моделей для вирішення проблем. Це, однак, більше мистецтво, ніж наука. Після того, як ми створили модель – а іноді навіть під час її створення – ми можемо почати досліджувати та усвідомлювати структуру й поведінку нашої системи, тестувати, як вона поводитиметься за різноманітних умов, грати та порівнювати сценарії та здійснювати оптимізацію. Коли ми знайдемо рішення, то зможемо відобразити його в реальному світі. Моделювання – це пошук шляху від проблеми до її вирішення без ризиків, коли нам дозволено робити помилки, скасовувати дії, повертатися в минуле та починати спочатку.

Існує багато типів моделей, у тому числі ментальні моделі, які ми використовуємо, щоб зрозуміти, як все працює в реальному світі: друзі, родина, колеги, водії автомобілів, місто, де ми живемо, речі, які ми купуємо, економіка, спорт і політика. Усі наші рішення: що ми повинні сказати нашій дитині, що ми повинні їсти на сніданок, за кого ми повинні проголосувати або куди ми повинні повести нашу дівчину на вечерю – усе базується на ментальних моделях.

Комп'ютери – потужні інструменти моделювання, вони пропонують нам гнучкий віртуальний світ, де ми можемо створювати майже все, що тільки можна уявити. Звісно, існує багато типів комп'ютерних моделей – від базових електронних таблиць, які дозволяють будь-кому моделювати витрати, до складних інструментів імітаційного моделювання, які допомагають досвідченим користувачам досліджувати динамічні системи, наприклад споживчі ринки чи поля битв.

Аналітичне та імітаційне моделювання. Попросіть команди зі стратегічного планування, прогнозування продажів, логістики, маркетингу або управління проєктами великої організації назвати улюблений інструмент моделювання, і ви швидко зрозумієте, що *Microsoft Excel* є найоптимальнішим варіантом.

Excel має декілька переваг: він доступний широкому загалу, простий у використанні та дозволяє додавати сценарії до формул, оскільки логіка вашої електронної таблиці стає дедалі складнішою (рис. 1.2).

Аналітична модель (електронна табл. *Excel*). Технологія моделювання на основі електронних таблиць проста. Ви вводите вхідні дані в одні комірки та переглядаєте вихідні дані в інших. Формули – а в більш складних моделях сценарії – поєднують вхідні та вихідні значення. Різноманітні доповнення дозволяють виконувати експерименти щодо зміни параметрів за допомогою методу Монте-Карло або оптимізації.

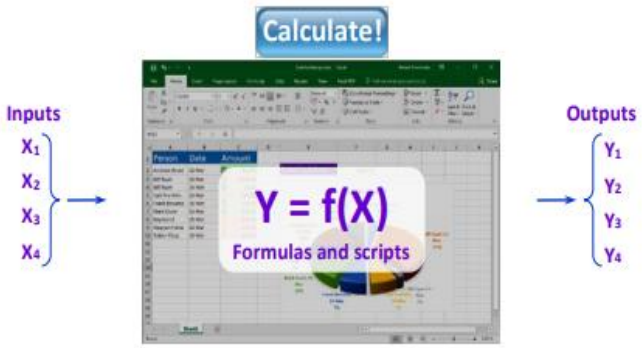



Рисунок 1.2 – Програма *Excel*

Однак існує також великий клас проблем, коли аналітичне (на основі формули) рішення або важко знайти, або його просто немає.

Цей клас включає динамічні системи, які характеризуються наявністю:

– нелінійної поведінки;

– «пам’яті»;

– неінтуїтивної взаємодії між змінними;

– часових та причинно-наслідкових залежностей;

– усього переліченого вище в поєднанні з невизначеністю та великою кількістю параметрів (здебільшого неможливо отримати правильні формули, а надто скласти уявну модель такої системи).

Розглянемо проблему, яка потребує оптимізації залізничного або вантажного парку. Недоцільно використовувати електронну таблицю *Excel* для керування такими факторами, як розклад подорожей, час завантаження та розвантаження, обмеження щодо часу доставки та пропускна здатність кінцевих пунктів. Доступність транспортного засобу в певному місці, даті та часі залежить

від послідовності попередніх подій, і визначення того, куди відправити транспортний засіб, коли він простоює, потребує аналізу майбутніх подій.

Формули, які виражають статичні залежності між змінними, зазвичай недоцільно використовувати для опису систем із динамічною поведінкою, тому для аналізу динамічних систем ми використовуємо іншу технологію моделювання – імітаційне моделювання. Імітаційна модель завжди є виконуваною моделлю: її запуск створює траєкторію змін стану системи. Подумайте про імітаційну модель, як про набір правил, які підказують вам, як перейти від поточного стану системи до майбутньої. Правила можуть набувати різних форм, включаючи диференціальні рівняння, діаграми станів, блок-схеми процесу та графіки. Результати моделі створюються та спостерігаються під час роботи моделі. Імітаційне моделювання потребує спеціальних програмних засобів, які використовують специфічні для моделювання мови.

Хоча для якісного імітаційного моделювання вам доведеться вчитися, ваш час і зусилля будуть винагороджені, коли модель надасть якісний аналіз динамічної системи. Багато людей, особливо ті, хто добре знає *Microsoft Excel* або має досвід програмування, для моделювання динамічної системи намагаються використовувати електронну таблицю. Коли вони намагаються охопити якомога більше деталей, то неминуче починають відтворювати функціональні можливості симуляторів *Excel*. Отримані моделі повільні та некеровані, і їх зазвичай швидко відкидають. В аналітичному рішенні практично неможливо використати жодну з цих деталей. Навіть якби існували формули для керування обраною конфігурацією, найменші зміни в процесі могли б анулювати їх і знадобився б професійний математик, щоб їх виправити.

Переваги імітаційного моделювання. Імітаційне моделювання має шість ключових переваг:

1. Імітаційні моделі дозволяють аналізувати системи та знаходити рішення там, де такі методи, як аналітичні обчислення та лінійне програмування, не дають результатів.

2. Після вибору рівня абстракції легше розробити імітаційну модель, ніж аналітичну. Зазвичай це потребує менше роздумів, а процес розробки масштабований, поетапний і модульний.

3. Структура імітаційної моделі відображає структуру системи.

4. В імітаційній моделі можна вимірювати значення та відстежувати сутності в межах рівня абстракції, а також будь-коли додавати вимірювання та статистичний аналіз.

5. Однією з найбільших переваг симуляції є можливість відтворення та анімування поведінки системи в часі. Анімацію можна використовувати для демонстрації, перевірки та налагодження.

6. Імітаційні моделі набагато переконливіші, ніж електронні таблиці *Excel*. Якщо ви використовуєте симуляцію, щоб підтвердити свою пропозицію, ви матимете велику перевагу щодо тих, хто використовує лише числа.

*Застосування симуляції*

Антиблокувальні гальма автомобіля, евакуація футбольних уболівальників зі стадіону, рух на перехресті, що регулюється світлофором, дії солдатів на полі бою – приклади проблем, які потребують малоабстрактного моделювання. Моделі у верхній частині абстрактні, зазвичай у них використовуються агрегати – сукупність споживачів і статистика зайнятості, а не окремі об’єкти. Оскільки їхні об’єкти взаємодіють на високому рівні, вони можуть допомогти зрозуміти взаємозв’язки, наприклад гроші, які компанія витрачає на рекламу, впливають на продажі, не потребуючи моделювання проміжних кроків.

Інші моделі мають проміжний рівень абстракції. Якщо ми моделюємо відділення невідкладної допомоги лікарні, нам потрібно формувати фізичний простір, щоб знати, скільки часу потрібно комусь для того, щоб пройти від відділення невідкладної допомоги до рентгенівської станції, а фізична взаємодія між людьми в будівлі не має значення, оскільки ми припускаємо, що будівля не перевантажена. У моделі бізнес-процесу або кол-центру ми можемо моделювати послідовність і тривалість операцій, а не їх розташування. У моделі транспортування ми ретельно враховуємо швидкість вантажівки чи залізничного

вагона, але в моделі ланцюга постачання вищого рівня тільки припускаємо, що для отримання замовлення потрібно від семи до десяти днів. Вибір правильного рівня абстракції має вирішальне значення для успіху проєкту моделювання.

У процесі розробки моделі потрібно – навіть бажано – час від часу переглядати рівень абстракції моделі. У більшості випадків необхідно починати з високого рівня абстракції й додавати потрібні деталі. У сучасному імітаційному моделюванні виокремлюють три методи: дискретно-подієвий, агентний і системної динаміки (рис. 1.3).

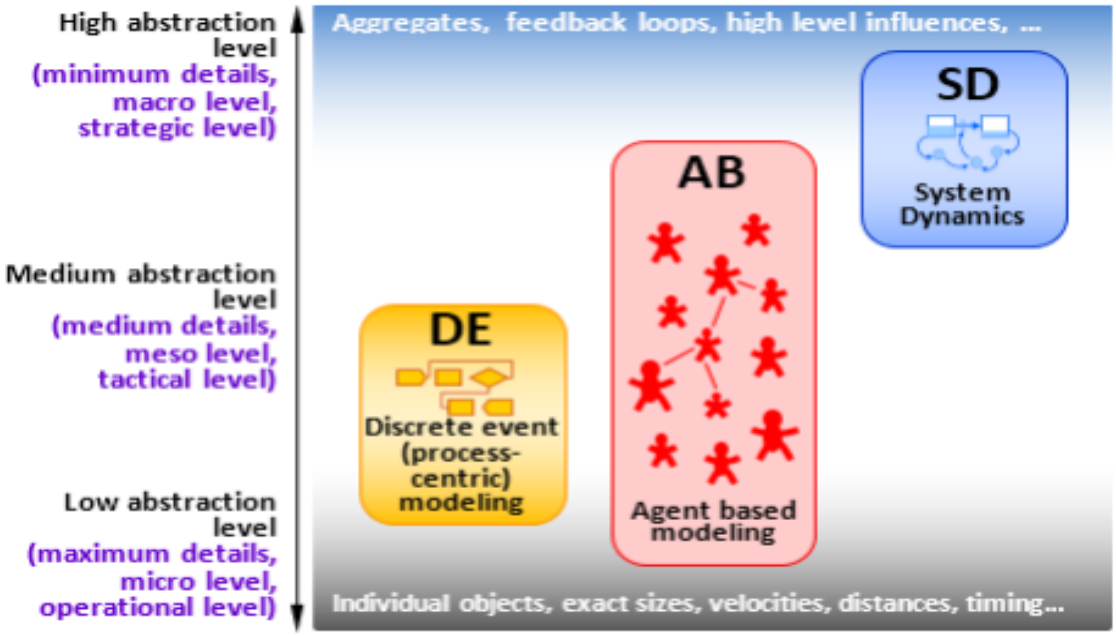



Рисунок 1.3 – Сучасне імітаційне моделювання

*Методи імітаційного моделювання*

У імітаційному моделюванні метод – це структура, яку ми використовуємо для відображення системи реального світу як її модель. Можна розглядати метод як тип мови або своєрідні «терміни та умови» для побудови моделі.

Відомі три методи:

– *System Dynamics;*

– *Discrete Event Modelin;*

– *Agent Based Modeling.*

Кожен метод обслуговує певний діапазон рівнів абстракції.

Системна динаміка передбачає дуже високу абстракцію і зазвичай використовується для стратегічного моделювання. Моделювання дискретних подій підтримує середню та середньо-низьку абстракцію. Посередині розміщені моделі на основі агентів, які можуть варіюватися від дуже детальних, де агенти представляють фізичні об’єкти, до дуже абстрактних, де агенти представляють конкуруючі компанії чи уряди. Після ретельного розгляду системи, яку ви хочете змоделювати, та відповідно до ваших цілей потрібно обрати свій метод. На рисунку 1.4 подано приклад проблеми, як  уявлення людини, що буде будувати процес. Блок-схема процесу така: клієнти є об’єктами системи, співробітники – ресурсами, модель створена на основі взаємодії агентів, де споживачі є агентами, на яких впливає реклама, комунікація, а також взаємодія з агентами та працівниками.

Ви також можете виявити, що найкращим способом моделювання різних частин системи є використання різних методів, і в такій ситуації багатометодна модель найкраще задовольнить ваші потреби.

Агентне моделювання є відносно новим методом порівняно з системною динамікою та моделюванням дискретних подій. Насправді моделювання на основі агентів було переважно науковою темою, допоки симулятори-практики не почали використовувати його (близько 15-ти років тому). Це спричинило:

1. Бажання глибше зрозуміти системи, які традиційні підходи моделювання охоплювали недостатньо.

2. Прогрес у технології моделювання, який став можливим завдяки інформатиці, наприклад об’єктно-орієнтоване моделювання, *UML* і діаграми станів.

3. Швидке зростання потужності ЦП і пам’яті. Агентні моделі є більш досконалими, ніж системна динаміка та моделі дискретних подій (рис. 1.4).


Рисунок 1.4 – Агентна модель

Стандартної мови для моделювання на основі агентів не існує, і структура моделі на основі агентів визначається графічними редакторами або сценаріями. Для визначення поведінки агента використовують різні способи. Зазвичай слово «агент» застосовується у значенні «стан», тому дії та реакції агента залежать від його стану. Отже, поведінку агента найдоцільніше визначати за допомогою діаграм стану. Іноді поведінка визначається правилами, які виконуються після спеціальних подій. У багатьох випадках найкращим способом охоплення внутрішньої динаміки агента є використання системної динаміки або підхіду до дискретних подій, а потім потрібно розмістити всередині агента фондову та блок-схему або блок-схему процесу. Подібним чином зовнішні агенти визначають динаміку середовища, у якому вони живуть, що природно моделювати за допомогою традиційних методів. Саме тому багато агентних моделей є моделями з кількома методами.

Агенти в агентній моделі можуть представляти різноманітні об’єкти: транспортні засоби, одиниці обладнання, проєкти, продукти, ідеї, організації, інвестиції, ділянки землі, людей у різних ролях тощо (рис. 1.5).

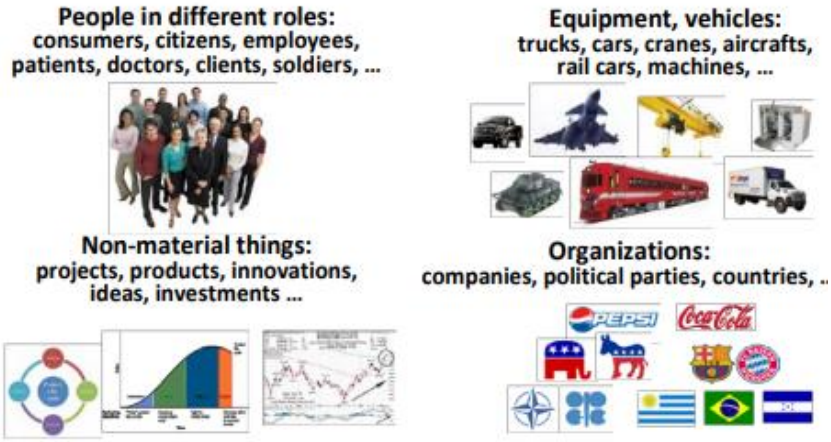



Рисунок 1.5 – Агентна модель

Учені ще досі сперечаються про те, які властивості повинен мати об’єкт, щоб бути «агентом»: проактивні та реактивні якості, просторове усвідомлення, здатність до навчання, соціальні здібності, інтелект тощо. Однак у прикладному моделюванні на основі агентів ви будете знаходити різних агентів: одні спілкуються, тоді як інші живуть у повній ізоляції, одні живуть у просторі, а інші без простору, деякі навчаються та адаптуються, а інші ніколи не змінюють своїх моделей поведінки.

## Завдання

1. Навести по два приклади використання кожного методу (*System Dynamics*, *Discrete Event Modeling*, *Agent Based Modeling*). Наведіть сильні та слабкі сторони кожного методу.

2. Встановить програмне забезпечення *AnyLogic*.

## Питання для самоконтролю

1. Які проблеми можна вирішити за допомогою методу моделювання?
2. Які методи використовуються під час імітаційного моделювання?
3. Які об’єкти можуть представляти агенти в агентній моделі?

## Практична робота № 2
## СТВОРЕННЯ СУКУПНОСТІ АГЕНТІВ

**Мета:** набути прикладних навичок зі створення агентної сукупності.

### Хід виконання

Почнемо зі створення простої моделі, яка зображує, як реклама спонукає споживачів купувати наш продукт. Споживачі нашої моделі спочатку не будуть використовувати продукт, але всі вони потенційно зацікавлені в ньому. Ми також представлятимемо вплив реклами на попит споживачів, дозволяючи певному відсотку з них зацікавитися покупкою продукту протягом певного дня. Для наших цілей ефективність реклами 0,1 визначає відсоток потенційних користувачів, які готові купити продукт протягом певного дня. Запустіть *AnyLogic* – відобразиться сторінка привітання. Сторінка привітання ознайомлює з *AnyLogic*, пропонує корисний огляд програми та її функцій, а також дозволяє відкривати приклади моделей (рис. 2.1).

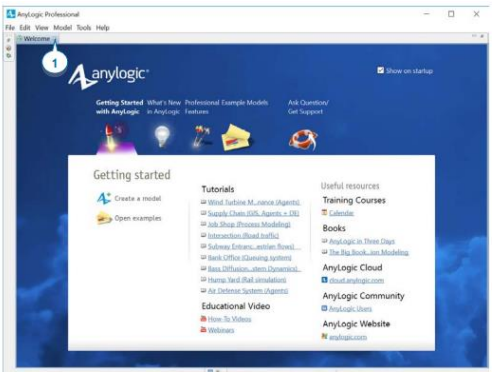



Рисунок 2.1 – Вітальна сторінка *AnyLogic*

Закрийте сторінку привітання та створіть нову модель, вибравши *File* > *New* > *Model* у головному меню *AnyLogic*. Відкриється майстер створення нової моделі (рис. 2.2):

1. У полі *Model name* введіть назву нової моделі: *Market*.

2. У полі *Location* виберіть папку, у якій потрібно створити модель. Ви можете переглянути папку, натиснувши «Огляд», або ввести назву папки, яку ви хочете створити у полі «Розташування». У полі *Model time units* встановіть показник *Days*.

3. Натисніть *Finish*.

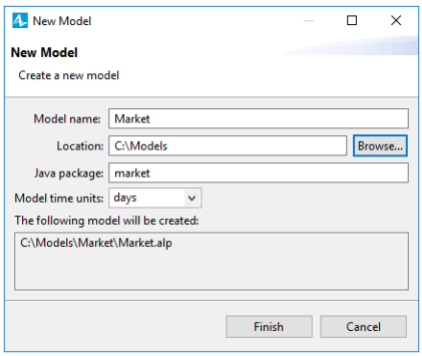



Рисунок 2.2 – Нова модель

Тепер коротко розглянемо інтерфейс *AnyLogic* (рис. 2.3).

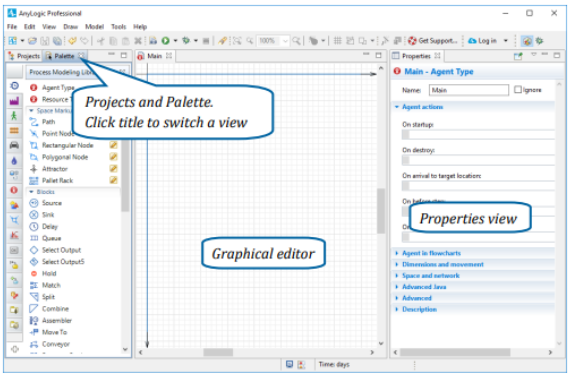



Рисунок 2.3 – Інтерфейс *AnyLogic*

**Робоча область *AnyLogic***:

1. *Graphical editor*. Графічний редактор дозволяє редагувати діаграму типу агента, і ви можете додавати елементи моделі, перетягуючи їх з палітри на діаграму та розміщуючи на *Pallete* редактора. Елементи, які ви розміщуєте в синій рамці, з’являться у вікні моделі, коли ви її запустите.

2. Перегляд *Projects* дозволяє отримати доступ до моделей *AnyLogic*, які ви відкрили в робочій області, а дерево робочої області допоможе вам легко переміщуватися по них.

3. Перегляд *Pallete* містить список об’єктів, згрупованих у палітри. Щоб додати елемент до моделі, перетягніть елемент із палітри до графічного редактора.

4. Перегляд *Properties* дозволяє переглядати та змінювати властивості вибраного елемента.

5. Щоб відкрити / закрити перегляд, виберіть відповідний пункт у меню «Вид». Якщо елемент вибрано, буде видно відповідне подання.

6. Щоб змінити розмір перегляду, використовуйте мишу, щоб перетягнути край перегляду.

7. Ви завжди можете скористатися опцією *Reset perspective* в меню «Інструменти» (*Tools*), щоб повернути види до стандартних положень.

8. Відкриємо вікно *Projects*, щоб перевірити структуру моделі. Перегляди *Pallete* та *Projects* розташовані в лівій частині робочої області, і ви можете перейти з перегляду «Палітра» на режим *Projects*, натиснувши на вкладку *Projects* (рис. 2.4).

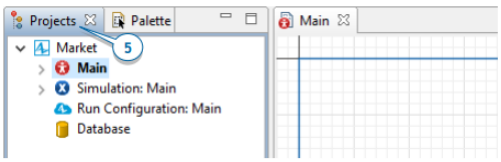



Рисунок 2.4 – Вкладка *Projects*

Навігація моделлю у вікні проєктів.

Перегляд *Projects* дозволяє отримати доступ до проєктів *AnyLogic*, які ви відкрили в робочій області, і ви зможете використовувати дерево робочої області для швидкої та легкої навігації по них.

*AnyLogic* використовує деревоподібну структуру для відображення вашої моделі. Верхній рівень відображає модель, рівень нижче відображає типи агентів і експерименти, а гілки нижнього рівня організовують елементи, які складають структуру агента.

За замовчуванням модель має один тип агента «Основний», один «Симуляція експерименту» та вбудовану базу даних для читання вхідних даних і запису вихідної бази даних симуляції (порожня за замовчуванням). Елемент *Run Configuration* дозволяє налаштовувати вхідні та вихідні дані моделі перед завантаженням її в *AnyLogic Cloud* (рис. 2.5).

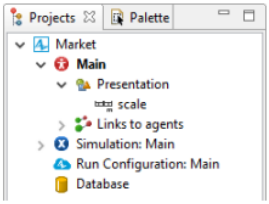



Рисунок 2.5 – Вигляд проєктів

Подвійне натискання на тип агента або експерименту відкриває його діаграму в графічному редакторі.

Натискання на елемент моделі в дереві вибирає елемент і відцентровує його в графічному редакторі. Це може бути доцільним, якщо ви не можете знайти елемент на графічній діаграмі. У графічному редакторі ви побачите порожню діаграму типу головного агента моделі.

Агенти:

– агенти є будівельними блоками моделі, і ви можете використовувати їх для моделювання будь-яких об’єктів реального світу, включаючи організації, компанії, вантажівки, станції обробки, ресурси, міста, роздрібних торговців, фізичні об’єкти, контролерів тощо;

– кожен агент зазвичай представляє один із логічних розділів моделі. Це дозволяє розкласти модель на багато рівнів деталізації.

Наша модель має один тип агента – *Main*. Щоб додати споживачів, нам потрібно створити тип агента для представлення споживачів, а потім сукупність агентів, що складається з екземплярів цього типу агента споживача. В *AnyLogic* ви можете використовувати корисний майстер нового агента для створення агентів.

9. Ми хочемо додати новий елемент моделі, але спочатку потрібно перейти до перегляду палітри, натиснувши вкладку палітри (рис. 2.6).

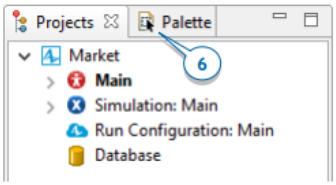



Рисунок 2.6 – Вкладка палітри

10. Відкриття палітри агента. Щоб відкрити певну палітру, перейдіть до перегляду палітри та наведіть вказівник миші на вертикальну навігаційну панель перегляду.

11. Вона розгорнеться, щоб показати назви всіх палітр і ви могли вибрати потрібну. Натисніть на палітру «Агент» у списку, щоб вибрати її (рис. 2.7).

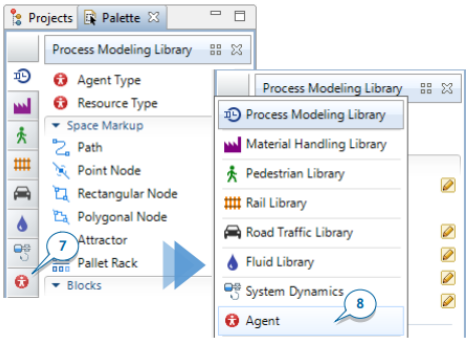



Рисунок 2.7 – Панель навігації

Ознайомившись із піктограмами, ви можете клацнути на потрібну піктограму палітри на панелі навігації.

12. Перетягніть агента з палітри агентів на головну діаграму, і відкриється майстер створення нового агента (рис. 2.8).

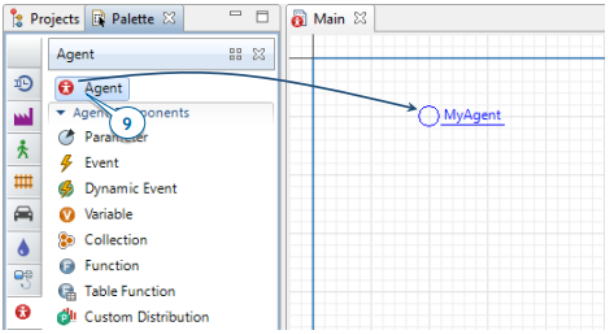



Рисунок 2.8 – Палітра агентів

13. На сторінці «Крок 1. Виберіть, що ви хочете створити» виберіть варіант, який найбільше відповідає вашим потребам. Оскільки ми хочемо створити кілька агентів одного типу, виберіть «Заповнення агентів» і натисніть «Далі» (рис. 2.9).

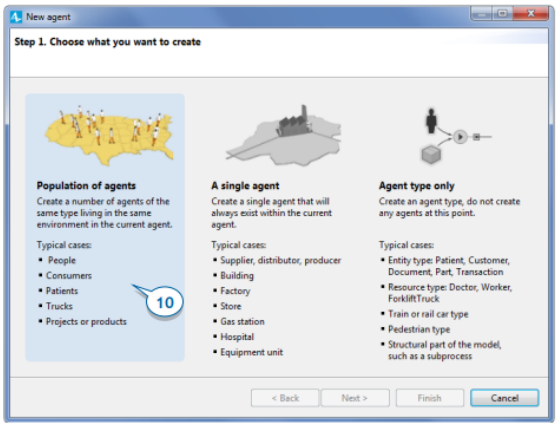



Рисунок 2.9 – Популяція агентів

14. На сторінці «Крок 2. Створення нового типу агента» в полі «Ім’я типу агента» введіть «Споживач» (*Consumer*). Інформація у полі «Ім’я популяції агента» автоматично зміниться на споживачів (рис. 2.10).

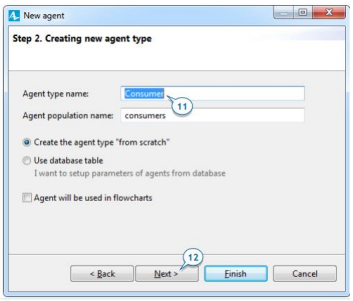



Рисунок 2.10 – Створення нового агента

15. Натисніть *Next*.

16. На сторінці *Agent animation* виберіть форму анімації агента. Оскільки ми створюємо просту модель, яка використовує 2D-анімацію, виберіть 2D та перший пункт загального списку «Людина» та натисніть кнопку «Далі» (рис. 2.11).

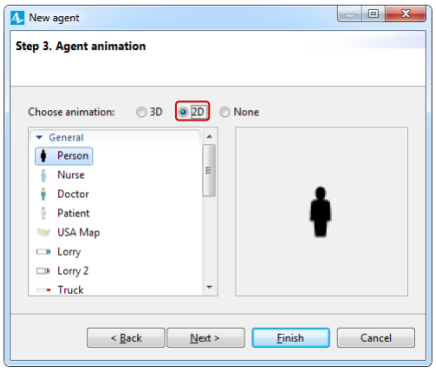



Рисунок 2.11 – Анімація агента

17. На сторінці *Agent Parameters* визначте параметри або характеристики агента. Оскільки наша модель розглядає лише покупки продукту, пов'язані з рекламою, ми додамо параметр *AdEffectiveness* для визначення відсотка потенційних користувачів, які готові придбати продукт протягом певного дня (рис. 2.12).

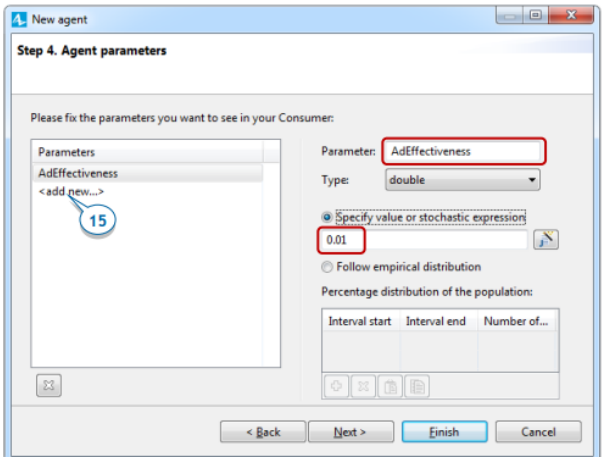



Рисунок 2.12 – Параметр агента

18. У лівій частині в таблиці *Parameters* клацніть, щоб створити параметр.

19. У полі «Параметр» змініть назву параметра за замовчуванням на *AdEffectiveness* і виберіть *Double* як параметр *Type*. Припустимо, що в середньому 1 % потенційних користувачів нашої моделі захочуть купити продукт протягом певного дня, тому вкажіть 0,01, як значення параметра.

20. Натисніть *Next*.

21. На сторінці «Розмір популяції» (*Size population*) введіть 5 000 у полі Створіть популяцію за допомогою ____ агентів, це дозволить створити 5 000 екземплярів типу «Споживач». Кожен екземпляр популяції буде моделювати конкретного агента-споживача. Хоча ми створили популяцію агентів, ми не побачимо анімаційних фігур із 5 000 осіб на головній діаграмі. Замість цього *AnyLogic* використовуватиме 5 000 агентів у популяції, яку ми назвали споживачами, щоб імітувати ринок під час запуску нашої моделі (рис. 2.13).

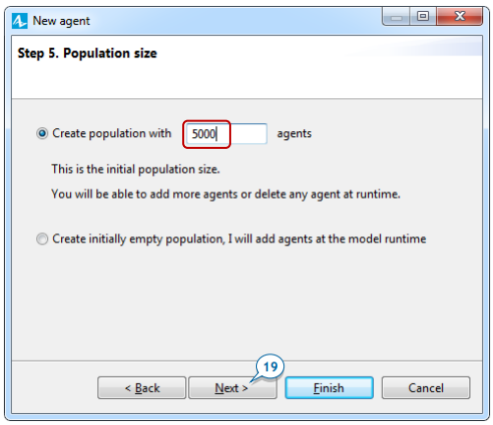



Рисунок 2.13 – Чисельність популяції

22. Натисніть *Next*.

23. На сторінці *Configure new environment* приймаємо значення за замовчуванням для типу простору середовища («Безперервний», *Continuous*), а також значення *Width* and *Height* (500). *AnyLogic* відобразить агентів у прямокутнику 500 пікселів × 500 пікселів.

24. Виберіть поле *Apply random layout*, щоб довільно розподілити агентів по ширині та висоті 500 пікселів, які ми визначили. Оскільки ми не хочемо створювати мережу агента, оберемо стандартний тип мережі *No network/User-defined* (рис. 2.14).

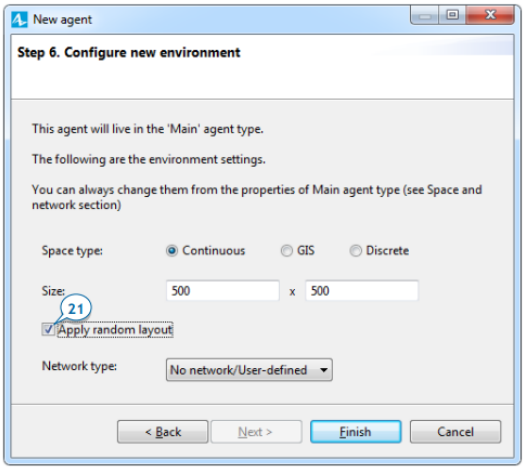



Рисунок 2.14 – Налаштування нового середовища

25. Натисніть *Finish*.

26. Скористаємося поданням *Projects*, щоб побачити нові елементи, створені майстром. Розгорніть гілки дерева моделі, щоб побачити внутрішні складники (рис. 2.15).

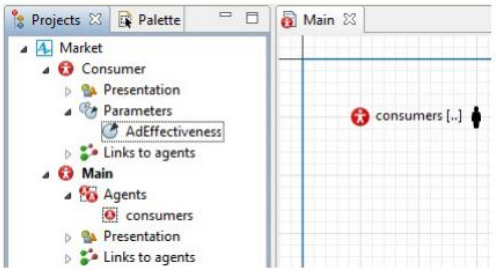



Рисунок 2.15 – Вигляд проєктів

Наша модель тепер має два типи агентів – основний і споживача:

– тип агента-споживача має вигляд анімації агента (особа, у гілці «Презентація») і параметр *AdEffectiveness;*

– тип агента «Основний» містить споживачів сукупності агентів (набір із 5 000 агентів типу «Споживач»).

У середовищі агента «Основний» агент діє, як середовище для популяції споживачів. Оскільки середовище визначає простір, макет, мережу та комунікацію, які використовують наші агенти, нам знадобиться середовище для

організації презентацій наших агентів і моделювання реклами «із вуст в уста», яка виникає під час взаємодії наших агентів.

27. Оберіть *Main* у вкладці *Projects*, щоб відкрити його властивості у вікні «Властивості» (ви знайдете «Властивості» у правій половині вікна *AnyLogic*). У розділі «Простір і мережа» (*Space and network*) основних властивостей ви можете налаштувати параметри середовища для сукупності агентів споживачів (рис. 2.16).

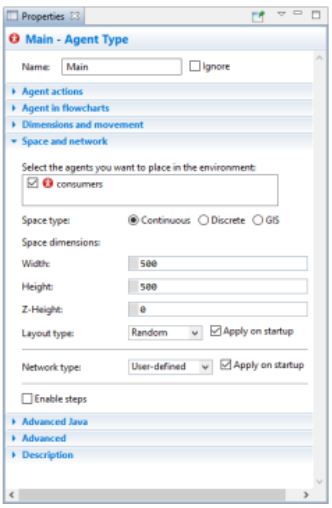



Рисунок 2.16 – Перегляд властивостей

28. На основній діаграмі виберіть нередаговану форму вбудованої анімації популяції агента, відкрийте розділ *Advanced* та виберіть параметр *Draw agent with offset to this position* (рис. 2.17).

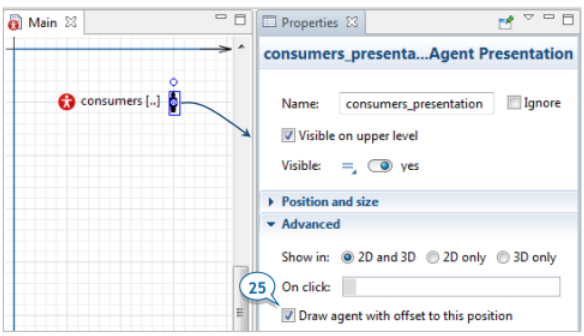



Рисунок 2.17 – Основна діаграма

Як ви можете побачити, на рисунку 2.18 форма анімації визначає лівий верхній кут простору 500 пікселів × 500 пікселів, де будуть перебувати окремі агенти під час запуску моделі.

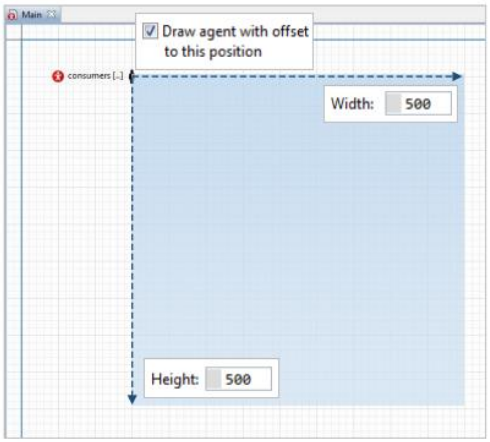



Рисунок 2.18 – Головне меню

Ми закінчили створення цієї дуже простої моделі, і тепер ви можете запустити її та спостерігати за її поведінкою.

29. На панелі інструментів натисніть кнопку *Build button*, щоб побудувати модель і перевірити її на наявність помилок.

30. Знайдіть кнопку *Run button* та клацніть на маленькому трикутнику праворуч. Виберіть експеримент, який хочете запустити, зі списку *Market / Simulation* (рис. 2.19).

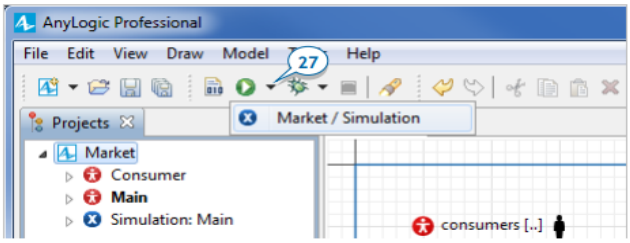



Рисунок 2.19 – Вигляд списку «Ринок» / «Симуляція»

Оскільки ви можете мати кілька відкритих моделей одночасно і кожна модель може мати кілька експериментів, ви повинні вибрати правильний експеримент. Після запуску моделі вікно моделі відображає презентацію

запущеного експерименту *Simulation*. За замовчуванням відображається назва моделі.

31. Клацніть на елементі керування *Run* внизу вікна моделі, щоб запустити модель (рис. 2.20).

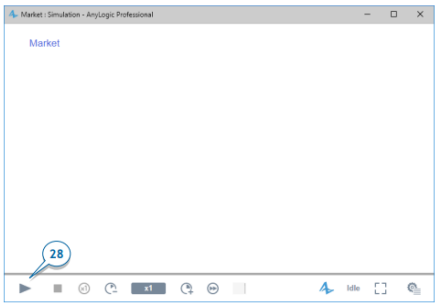



Рисунок 2.20 – Контроль запуску

Ви побачите презентацію моделі (презентацію, яку ви створили для головного агента), яка показує 5 000 анімацій для агентів, які складають сукупність споживачів. Оскільки ми не створювали жодної поведінки для наших агентів, анімація все одно з’явиться (рис. 2.21).

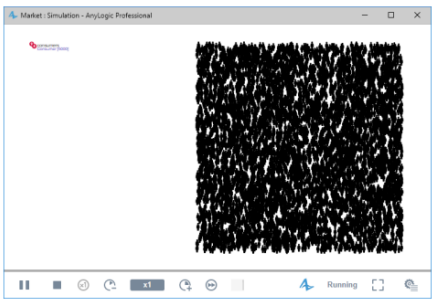



Рисунок 2.21 – Панель керування

Панель керування вікном моделі.

Ви можете використовувати панель керування внизу вікна моделі, щоб керувати виконанням моделі.

*Run* [*Visible when the model is not running*] Запускає симуляцію або, якщо симуляцію було призупинено, відновлює її.

*Pause* [*Visible when the model is running*] призупиняє симуляцію. Ви можете будь-коли відновити призупинену симуляцію.

*Stop* припиняє поточне виконання експерименту.

Щоб переконатися, що модель працює, подивіться на статус симуляції моделі (*Running, Paused, Idle, or Finished*), що відображається на панелі керування.

32. Ми готові визначити логіку споживача. Щоб продовжити розробку моделі, закрийте вікно моделі.

## Питання для самоконтролю

1. Як визначити відсоток потенційних користувачів, які готові придбати продукт протягом певного дня?
2. У якому розділі можна налаштувати параметри середовища для сукупності агентів споживачів?
3. Які дії потрібно виконати для додавання нового споживача?

## Практична робота № 3

## ВИЗНАЧЕННЯ ПОВЕДІНКИ СПОЖИВАЧА

**Мета:** набути прикладних навичок із визначення поведінки споживача.

### Хід виконання

Продовжимо розвивати нашу модель, визначаючи характеристики та поведінку споживачів. Найкращий спосіб визначити поведінку – використати діаграму станів:

1. Діаграми станів є найдосконалішою конструкцією для опису поведінки, керованої подіями та часом. Для деяких об’єктів цей порядок подій і часу операцій настільки поширений, що ви можете найкраще охарактеризувати їх поведінку за допомогою діаграми переходів станів.

2. Діаграми станів мають стани та переходи. Стани діаграми станів є альтернативними, а це означає, що об’єкт може перебувати лише в одному стані. Виконання переходу може привести до зміни стану, що робить новий набір переходів активним. Стани діаграми станів можуть бути ієрархічними – вони можуть містити інші стани та переходи.

3. Один агент може мати кілька діаграм станів, які описують незалежні частини поведінки агента (рис. 3.1).

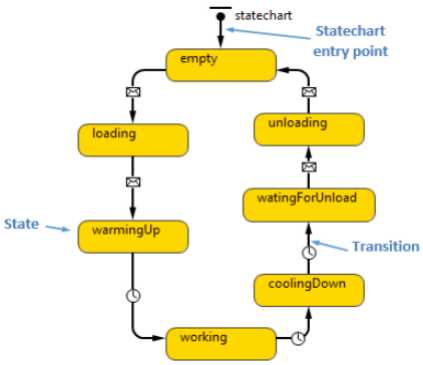



Рисунок 3.1 – Один агент

Ми визначимо поведінку споживача як послідовність двох станів:

– споживач у стані *PotentialUser* лише потенційно зацікавлений у покупці продукту;

– споживач у стані користувача придбав продукт:

1. У вікні *Projects* відкрийте діаграму споживача, натиснувши по ній двічі. Ви побачите графічну діаграму агента з анімаційною фігурою в центрі осі та параметром. Як дізнатися, який тип агента ви редагуєте? Оскільки наша модель має два типи агентів, ви можете запитати, який тип агента ви редагуєте в графічному редакторі.

*AnyLogic* вибирає вкладку типу агента, який ви відкрили в графічному редакторі, і виділяє її елемент у дереві проєктів (рис. 3.2).

Ви можете переміщатися між відкритими графічними діаграмами різних типів агентів, натискаючи назви вкладок (наприклад, *Main* and *Consumer* у прикладі нижче) (рис. 3.2).

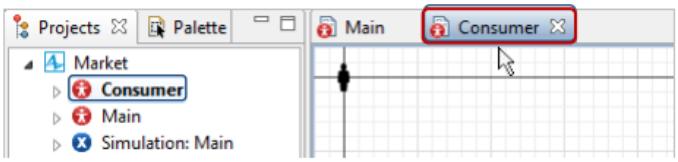



Рисунок 3.2 – Основний і Споживач

2. Почніть малювати діаграму станів із зображення двох станів. Відкрийте палітру *Statechart.*

3. Перетягніть точку входу в діаграму станів із палітри діаграми станів на діаграму споживача. Ви починаєте малювати діаграму станів, додаючи точку входу в діаграму станів. Точка входу визначає початок потоку керування діаграмою станів та ім’я діаграми станів (рис. 3.3).

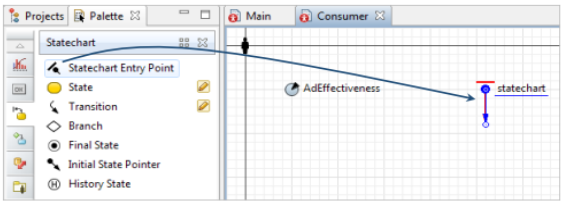



Рисунок 3.3 – Точка входу в діаграму станів

Будьте обережні: точку входу в діаграму стану легко сплутати з покажчиком початкового стану або переходом, оскільки вони схожі. Ви можете побачити, як *AnyLogic* виділив точку входу в діаграму станів червоним кольором. Це означає, що точка входу не пов'язана з жодним станом, а поточна діаграма станів недійсна. Додамо перший стан у діаграмі станів споживача.

4. Перетягніть стан із палітри діаграми станів на графічну діаграму та підімкніть його до точки входу діаграми станів (рис. 3.4).

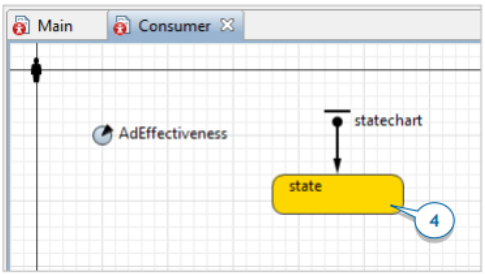



Рисунок 3.4 – Діаграма споживача

Переконайтеся, що ви малюєте діаграму станів на діаграмі *Consumer*, а не на *Main*.

5. Виберіть стан у графічному редакторі та змініть його властивості. Змініть назву блоку зі *State* на *PotentialUser*.

6. Використовуйте елемент керування *Fill color*, щоб змінити колір стану на лавандовий (рис. 3.5).

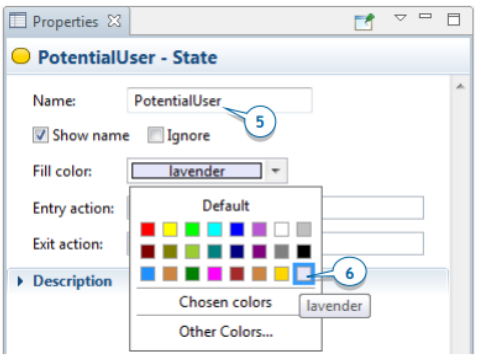



Рисунок 3.5 – Керування кольором заливки

7. Введіть наступний код *Java* у поле *Entry action* стану: *shapeBody.setFillColor(lavender)* (рис. 3.6).

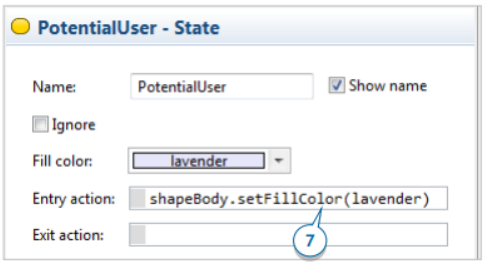



Рисунок 3.6 – Код *Java*

Помічник заповнення коду.

Ви можете скористатися помічником автозавершення коду, щоб не вводити повні назви елементів і функцій. Щоб відкрити помічника, оберіть потрібну позицію в полі редагування та натисніть Ctrl + пробіл (*Alt* + пробіл у *Mac OS*). У спливаючому вікні перелічено елементи моделі, які доступні в цьому контексті, наприклад змінні моделі, параметри або функції (рис. 3.7).

Перейдіть до назви елемента, який ви бажаєте додати, або введіть перші літери елемента, доки він не з’явиться у списку, і натисніть *Enter*, щоб вставити назву елемента у вікно редагування.

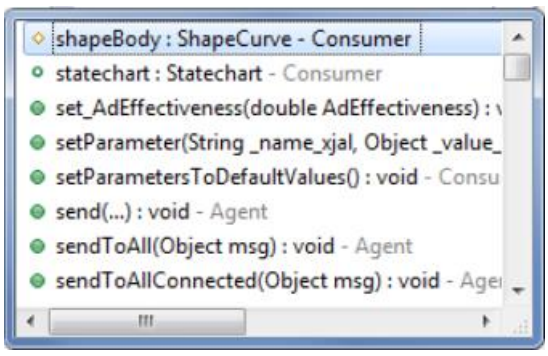



Рисунок 3.7 – Презентація споживача

Дія входу виконується, коли споживач переходить в інший стан. Цей код відображає зміну стану шляхом зміни кольору анімації споживача. Тут *shapeBody* – це ім’я форми анімації споживача, створеної майстром нового агента (якщо ви розгорнете гілку *Presentation branch* в дереві проєктів, ви

побачите фігуру *shapeBody* всередині групи осіб). Тут ми називаємо функцію *shapeBody*. Щоб отримати доступ до функції елемента, введіть назву елемента (*shapeBody*), введіть крапку, а потім скористайтеся функцією автозаповнення коду, щоб перелічити функції елемента, або виберіть назву функції зі списку (рис. 3.8).

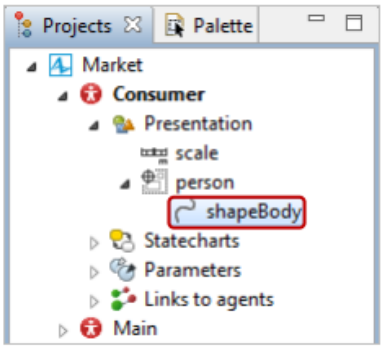



Рисунок 3.8 – Проєкти

*setFillColor*() – це одна зі стандартних функцій фігури, яка дозволяє динамічно змінювати колір заливки фігури. Потрібен лише один аргумент – новий колір.

8. Додайте інший стан у діаграму станів споживача (рис. 3.9).

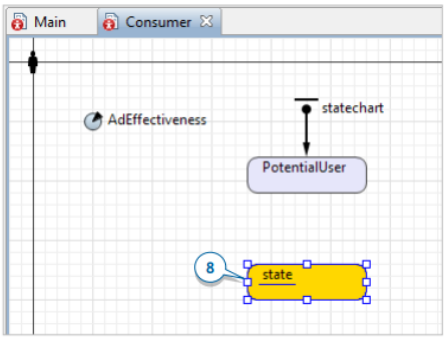



Рисунок 3.9 – Діаграма стану споживача

9. Змініть властивості стану, як ви робили раніше: «Ім’я: Користувач». Колір заливки – жовто-зелений.

Вхідна дія – *shapeBody.setFillColor(yellowGreen)* (рис. 3.10).

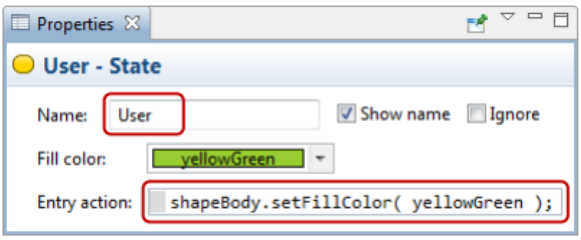



Рисунок 3.10 – Потенційний користувач

10. Намалюйте перехід від потенційного користувача до стану користувача, щоб змоделювати, як люди купують продукт і стають його користувачами. Для цього двічі клацніть на елементі *Transition* палітри *Statechart* (піктограма палітри елемента має змінитися), оберіть блок *PotentialUser*, а потім натисніть на блок *User* (рис. 3.11). Буде побудовано зв'язок між цими двома елементами.

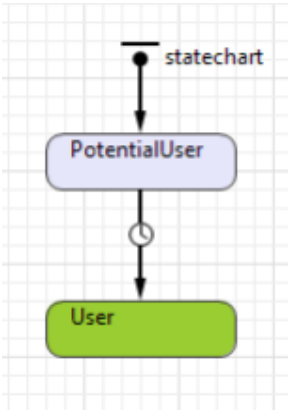



Рисунок 3.11 – Приклад побудови зав'язків між блоками

Переконайтеся, що перехід з'єднує стани. Якщо перехід не підімкнений, *AnyLogic* підсвічує його червоним.

11. Назвіть перехід *Add* для представлення «реклами» (рис. 3.12).

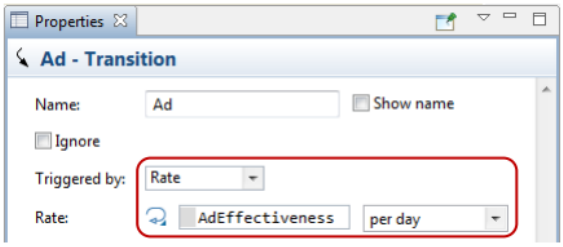



Рисунок 3.12 – AdEffectiveness

12. Установіть прапорець «Показати назву», щоб відобразити назву переходу на графічній діаграмі (рис. 3.12).

13. Перехід від стану *PotentialUser* до *User* моделюватиме, як реклама спонукає людину купити продукт. У списку «Запущено» клацніть «Рейтинг». У полі *Rate* введіть *AdEffectiveness*, а потім натисніть *per day* (рис. 3.12).

Можна бачити, що піктограма, намальована над переходом, змінилася. Цей знак показує тип тригера переходу. Щоб перемістити назву або піктограму переходу, виберіть перехід і за допомогою миші перетягніть відповідний елемент на нове місце (рис. 3.13).

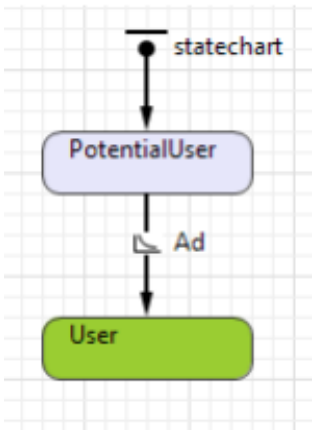



Рисунок 3.13 – Нове розташування

Типи тригерів переходу. Багато типів подій можуть ініціювати перехід. У наведеній нижче таблиці наведено типи тригерів переходів, а також піктограми, намальовані над переходами, щоб допомогти вам зрозуміти їхній тип тригерів (рис. 3.14).

| Transition trigger | Description |
|---|---|
| **Timeout** | Transition occurs after a specified time interval counted from the moment the statechart enters the "source" state of the transition. The timeout expression can be stochastic or deterministic.<br>Primary uses:<br>**Delay**: stay in a state for a given time, then leave.<br>**Timeout**: change state if other awaited events don't occur within the specified time interval. |
| **Rate** | Used to implement a sporadic state change with a known mean time. Acts in the same way as a timeout triggered transition, but the time interval is drawn from an exponential distribution parameterized with the given rate. For example, if the rate is 0.2 the timeouts will have mean values of 1/0.2 = 5 time units. |
| **Condition** | Transition monitors a specified Boolean condition and reacts when it becomes true. The condition is an arbitrary boolean expression and may depend on the states of any objects in the whole model with continuous as well as discrete dynamics.<br>*Please note that the condition is checked only when some events occur in the model. To ensure you do not miss the state switch moment, we recommend you add a cyclic event inside the agent and make it occur often enough not to miss the moment when the transition's condition becomes true.* |
| **Message** | Reacts to messages from other agents. The messages can model communication between people, commands given to a machine, etc. You can define the message template in the transition properties, but only the messages that match this template will trigger the transition. |
| **Arrival** | Reacts to arrival of this agent to its destination.<br>*Please note that the transition reacts only if the movement was initiated by calling the agent's function moveTo().* |

Рисунок 3.14 – Типи тригерів

Наш перехід запускається з указаною швидкістю. У нашому випадку, коли діаграма станів переходить у стан *PotentialUser*, виконується розіграш із експоненціального розподілу та встановлюється час очікування. Час прийняття для кожного споживача буде різним, хоча в середньому 1 % потенційних користувачів купуватимуть продукт у певний день.

14. Встановимо одиниці вимірювання часу моделі. Щоб налаштувати параметри моделі, перемикаємося із вкладки «Палітри» на «Проєкти», а потім обираємо елемент моделі в дереві (верхній об'єкт дерева, ринок). У вікні *Properties* виберіть дні як одиниці модельного часу (рис. 3.15).

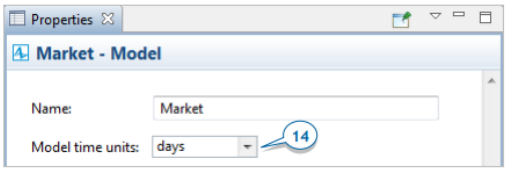



Рисунок 3.15 – Модельні одиниці часу

Модельний час. Модель одиниць часу.

Модельний час – це віртуальний (змодельований) час, який підтримує система моделювання *AnyLogic*. Час моделі не пов'язаний із реальним часом або годинником комп'ютера, хоча ви можете запустити модель у масштабі реального часу.

Щоб встановити співвідношення між модельним часом і часом реального світу, де живе модельована система, вам потрібно буде визначити одиниці часу. Ви повинні вибрати відповідну одиницю часу моделі для вашої моделі, близьку до типової тривалості роботи вашої моделі. Наприклад, моделі пішохідного потоку зазвичай використовують секунди, а системи обслуговування виробництва зазвичай використовують хвилини, але деякі глобальні економічні, соціальні та екологічні моделі, визначені в стилі системної динаміки, можуть використовувати місяці або навіть роки.

15. Запустіть модель. Популяція повинна поступово ставати зеленою – зміна, яка представляє ефект реклами – поки кожен споживач не купить продукт (рис. 3.16).


Рисунок 3.16 – Симуляція ринку

Коли ефекти реклами спонукають агента придбати продукт, стан агента *User* стає активним, виконується дія стану *Entry*, а колір форми анімації агента змінюється на жовто-зелений. Якщо більше людей купуватимуть продукт, анімація агента вашої моделі поступово стане зеленою.

Режими виконання моделі. Ви можете запустити модель *AnyLogic* у режимі реального або віртуального часу.

У режимі реального часу ви встановлюєте співвідношення між часом вашої моделі та реальним часом, вибираючи, скільки одиниць модельного часу дорівнює одній секунді фактичного часу. Ви зазвичай використовуєте режим реального часу, коли хочете, щоб ваша анімація виглядала реалістично.

У режимі віртуального часу модель працює на максимальній швидкості. Режим віртуального часу корисний, коли вам потрібно змоделювати свою модель протягом тривалого часу, і модель не вимагає від вас визначення співвідношення між одиницями модельного часу та секундами астрономічного часу. У режимі реального часу ви можете збільшити або зменшити швидкість виконання моделі, змінивши шкалу швидкості моделювання моделі. Наприклад, $x^2$ означає, що модель працює вдвічі швидше за вказану швидкість

моделі. Налаштувати швидкість виконання моделі можна в панелі керування вікна моделі.

16. Щоб налаштувати швидкість виконання моделі, клацніть на панелі інструментів кнопки *Slow down or Speed up buttons*. Якщо ви збільшите швидкість до 10 разів, то побачите, що швидкість, з якою населення стає зеленим, також збільшиться.

## Питання для самоконтролю

1. За допомогою яких дій можна змінити колір?

2. Які стани та переходи мають діаграми?

3. Як встановити співвідношення між модельним часом і часом реального світу, де перебуває модельована система?

## Практична робота № 4

## ДОДАВАННЯ ДІАГРАМИ ДЛЯ ВІЗУАЛІЗАЦІЇ РЕЗУЛЬТАТУ МОДЕЛІ

**Мета:** набути прикладних навичок щодо створення діаграм для перегляду результатів моделі.

### Хід виконання

Ми хочемо знати, скільки людей купили наш продукт у певний момент. Для цього визначимо функції, які підраховують наявних і потенційних користувачів нашого продукту, а потім додамо діаграму, щоб показати динаміку:

1. Спочатку визначте функцію для підрахунку потенційних користувачів. Щоб додати нову функцію, яка збирає статистику для агентів, відкрийте діаграму агента типу *Main*, виберіть споживачів сукупності агентів і перейдіть до розділу властивостей статистики.

2. Натисніть кнопку *Add statistics button* (рис. 4.1).

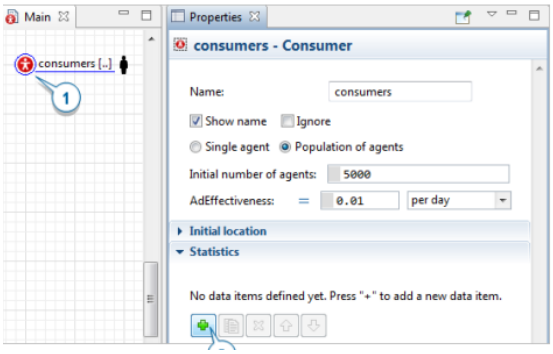



Рисунок 4.1 – Кнопка «Додати статистику»

Нам потрібно визначити, скільки агентів перебувають у стані *PotentialUser*.

3. Визначте функцію типу *Count* з іменем *NPotential*. Статистика типу *Count* повторює задану популяцію – у нашому випадку кількість агентів – для підрахунку тих, хто відповідає вибраній умові (рис. 4.2).

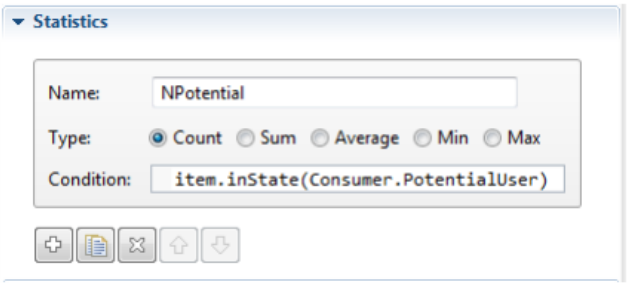



Рисунок 4.2 — Підрахунок із назвою *NPotential*

4. Введіть *item.inState* (*Consumer.PotentialUser*) як функцію *Condition*:

– *item* представляє агента, який перевіряється в ітерації;

– *inState()* – функція, яка перевіряє, чи активний вказаний стан діаграми станів;

– *PotentialUser* – назва стану, визначеного агентом, тому йому потрібен префікс типу агента *Consumer*.

5. Визначте другу статистичну функцію для обчислення кількості користувачів продукту. Назвіть його *NUser* і дозвольте йому підрахувати кількість агентів, що відповідають умові *item.inState(Consumer.User)*. Ви можете скопіювати іншу статистичну функцію, натиснувши кнопку дублювати та змінивши її назву й умову. Далі додамо діаграму, щоб показати статистику, яку збирають ці функції, і відобразити динаміку процесу впровадження (рис. 4.3).

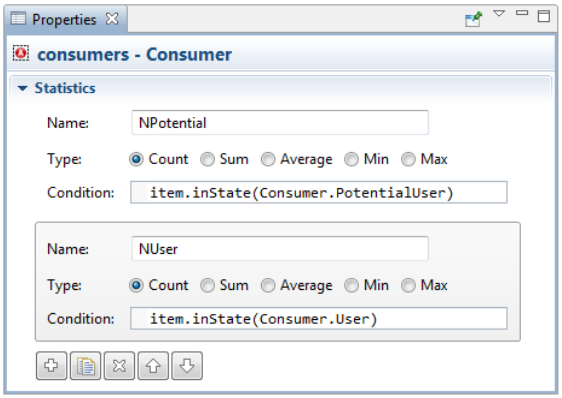



Рисунок 4.3 – Потенційний користувач

6. Відкрийте палітру аналізу(*Analysis*) та перетягніть діаграму стека часу (*Time Stack Chart*) з палітри аналізу на головну діаграму, щоб створити діаграму, яка відображатиме динаміку користувачів і потенційних користувачів. Збільшіть діаграму стеку часу, як показано на рисунку 4.4.

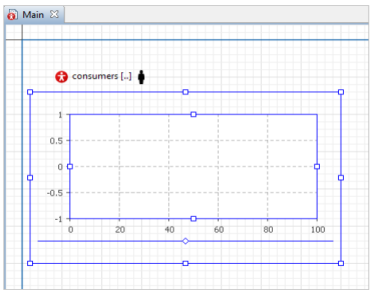



Рисунок 4.4 – Основна діаграма

*AnyLogic* надає кілька діаграм, які можна використовувати для візуалізації даних, які створює ваша модель. Ви можете знайти їх на панелі *Analysis palette in the Charts section*.

Bar *Chart Displays* елементи даних у вигляді стовпчиків, вирівняних на одному кінці. Розміри стовпчиків пропорційні значенням відповідних елементів даних (рис. 4.5).

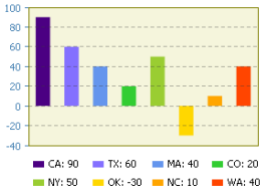



Рисунок 4.5 – Відображення гістограми

*Stack Chart Displays* вносить кілька елементів даних у загальну суму у вигляді стовпчиків із накопиченням. Розміри стовпчиків пропорційні значенням відповідних елементів даних (рис. 4.6).

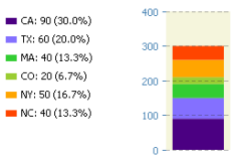



Рисунок 4.6 – Відображення стекової діаграми

*Pie Chart Displays* внесить кілька елементів даних у загальну суму як секторів кола. Дуги секторів пропорційні відповідним значенням елементів даних (рис. 4.7).

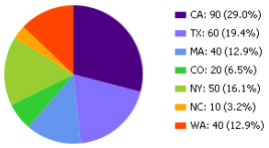



Рисунок 4.7 – Відображення секторної діаграми

Діаграма відіграє роль фазової діаграми. Кожен набір даних є набором пар значень. Графік відображає значення Y для набору даних, нанесених на графік із відповідними значеннями X. Значення X відображаються на осі X, значення Y – на осі Y. Графік може відображати декілька наборів даних одночасно (рис. 4.8).

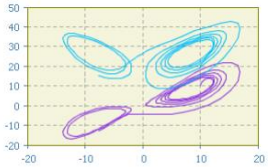



Рисунок 4.8 – Відображення на графіку значення Y

Графік часу відображає історію кількох елементів даних за останній часовий горизонт. Залежно від типу інтерполяції лінія між двома вибірками даних інтерполюється лінійно або зберігає попереднє значення до наступного (рис. 4.9).

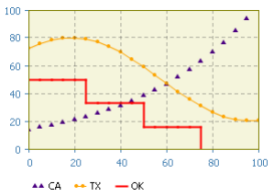



Рисунок 4.9 – Відображення графіка часу

Діаграма стеку часу. Відображає історію вкладу ряду елементів даних у загальну суму протягом останнього часового горизонту у вигляді областей накопичення. Значення постійно накладаються одне на інше з першим доданим елементом даних внизу (рис. 4.10).

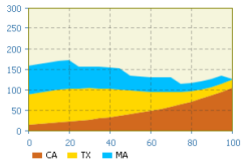



Рисунок 4.10 – Відображення діаграми стеку часу

Кольорова діаграма часу відображає тенденцію ряду наборів даних за останній час у вигляді смужок горизонтальних смуг різних кольорів (колір залежить від значення даних). Якщо умова оцінюється як істинна, колір смуги відповідатиме кольору, який ви визначили для цієї умови. Використовуйте діаграму, щоб візуалізувати зміну стану агента з часом, наприклад зайнятий / неактивний (рис. 4.11).

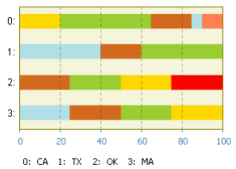



Рисунок 4.11 – Відображення кольорової діаграми часу

Гістограма відображає статистику, зібрану об’єктами даних гістограми. Гістограми також масштабуються вздовж осі Y, тому найвища смуга гістограми займає всю висоту зображення. Ви також можете вибрати відображення смуг *PDF*, лінії *CDF* і середнього розташування (рис. 4.12).

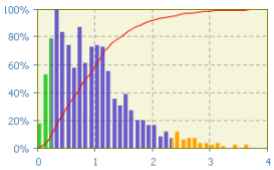



Рисунок 4.12 – Відображення гістограми

*Гістограма 2D*-колекція двовимірних гістограм. Кожна гістограма зображується як ряд прямокутних кольорових плям, що відображають значення *PDF* або конверт у відповідних X і Y. Осі X і Y діаграми завжди масштабуються відповідно до всіх гістограм (рис. 4.13).

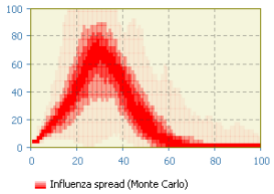



Рисунок 4.13 – Відображення *Histogram2D*

Додайте два елементи даних для відображення діаграми. Тут ми назвемо наші статистичні функції *NUser* і *NPotential*, які ми визначили для сукупності споживачів на попередньому кроці.

7. Використайте *Click Add data item*, щоб додати статистику, яку ви хочете відобразити на діаграмі (рис. 4.14).

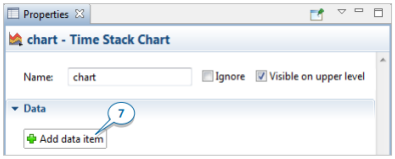



Рисунок 4.14 – Додати елемент даних

8. Змініть властивості елемента даних:

– назва: *Users – the data item's title*;

– колір: *yellowGree;*

– значення: *consumers.NUser().*

Ім'я популяції нашого агента – споживачі, а NUser() – функція статистики, яку ми визначили для цієї популяції (рис. 4.15).

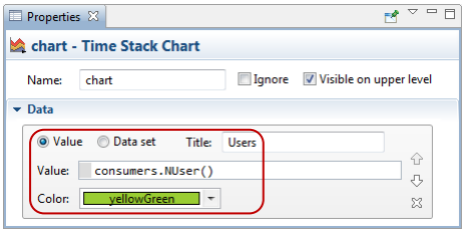



Рисунок 4.15 – Зміна властивостей елемента даних

9. Додайте ще один елемент даних:

– назва: *Potential users;*

– колір: *lavender* (рис. 4.16);

– цінність: *consumers. NPotential.*

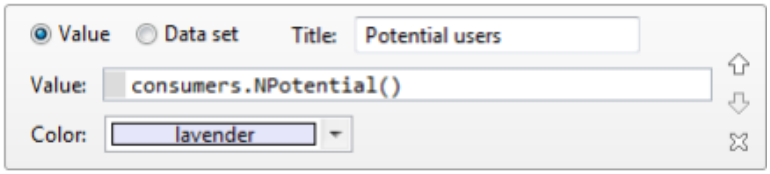



Рисунок 4.16 – Додавання ще одного елемента даних

Налаштування шкали часу діаграми.

1. Графіки з історією (діаграма часу, діаграма стеку часу, кольорова діаграма часу) дають змогу регулювати часову шкалу.

2. Ви налаштовуєте часовий діапазон часової діаграми за допомогою властивості «Часове вікно». Оскільки часові діаграми відображають лише обмежену кількість вибірок даних у певний момент, переконайтеся, що у вас є достатня кількість вибірок для вибраного часового вікна.

3. Якщо ви запускаєте свою модель і ваша діаграма схожа на малюнок нижче, вам потрібно збільшити кількість зразків даних, які відображає діаграма, або зменшити часове вікно діаграми (рис. 4.17).

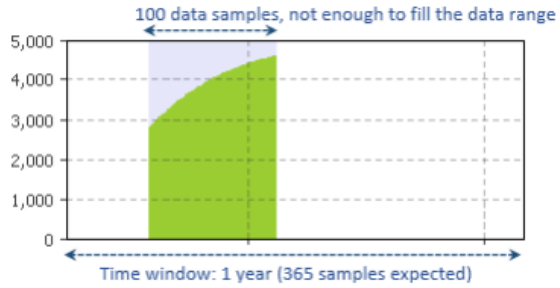



Рисунок 4.17 – Шкала часу діаграми

Оскільки ми хочемо показати діапазон за один рік, нам потрібно налаштувати параметри діаграми:

1. Перейдіть до розділу «Масштаб» і встановіть «Часове вікно» один рік (рис. 4.18).

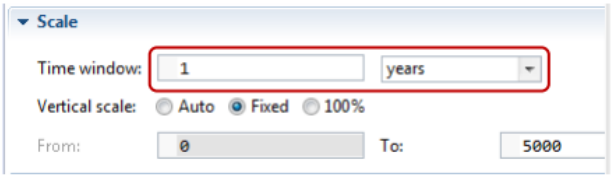



Рисунок 4.18 – Розріз шкали

2. Оскільки наша діаграма показуватиме статистику для покупців, а наша модель містить 5 000 споживачів, встановіть вертикальний масштаб діаграми на фіксований і введіть 5 000 у полі «Кому» (рис. 4.19).

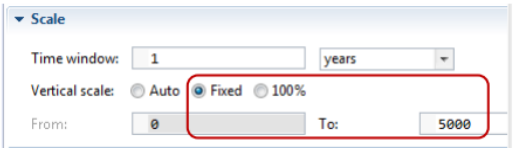



Рисунок 4.19 – Вертикальний масштаб

3. Коли часове вікно встановлене, змініть максимальну кількість зразків даних, які відображається на діаграмі, перейшовши до розділу «Оновлення даних» і встановивши параметр «Відображати» до 365-ти останніх зразків. Оскільки ми будемо додавати одну вибірку даних щодня, 365 вибірок даних є ідеальною кількістю для одного року (рис. 4.20).

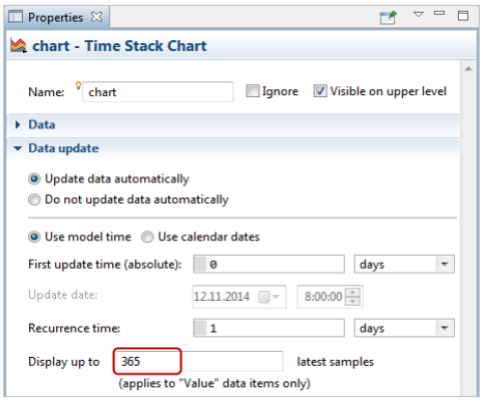



Рисунок 4.20 – Дисплей вгору

4. Перейдіть до властивостей вигляду діаграми стеку часу та встановіть для нього відображення дати моделі (тільки дата) біля осі часу (рис. 4.21).

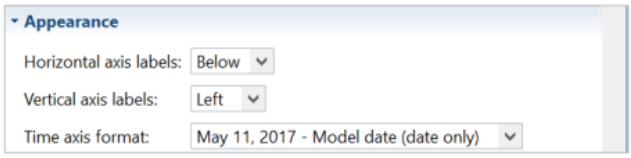



Рисунок 4.21 – Властивості зовнішнього вигляду

Форматування позначок часу в мітках діаграми часу діаграми з історією можуть відображати модельні дати в мітках осі часу (x–), і ви можете відформатувати мітки часу, вибравши один із запропонованих форматів. Налаштуйте формат мітки часу у властивості «Формат осі часу» (розміщується на діаграмі розділу *Appearance section*). У розділі нижче показано декілька прикладів форматів позначок часу (рис. 4.22).

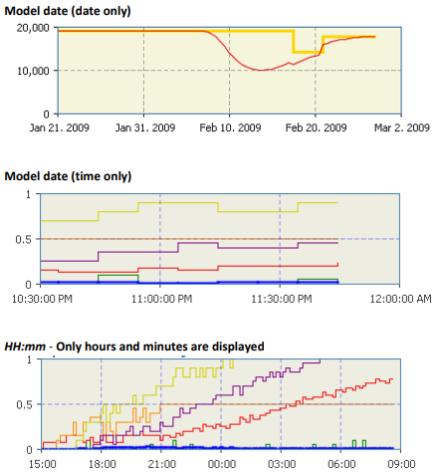



Рисунок 4.22 – Формати позначок часу

5. На головній діаграмі перемістіть подання сукупності агентів споживачів праворуч (рис. 4.23).

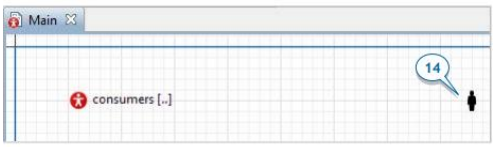



Рисунок 4.23 – Основна діаграма

**Питання для самоконтролю**

1. Які відомості надає кольорова діаграма часу?
2. Що відображає гістограма «2D-колекція двовимірних гістограм»?
3. Як додати функцію для підрахунку потенційних користувачів?

## Практична робота № 5
## ДОДАВАННЯ ЕФЕКТУ «САРАФАННОГО РАДІО»

**Мета:** набути практичних навичок щодо додавання ефекту «сарафанного радіо».

### Хід виконання

На цьому етапі ми змоделюємо те, що часто називають ефектом «сарафанного спілкування» – спосіб, у який люди переконують інших придбати наш продукт:

1. Дозвольте людям контактувати один з одним. У нашій моделі споживач контактує в середньому з однією особою щодня.

2. Поточні користувачі нашого продукту можуть впливати на потенційних користувачів під час цих зустрічей. Ми визначимо ймовірність того, що потенційний користувач придбає продукт, як *AdoptionFraction* = 0,01. Розвинемо

логіку моделі, додавши два параметри споживача: *ContactRate* і *AdoptionFraction*.

3. У дереві проєктів відкрийте діаграму *Consumer*, двічі клацнувши на *Consumer*.

4. Додайте параметр для визначення середньої кількості щоденних контактів споживача. Перетягніть параметр із палітри агента на діаграму.

5. Назвіть параметр *ContactRate*.

6. Норма становить один контакт на день, тому введіть 1 як значення параметра за замовчуванням.

7. Додайте ще один параметр – *AdoptionFraction,* щоб визначити вплив людини на інших, число, яке ми виразимо, як відсоток людей, які використовуватимуть продукт після того, як вони вступлять у контакт із споживачем. Залишіть тип параметра за замовчуванням – *double* і встановіть значення за замовчуванням 0,01. Діаграма споживача повинна виглядати так (рис. 5.1).

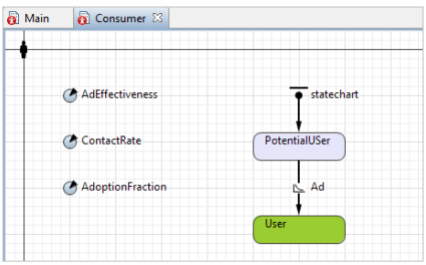



Рисунок 5.1 – Діаграма споживача

Тепер ми дозволимо нашим агентам взаємодіяти. Це усні дискусії, які переконають відсоток споживачів купити продукт. Взаємодія агента *AnyLogic* підтримує унікальний для агентного моделювання механізм зв’язку – передачу повідомлень:

– агент може надіслати повідомлення окремому агенту або групі агентів;

– повідомлення може бути об’єктом будь-якого типу або будь-якої складності, включаючи текстовий рядок, ціле число, посилання на об’єкт або структуру з кількома полями;

– щоб надіслати повідомлення іншому агенту, використайте певну функцію агента. У наведеній нижче інформації перелічені функції, які найчастіше використовуються для надсилання повідомлень від одного агента до іншого: *sendToAll*( msg ) надсилає повідомлення всім агентам однієї групи; *sendToRandom*( msg ) надсилає повідомлення одному випадково обраному агенту з тієї самої популяції; send( msg, agent ) надсилає повідомлення заданому агенту (ви передаєте посилання на агента-одержувача як другий аргумент функції). У нашій моделі лише користувачі, які перебувають у стані *User*, надсилатимуть повідомлення. Найкращий спосіб визначити діяльність, яку виконує агент, перебуваючи в певному стані – тобто діяльність, яку він виконує без виходу з поточного стану – це використовувати внутрішній перехід;

– відкрийте діаграму споживача та збільшіть стан користувача відповідно до внутрішнього переходу, який ми намалюємо всередині стану на наступному кроці;

– намалюйте внутрішній перехід всередині стану користувача. Щоб намалювати перехід, подібний до поданого нижче, перетягніть перехід із палітри стану всередину стану, щоб початкова точка переходу розміщувалася на кордоні держави. Після цього кінцеву точку переходу можна перенести на іншу точку державного кордону. Для додавання виступаючої точки двічі клацніть на переході (рис. 5.2).

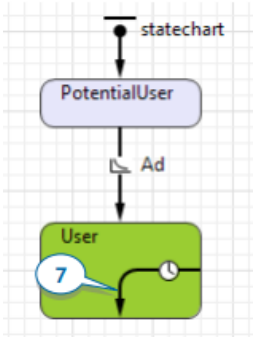



Рисунок 5.2 – Подвійне клацання на переході

Внутрішні та зовнішні переходи поводяться по-різному, тому ви повинні переконатися, що ваш новостворений перехід розміщується всередині стану.

Характеристики внутрішніх переходів:

– внутрішній перехід – це циклічний перехід, який розміщується всередині стану. Початкова і кінцева точки переходу розміщуються на державному кордоні;

– оскільки внутрішній перехід не виходить із охоплюючого стану, він не виводить діаграму станів із цього стану. Ні дії виходу, ні дії входу не виконуються, якщо відбувається перехід, і поточний простий стан у стані не виходить.

8. Змініть властивості переходу. Цей перехід відбудеться з указаною швидкістю *ContactRate* (використовуйте автозавершення коду замість введення повної назви параметра). Назвіть перехід *Contact* і встановіть для нього відображення свого імені (рис. 5.3).

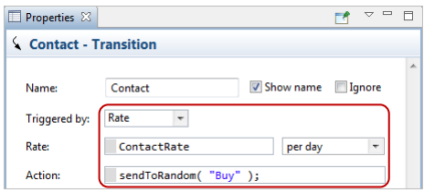



Рисунок 5.3 – Контакт

9. Укажіть дію, яка буде виконана під час запуску цього переходу (використовуйте автозавершення коду, щоб написати код): *sendToRandom*(«Buy»).

Оскільки ми хочемо, щоб користувачі нашого продукту спілкувалися з потенційними користувачами, ми налаштуємо циклічний перехід у стан «Користувач». Кожного разу, коли відбувається перехід, код *sendToRandom*(«Buy») змушує споживача випадково вибирати іншого агента та надсилати йому текстове повідомлення «Купити». Якщо агент, який отримує повідомлення, є потенційним користувачем (тобто якщо агент-одержувач перебуває у стані *PotentialUser*), стан агента-одержувача зміниться на *User*.

Давайте цей перехід.

10. Намалюйте інший перехід від стану *PotentialUser* до стану користувача та назвіть його *WOM*. Цей перехід моделюватиме покупки, спричинені усною інформацією.

11. Змініть властивості переходу:

– у списку *Triggered* клацніть на *Message;*

– в області переходу *Fire* виберіть *On specific message;*

– у полі *Message field* введіть *Buy.*

Оскільки ми знаємо, що не кожен контакт є успішним – тобто контакт може не переконати потенційного користувача купити наш продукт – ми використовуватимемо *AdoptionFraction*, щоб зробити успішні контакти менш поширеними. Вкажіть *Guard* переходу: *randomTrue* (*AdoptionFraction*) (рис. 5.4; 5.5).

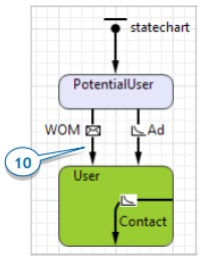



Рисунок 5.4 – Частка прийняття

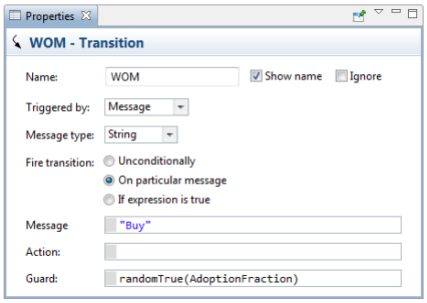



Рисунок 5.5 – Вказівка *Guard* переходу

Охоронці в переходах.

Коли діаграма станів переходить у простий стан, тригери всіх вихідних переходів збираються, і діаграма станів починає чекати, коли відбудеться будь-який із них.

Коли відбувається тригерна подія, оцінюється захист відповідного переходу. Якщо *Guard* істинне, перехід може бути здійснено (хоча альтернативні одночасні події можуть скинути тригер). Цей алгоритм оцінки захисника називається «захисники-після-тригери». Це останній крок у моделюванні маркетингу «з уст в уста». *AnyLogic* пересилає повідомлення від агента до діаграми станів, і, якщо діаграма станів перебуває в стані *PotentialUser*, це спричиняє негайний перехід до стану користувача. Якщо діаграма станів перебуває в будь-якому іншому стані, вона ігноруватиме повідомлення.

12. У вікні *Projects* ви можете побачити зірочку біля елемента моделі, який показує, що ваша модель має незбережені зміни. На панелі інструментів натисніть *Save*, щоб зберегти модель (рис. 5.6).

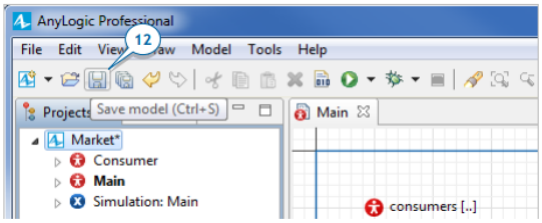



Рисунок 5.6 – Збереження моделі

13. Запустіть модель. Насичення ринку має відбуватися швидше. На графіку показано добре відому S-подібну криву прийняття продукту (рис. 5.7).

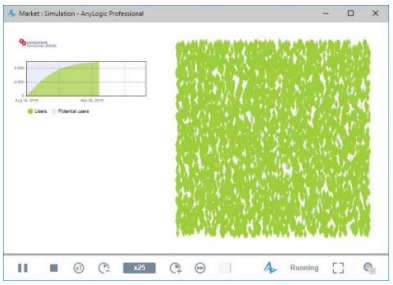


Рисунок 5.7 – S-подібний продукт

**Питання для самоконтролю**

1. Як намалювати внутрішній перехід всередині стану користувача?
2. За допомогою яких команд можна змінити властивості переходу?
3. Про що свідчить зірочка у вікні *Projects*?

## Практична робота № 6
## РОЗГЛЯД ВИКИДУ ПРОДУКЦІЇ

**Мета:** набути практичних навичок щодо утилізації продукції.

### Хід виконання

На цьому етапі ми змоделюємо відмову від продукту.

Припустимо, що середня тривалість активного використання нашого продукту становить шість місяців.

Коли користувач викидає або споживає продукт, йому знадобиться заміна. Ми будемо моделювати поведінку повторних покупок, припускаючи, що користувачі стають потенційними користувачами, коли вони відкидають або споживають свої перші одиниці (тобто коли користувач повертається до стану потенційного користувача):

1. Відкрийте діаграму споживача та додайте параметр *DiscardTime* (рис. 6.1).

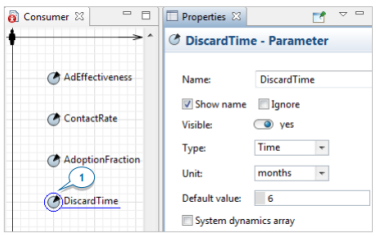



Рисунок 6.1 – Діаграма споживача

2. Цей параметр визначатиме термін служби нашого продукту. Виберіть *Time* як тип параметра, клацніть на місяці у списку «Одиниця» та введіть 6, як значення за замовчуванням.

3. Намалюйте перехід від стану *User* до *PotentialUser*, щоб змоделювати відмову від продукту. Щоб намалювати перехід із помітними точками, як показано на рисунку 6.2, двічі клацніть на елементі *Transition* палітри *Statechart.* Це має змінити піктограму елемента на палітрі на ), клацніть на вихідному стані

переходу *User* та у місцях з помітною точкою і на цільовому стані PotentialUser (рис. 6.2).

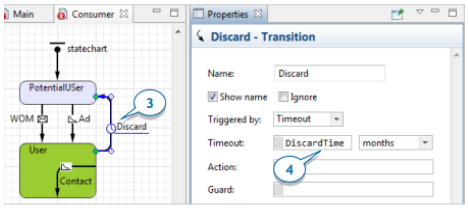
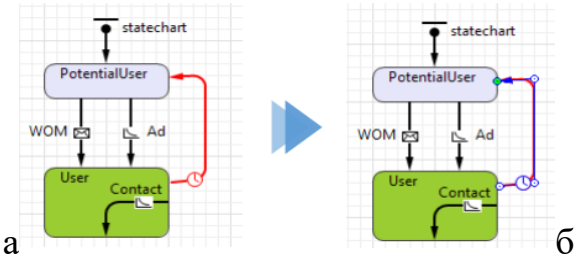



Рисунок 6.2 – Потенційний користувач

4. Назвіть перехід *Discard* і встановіть, щоб він запускався через постійний час очікування *DiscardTime*. У списку праворуч клацніть на місяці.

*AnyLogic* використовує червоне виділення, щоб привернути вашу увагу до переходів (рис. 6.3, а), де кінцева точка не пов'язана зі станом. Щоб знайти помилку, виберіть перехід, і з'єднані точки будуть виділені блакитним кольором (рис. 6.3, б, підімкнення до *PotentialAdopter*). Якщо *AnyLogic* не висвітлює початкову точку переходу в *User*, потрібно вручну перемістити цю точку в стан для встановлення підімкнення та виправити помилку (рис. 6.3).




а б

Рисунок 6.3 – AnyLogic

Виправлення помилок введення. Типовою є помилка в назві елемента моделі. Імена *AnyLogic* чутливі до регістру, а це означає, що введення *Discardtime* (замість *DiscardTime*) у властивості елемента моделі призведе до такої помилки (рис. 6.4).

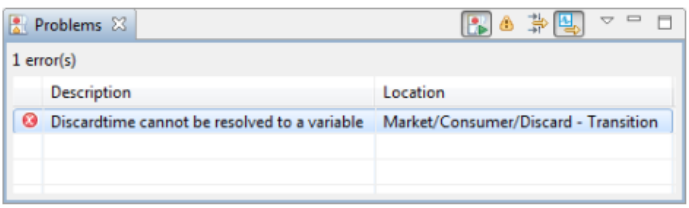



Рисунок 6.4 – Уведення *Discardtime* замість DiscardTime

Щоб виправити помилку, двічі клацніть по ній у вікні проблем. Якщо помилка графічна, *AnyLogic* виділить елемент, який спричинив помилку, у графічному редакторі. Якщо помилка у властивості елемента, *AnyLogic* відкриє властивості елемента та відобразить поле, у якому виникла проблема. Наша робота з моделювання вилучення продуктів завершена, будь-які повторні вилучення приведуть до необхідності зробити заміну.

5. Запустіть модель і спостерігайте, як вилучення впливає на динаміку впровадження. І після того, як наш продукт наситить ринок, ви помітите випадкове вилучення продукту (рис. 6.5).

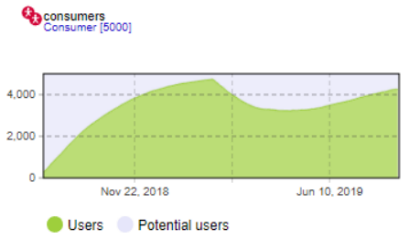



Рисунок 6.5 – Утилізація продукту

### Питання для самоконтролю

1. На що необхідно натиснути для переходу із помітними точками?
2. Яким має бути алгоритм відмови від продукту?
3. Перелічіть чинники, що впливають на відмову від товару?

## Практична робота № 7
## ВРАХУВАННЯ ЧАСУ ДОСТАВКИ

**Мета:** набути практичних навичок щодо врахування часу доставки продукції.

### Хід виконання

Наша модель передбачає, що продукт завжди доступний, а перехід від *PotentialUser* до *User* є безумовним і негайним. Поліпшимо модель, додавши стан до діаграми станів, який відображає проміжок часу між рішенням агента придбати продукт і моментом, коли він його отримає:

1. Підготуйте місце для іншого стану між *PotentialUser* і *User*, перемістивши стан *User* у нижню частину екрана (рис. 7.1).

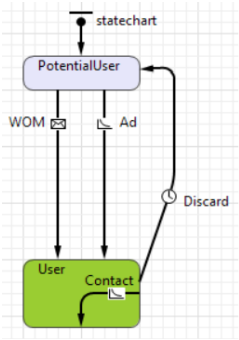



Рисунок 7.1 – Стан користувача

2. Відімкніть стан користувача від переходів. Виберіть переходи *WOM* і *Ad*, перемістіть їхні кінцеві точки до верхньої частини екрана та від'єднайте перехід *Discard* від *PotentialUser*. Після цього ви помітите, що роз'єднані переходи позначені червоним кольором (рис. 7.2).

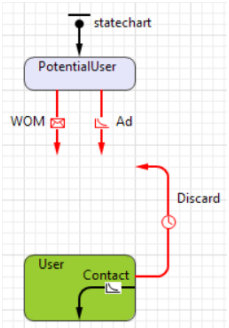



Рисунок 7.2 – Відімкнення користувача

3. Додайте ще один стан із палітри *Statechart* до середини діаграми станів споживача та назвіть його *WantsToBuy*. Споживачі в цьому стані вирішили придбати продукт, але не зробили цього (рис. 7.3).

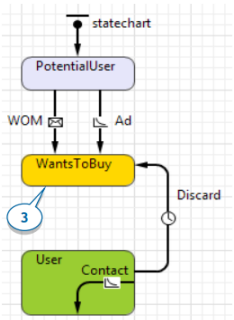



Рисунок 7.3 – Палітра *Statechart*

4. Знову підімкніть переходи до середнього стану: переходи *WOM*, *Ad* і *Discard* тепер мають закінчуватися у стані *WantsToBuy*.

5. Змініть *WantsToBuy* подібно до інших станів (колір заливки: золотий, дія введення: *shapeBody.setFillColor(gold)* (рис. 7.4).

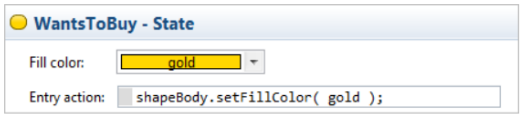



Рисунок 7.4 – Заміна *WantsToBuy*

6. Додайте перехід від *WantsToBuy* до стану користувача, щоб змоделювати доставку товару та назвати її *Purchase* (рис. 7.5).

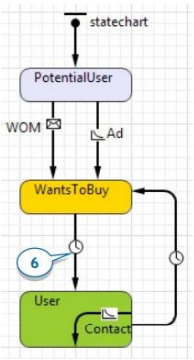



Рисунок 7.5 – Додавання переходу

7. Припустимо, що користувачу зазвичай потрібно два дні, щоб отримати продукт. Це означає, що як тільки діаграма стану споживача перейде в стан *WantsToBuy*, вона перейде до стану *User* із дводенною затримкою. Зважаючи на це, встановіть два дні тайм-ауту для переходу покупки (рис. 7.6):

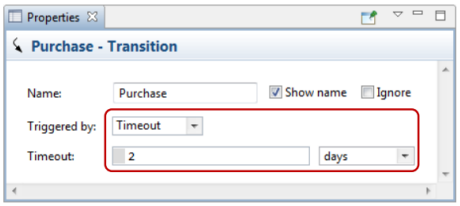



Рисунок 7.6 – Перехід покупки

8. Визначте ще одну статистичну функцію для підрахунку ринкового попиту на продукт. У редакторі *Main* клацніть по споживачах, перейдіть до розділу властивостей статистики та додайте елемент статистики *NWantToBuy* з умовою *item.inState(Consumer.WantsToBuy)* (рис. 7.7).

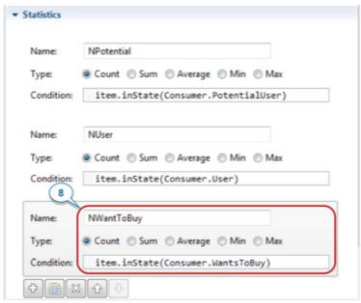



Рисунок 7.7 – Розділ властивостей статистики

9. На головному екрані виберіть діаграму стеку часу та додайте ще один елемент даних, який буде відображатися разом із діаграмою: *consumers.NWantToBuy()* із заголовком *Want to buy* золотим кольором (рис. 7.8).

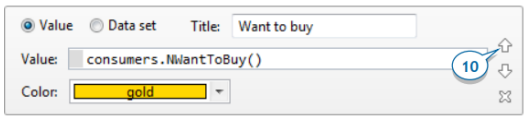



Рисунок 7.8 – На головному екрані

10. Зробіть щойно визначений елемент даних другим у списку, вибравши розділ елемента та натиснувши кнопку *up* (рис. 7.9).

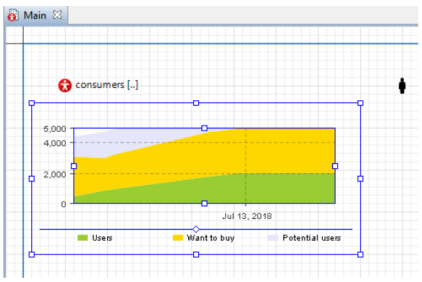



Рисунок 7.9 – Розділ елемента

11. Запустіть модель, і ви помітите, що *AnyLogic* відображає кількість споживачів, які очікують на продукт, жовтим кольором (рис. 7.10).

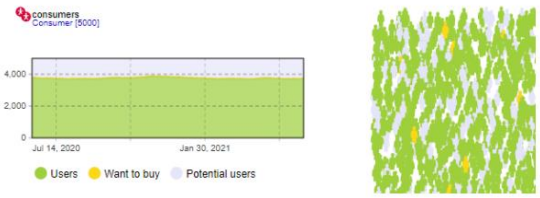



Рисунок 7.10 – Дисплеї *AnyLogic*

### Питання для самоконтролю

1. Як можна відімкнути стан користувача для переходів?

2. Як можна змоделювати доставку товару?

3. За допомогою якої статистичної функції розраховують ринковий попит на продукт?

## Практична робота № 8
## ІМІТАЦІЯ СПОЖИВЧОГО НЕТЕРПІННЯ

**Мета:** набути практичних навичок щодо імітації споживчого нетерпіння.

### Хід виконання

Наша модель має враховувати різну кількість часу, який споживачі готові витратити на доставку свого продукту. Якщо час доставки перевищує час, який споживач готовий чекати, він перегляне своє рішення та повернеться до потенційного користувача, а не до того, хто хоче купити. Почнемо з визначення двох параметрів в *Main*: максимальний час доставки товару (25 днів) і максимальний час очікування споживача (7 днів):

1. Відкрийте діаграму типу головного агента.

2. Оскільки ми не хочемо, щоб вікно моделі відображало параметри моделі під час виконання, ми можемо розмістити їх за межами стандартної області відображення вікна моделі. На головному екрані вікно моделі зображено синьою

прямокутною рамкою. Елементи всередині рамки будуть видимі під час виконання моделі, але їх можна приховати, перемістивши полотно графічної діаграми вліво і розмістивши два параметри (рис. 8.1). Щоб перемістити полотно графічної діаграми, утримуйте праву кнопку миші під час переміщення миші.


Рисунок 8.1 – Діаграма типу основного агента

3. Налаштуйте параметри. *MaxWaitingTime* визначає максимальний час, протягом якого споживач чекатиме продукт (у цьому випадку сім днів) (рис. 8.2).

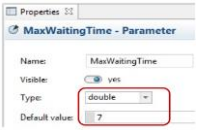



Рисунок 8.2 – Налаштування параметра *MaxWaitingTime*

4. Встановіть для іншого параметра *MaxDeliveryTime* значення 25 днів, щоб відобразити наше припущення, що доставка продукту може зайняти до 25 днів (рис. 8.3).

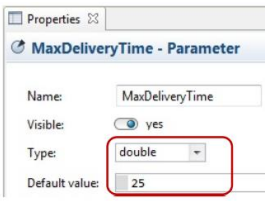



Рисунок 8.3 – Налаштування параметра *MaxDeliveryTime* (до 25 днів)

Припускаємо, що доставка продукту займає від одного до 25 днів, у середньому – два дні. Маючи це на увазі, змінимо час доставки з фіксованого дводенного періоду на стохастичний вираз, який описує цю модель. Функції розподілу ймовірностей.

Повний список дистрибутивів *AnyLogic* можна знайти у розділі довідки програми (рис. 8.4).

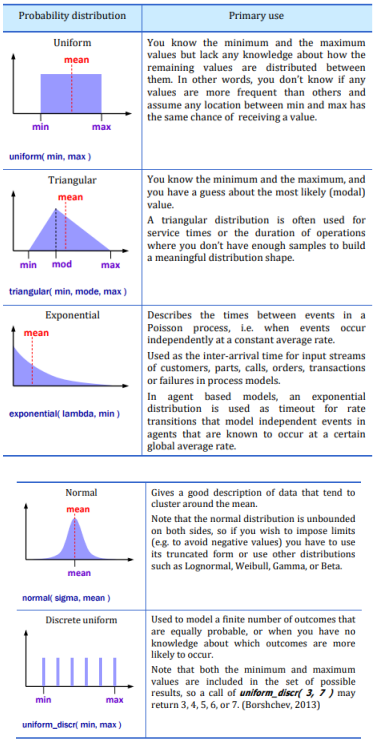



Рисунок 8.4 – Розділ довідки

Трикутний розподіл ймовірностей є найпростішим способом визначення необхідної моделі часу.

5. Відкрийте діаграму споживача та виберіть перехід покупки. Змінити вираз часу очікування переходу можна за допомогою майстра й вибрати функцію розподілу та вставити назву функції у властивість. Щоб замінити існуюче значення, виберіть за допомогою миші наявний вираз *Timeout* (рис. 8.5).

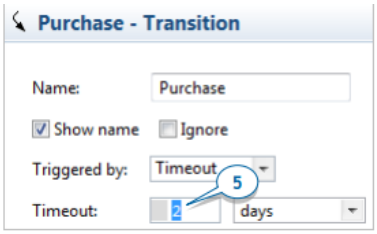



Рисунок 8.5 – Вираз часу очікування

6. Натисніть кнопку панелі інструментів «Вибрати розподіл ймовірностей» (рис. 8.6).

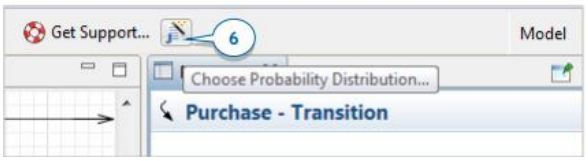



Рисунок 8.6 – Кнопка панелі інструментів

7. Ви побачите діалогове вікно «Вибір розподілу ймовірностей» (рис. 8.7).

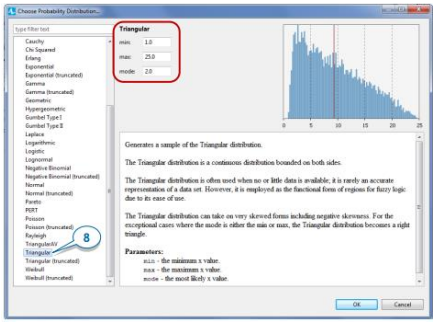



Рисунок 8.7 – Діалогове вікно

8. У вікні «Вибрати опис ймовірності» можна переглянути список підтримуваних розповсюджень, і ви можете клацнути на будь-якому імені у списку, щоб переглянути опис розповсюдження. Виберіть у списку трикутник. Встановіть параметри *min, max і mode* рівними 1, 25, 2 відповідно. У верхньому правому кутку ви побачите *pdf*-файл, миттєво створений для розповсюдження із зазначеними параметрами. Натисніть *OK*, коли закінчите.

9. Ви побачите вираз *triangular* (1, 25, 2), автоматично вставлений, як значення часу очікування. Змінимо рядок на *triangular(1, main.MaxDeliveryTime*, 2). Тут *main* показує, як ми отримуємо доступ до агента *Main* з агента споживача.

10. Намалюйте останній перехід *CantWait*, який переходить від стану *WantsToBuy* до стану *PotentialUser*. Цей перехід буде моделювати, як нетерпіння

споживача змушує його змінити своє рішення про покупку, і діаграма споживача виглядатиме так (рис. 8.8):

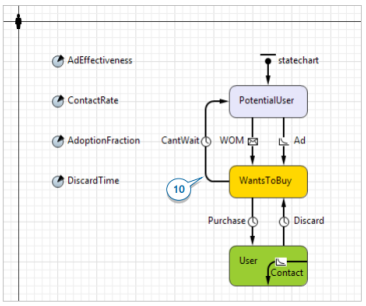



Рисунок 8.8 – Діаграма споживача

11. Змініть властивості переходу, щоб він ініціював час очікування, який дорівнює *triangularAV(main.MaxWaitingTime, 0,15)* днів (рис. 8.9).

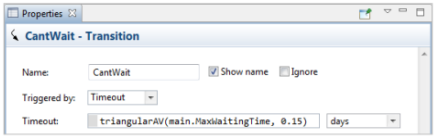



Рисунок 8.9 – Час очікування

Замість встановлення максимального часу очікування, що дорівнює постійному *MaxWaitingTime*, ми припускаємо, що він дотримується трикутного розподілу із середнім тижневим значенням і можливим відхиленням до 15 %.

Можна легко визначити максимальний час очікування та максимальний час доставки як постійні параметри, але нам потрібно динамічно змінювати ці числа та бачити, як ці зміни впливають на поведінку системи. Один із способів додати інтерактивність нашій моделі – це додати елементи керування та зв’язати їх із параметрами моделі.

Елементи керування.

Елементи керування *AnyLogic* можуть допомогти вам додати інтерактивності моделі. Ви можете використовувати їх для налаштування параметрів перед виконанням моделі та змінювати модель на ходу. Контроль може запускати код або вносити зміни в параметри моделі. Ви також можете

пов’язати довільну дію, таку як виклик функції, планування події, надсилання повідомлення або зупинка моделі з елементом керування. Дія виконується кожного разу, коли користувач торкається елемента керування. Значення елемента керування зазвичай доступне як значення в полі коду дії елемента керування, а також повертається функцією елемента керування *getValue()*. У наведеній нижче таблиці коротко описано кожен елемент керування (рис. 8.10).

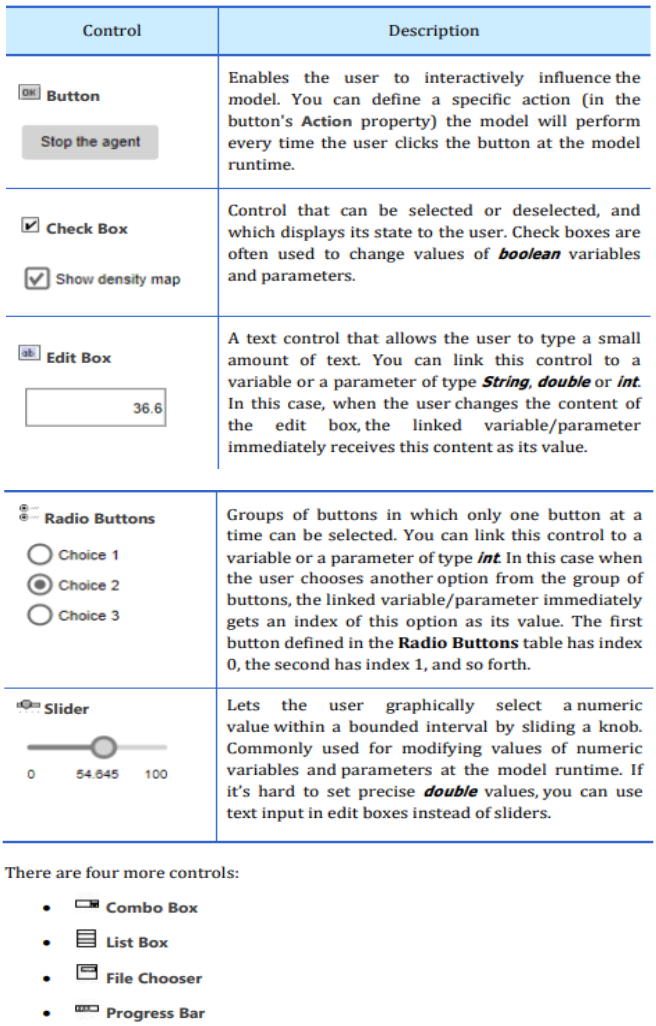



Рисунок 8.10 – Елементи керування *AnyLogic*

Додамо повзунок, який дозволить вибрати числове значення в межах обмеженого інтервалу. Повзунки зазвичай використовуються для змінювання значень числових змінних і параметрів.

12. Поверніться до головної діаграми. Відкрийте палітру елементів керування та перетягніть два повзунки на діаграму під діаграмою. Згодом ми зв’яжемо повзунки з нашими двома параметрами (рис. 8.11).

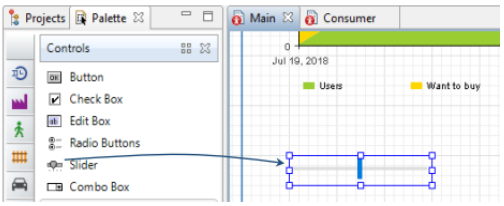



Рисунок 8.11 – Основна діаграма

13. Змініть властивості повзунка:

– установіть прапорець *Link to* та виберіть параметр *MaxWaitingTime* праворуч;

– установіть мінімальне та максимальне значення повзунка. Значення параметра може змінюватися в діапазоні, який визначається, і ми встановимо 2 як мінімальне значення та 15 як максимальне значення;

– нарешті натисніть кнопку «Додати мітки…», щоб відобразити мінімальне, максимальне та поточне значення повзунка під час виконання (під повзунком з’являться текстові форми мінімального та максимального значення) (рис. 8.12).

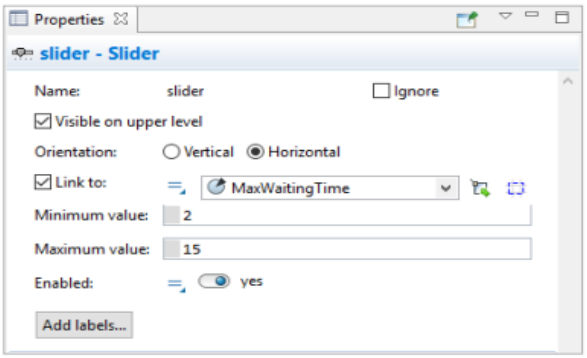



Рисунок 8.12 – Додавання мітки

14. Додайте ще один повзунок нижче та налаштуйте його так (рис.  8.13).

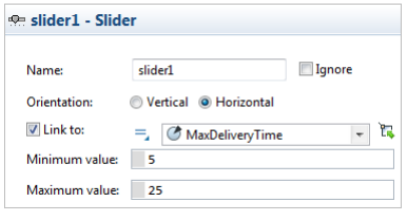



Рисунок 8.13 – Форма тексту

Деякі елементи керування мають вбудовані мітки, але ви повинні використовувати фігуру *Text*, щоб вручну створити їх для повзунків.

15. Відкрийте палітру *Presentation*, перетягніть дві фігури *Text* на діаграму та розмістіть їх над повзунками. Налаштуємо заголовки цих елементів керування (рис. 8.14).

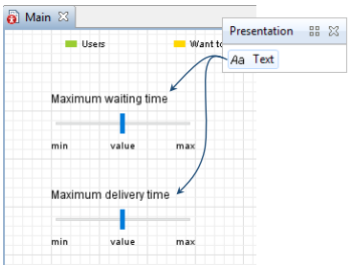



Рисунок 8.14 – Палітра презентації

16. У вікні властивостей у розділі «Текст» введіть текст, який відображатиме модель. Використовуючи текстові фігури, назвіть один повзунок «Максимальний час очікування», а інший – «Максимальний час доставки» (рис.  8.15).

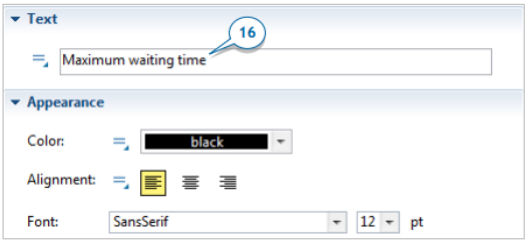



Рисунок 8.15 – Максимальний час доставки

17. У властивостях у розділі *Appearance* ви можете налаштувати колір тексту, вирівнювання, шрифт і розмір. Мітки, які відображають мінімальне, поточне та максимальне значення повзунка, також є текстовими фігурами. Їх динамічні властивості відображатимуть мінімальне, поточне та максимальне значення повзунка під час виготовлення моделі, і ви зможете редагувати їхні мітки, як редагуєте будь-яку текстову форму. Ви також можете перемістити елемент споживачів ліворуч, за рамку вікна моделі.

18. Запустіть модель і спостерігайте за поведінкою. Коли ви використаєте повзунки для змінювання максимального часу очікування або часу доставки, ви побачите, що ваші зміни відобразяться на поведінці споживачів і загальній динаміці прийняття (рис. 8.16).

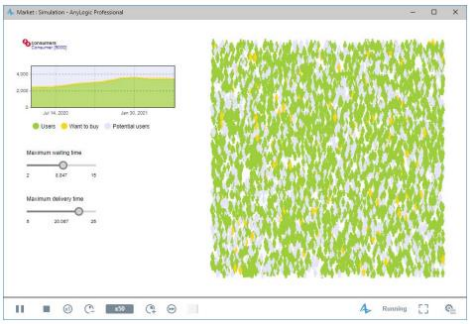


Рисунок 8.16 – Змінювання максимального часу очікування

**Питання для самоконтролю**

1. Для чого використовуються елементи керування?
2. У якому розділі можна обрати колір тексту?
3. Де можна переглянути опис розповсюдження?

## Практична робота № 9
## ПОРІВНЯННЯ ПРОГОНІВ МОДЕЛІ З РІЗНИМИ ЗНАЧЕННЯМИ ПАРАМЕТРІВ

**Мета:** набути практичних навичок порівняння прогонів моделі з різними значеннями параметрів.

### Хід виконання

На цьому етапі нам потрібно запустити модель і поспостерігати за процесом її прийняття в різних налаштуваннях. Ми могли б вручну змінити значення параметрів, запустити модель і зберегти результати, але набагато легше використовувати вбудовані експерименти *AnyLogic* для порівняння результатів. Спочатку побудуємо експеримент, який дозволить нам вручну змінювати параметр *ContactRate* і порівнювати поведінку моделі. Ми хочемо, щоб наш експеримент досліджував дані за період, що перевищує рік, тому використаємо 500 днів. Цей інтерактивний експеримент дозволяє вводити параметри моделі, запускати моделювання та додавати результати моделювання до діаграм для подальшого порівняння. Стандартний інтерфейс користувача експерименту містить поля введення та вихідні діаграми. Параметри визначають вхідні дані:

1. Відкрийте головну діаграму та додайте набір даних із палітри аналізу. Назвіть його *usersDS* (рис. 9.1).

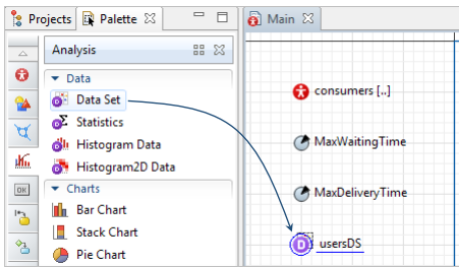



Рисунок 9.1 – Набір даних

Набір даних здатний зберігати *2D* (*X*,*Y*) дані типу *double*. Ми хочемо, щоб цей набір даних зберігав історію динаміки продажів продукції. Ми

зберігатимемо зразки даних, кожен із міткою часу та поточною кількістю користувачів продукту.

2. Щоб зберегти мітки часу, залишіть вибраним параметр набору даних «Використовувати час» як значення горизонтальної осі.

3. Встановіть значення, яке зберігатиме набір даних. У властивості «Значення вертикальної осі» введіть *consumers.NUser*.

4. Набір даних зберігає обмежену кількість останніх елементів даних, і ми обмежимо розмір вибірки до 500. Встановіть набір даних на «Зберігати» до 500 останніх зразків. Встановіть для нього значення «Автоматичне оновлення даних» із часом повторення за замовчуванням: додамо одну вибірку даних за один день життя моделі (рис. 9.2).

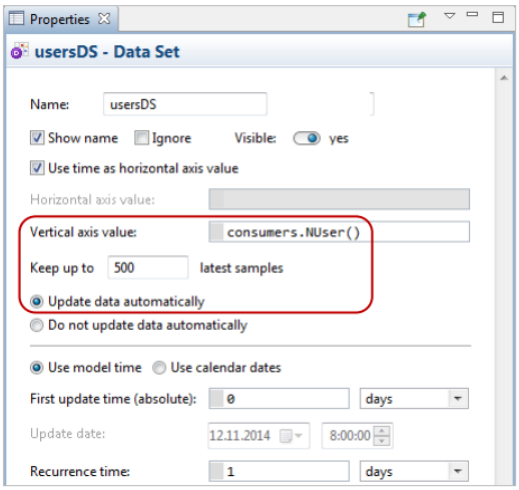



Рисунок 9.2 – Один зразок даних

Тепер у вас є набір даних, який зберігатиме історію ключової змінної (кількість користувачів продукту). Він отримує зразки даних, викликаючи статистичну функцію *NUser()*, яку ми створили для споживачів сукупності агентів.

5. Потім внесіть зміни в розділ «Редактор значень» для обох параметрів на головній діаграмі (*MaxWaitingTime і MaxDeliveryTime*). Виберіть повзунок як тип керування, встановіть значення *min* і *max* такі ж, як у повзунків на *Main*. Якщо

потрібно, змініть мітку за замовчуванням (наприклад *Maximum waiting time*) (рис. 9.3).

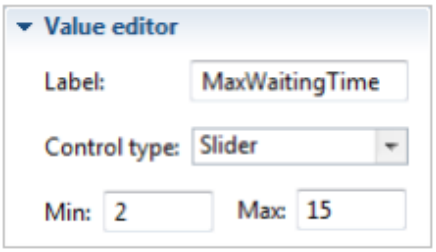



Рисунок 9.3 – Розділ редактора значень

Тепер ми готові до створення експерименту *Compare Runs*.

6. Відкрийте вікно *Projects*, клацніть правою кнопкою миші на елементі моделі та виберіть у контекстному меню *New* > *Experiment*. З'явиться майстер нового експерименту.

7. Оберіть *Compare Runs experiment* зі списку типів експериментів і натисніть *Next* (рис. 9.4).

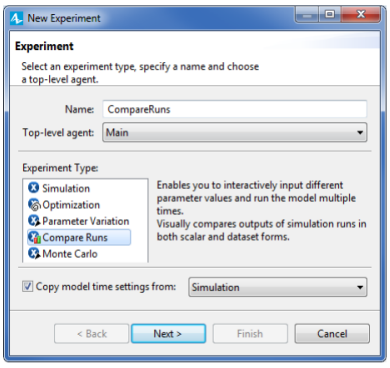



Рисунок 9.4 – Вибір *Compare Runs*

8. На сторінці *Parameters* додайте обидва параметри до стовпця «Вибір». Щоб додати параметр, виберіть його в списку доступних ліворуч і натисніть на стрілку. Ви також можете додати всі параметри.

Натисніть *Next* після того, як обидва параметри будуть у *Selection* (рис. 9.5).

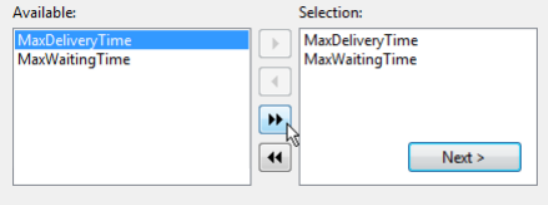



Рисунок 9.5 – Параметри

9. На наступній сторінці майстра налаштуйте вихідні діаграми для цього експерименту. На діаграмі відображатимуться дані, зібрані користувачами набору *usersDS*. У таблиці «Діаграми» виконайте всі наведені нижче дії. У стовпці «Тип» виберіть «Набір даних. b». У стовпці «Назва діаграми» введіть «Користувачі. в». У стовпці *Expression* зверніться до набору даних, який ви визначили в *Main* як r*oot.usersDS, де root* – це агент верхнього рівня моделі.

10. Натисніть кнопку «Готово» (рис. 9.6).

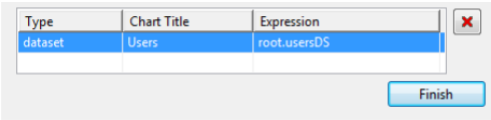



Рисунок 9.6 – Кнопка «Готово»

Діаграма експерименту *CompareRuns* має відкритися автоматично, і ви побачите інтерфейс користувача за замовчуванням, який ми створили за допомогою майстра (рис. 9.7).

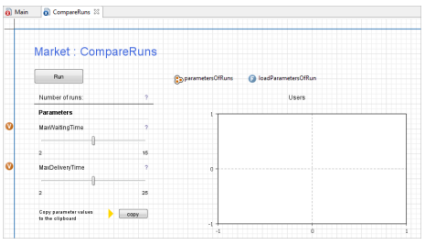



Рисунок 9.7 – Діаграма експерименту *CompareRuns*

11. Потрібно, щоб наш експеримент імітував модель лише протягом 500 днів. Для цього в дереві проєктів виберіть експеримент *CompareRuns*. У властивостях експерименту відкрийте розділ «Властивості моделі часу» та введіть 500 у полі *Stop*.

12. Проведіть дослід. Виберіть щойно створений експеримент зі списку *Run: Market / CompareRuns* або клацніть правою кнопкою миші на експерименті *CompareRuns* у дереві проєктів і виберіть *Run* з контекстного меню (рис. 9.8).

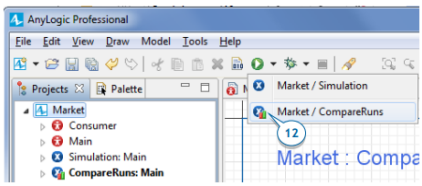



Рисунок 9.8 – Експеримент «Ринок» (*CompareRuns*)

13. У вікні моделі натисніть кнопку *Run*, щоб побачити результат, пов’язаний зі значеннями параметрів за замовчуванням. Після цього змініть значення параметрів і знову натисніть *Run*, щоб спостерігати за поведінкою системи для нових параметрів. На діаграмі відобразяться всі результати для перегляду (рис. 9.9).

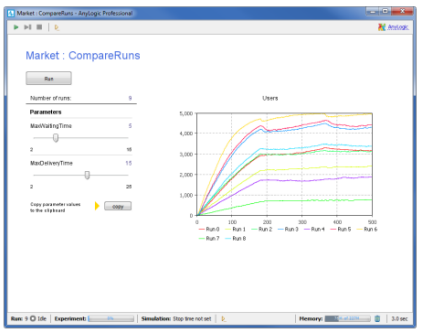



Рисунок 9.9 – Діаграма

14. Кожна крива на діаграмі відповідає певному циклу моделювання, і ви можете клацнути на будь-якому елементі у легенді діаграми, щоб виділити криві, які відповідають циклу. Елементи керування ліворуч відображатимуть значення,

які призвели до цього результату. Щоб скасувати виділення кривої, клацніть на її легенді вдруге (рис.  9.10).

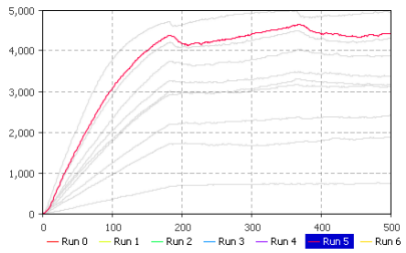



Рисунок 9.10 – Елементи керування

15. Ви можете скопіювати набори даних, клацнувши на легенді та вибравши *Copy all or Copy selected* в контекстному меню. Тепер, коли ви завершили розробку моделі ринку на основі агента, ви можете розширити її, зробивши логіку споживача складнішою (наприклад, представивши конкуруючі продукти). Ви можете знайти подібну модель *State Chart* для вибору конкуруючих продуктів у моделях із розділу «Велика книга імітаційного моделювання» прикладів моделей *AnyLogic*. Щоб переглянути моделі, виберіть *Examples from the Help menu*.

Моделювання дискретних подій є майже ровесником системної динаміки. У 1961 році інженер *IBM* Джеффрі Гордон представив *GPSS*, який вважається першою програмною реалізацією методу моделювання дискретних подій. Сьогодні деякі програми, зокрема сучасні версії *GPSS*, пропонують моделювання дискретних подій.

Моделювання дискретних подій вимагає від модельєра думати про систему, яку він чи вона хоче змоделювати, як про процес – послідовність операцій, які виконують агенти.

Операції моделі можуть включати затримки, обслуговування різними ресурсами, вибір гілок процесу, розбиття та багато іншого. Поки агенти конкурують за обмежені ресурси та можуть бути відкладені, черги будуть частиною майже всіх моделей дискретних подій. Модель визначається графічно, як блок-схема процесу, де блоки представляють операції.

Блок-схема зазвичай починається з блоків «Джерело», які генерують агентів і вводять їх у процес, і закінчується блоками «Приймач», які їх видаляють. Агенти (спочатку названі транзакціями в *GPSS* або об'єктами в іншому програмному забезпеченні моделювання) можуть представляти клієнтів, пацієнтів, телефонні дзвінки, фізичні та електронні документи, частини, продукти, піддони, комп'ютерні транзакції, транспортні засоби, завдання, проєкти, ідеї тощо. Ресурси представляють персонал, лікарів, операторів, робітників, сервери, ЦП, комп'ютерну пам'ять, обладнання та транспорт. Час обслуговування та час прибуття агента зазвичай є стохастичними, і, оскільки вони витягуються з розподілу ймовірностей, моделі дискретних подій самі є стохастичними.

Типовий результат, очікуваний від моделі дискретних подій, включає:

- використання ресурсів;
- час перебування агента в системі або її частині;
- час очікування *AnyLogic 8* за три дні 133;
- довжина черги;
- пропускна здатність системи;
- вузькі місця.

Модель магазину роботи.

Наша мета – створити модель із дискретними подіями, яка моделюватиме виробничі процеси та процеси доставки в невеликій майстерні. Сировина, що надходить на приймальний пункт, зберігається до моменту переробки на верстаті з ЧПК.

## Питання для самоконтролю

1. Як зберегти мітки часу?
2. Як встановити набір даних на зберігання до 500 останніх зразків?
3. Які дані можна отримати з кривої діаграми?

## Практична робота № 10
## СТВОРЕННЯ ПРОСТОЇ МОДЕЛІ

**Мета:** набути практичних навичок зі створення простої моделі.

### Хід виконання

Почнемо зі створення простої моделі, яка імітуватиме прибуття піддонів до робочого цеху, їхнє зберігання в транспортному доку та прибуття до зони навантажувача.

1. Створіть нову модель. У майстрі нової моделі встановіть назву моделі: Job Shop і одиниці вимірювання часу моделі: хвилини.

2. Відкрийте палітру *Presentation* . Палітра містить кілька форм, які можна використовувати для малювання анімації моделі, включаючи прямокутник, лінію, овал, ламану лінію та криву.

3. На палітрі *Presentation* виберіть фігуру *Image*, а потім перетягніть її на головну діаграму. Ви можете використовувати фігуру «Зображення», щоб додати зображення у кількох графічних форматах, зокрема *PNG, JPEG, GIF* і *BMP*, до вашої презентації (рис. 10.1).

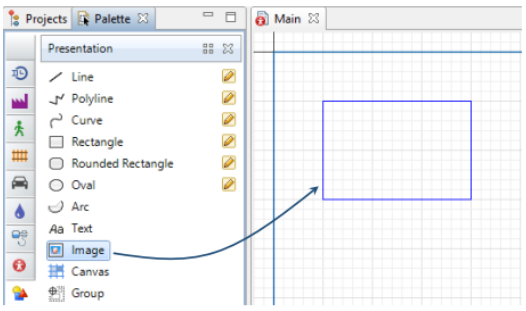



Рисунок 10.1 – Палітра презентації

4. Ви побачите діалогове вікно, у якому буде запропоновано вибрати файл зображення, який буде відображатися.

5. Перейдіть до наступного розташування та виберіть зображення *layout.png*: папка *AnyLogic/resources/AnyLogic* за 3 дні/Job Shop. Після вибору

зображення *layout.png* діаграма типу головного агента має виглядати так, як показано на рисунку 10.2.

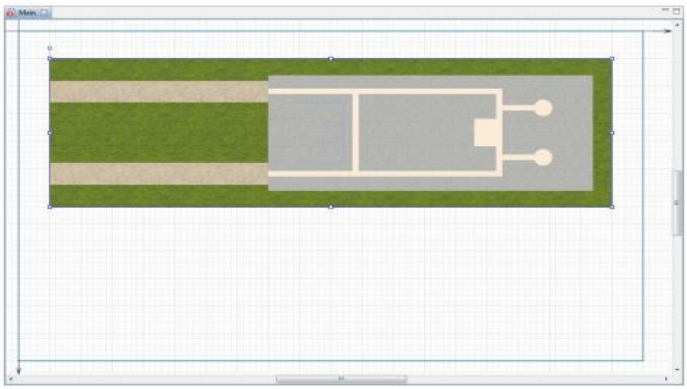


Рисунок 10.2 – Тип основного агента

*AnyLogic* додає зображення в оригінальному розмірі на головну діаграму, але ви також можете змінити ширину або довжину зображення. Якщо ви спотворите пропорції зображення, як показано на рисунку 10.3, ви повернетеся до початкового розміру зображення, відкривши вікно *Properties view* та натиснувши на *Reset* to original size.

Рисунок 10.3 – Основна діаграма

6. Виберіть зображення в графічному редакторі. У вікні *Properties view* встановіть прапорець *Lock* , щоб заблокувати зображення (рис. 10.4).

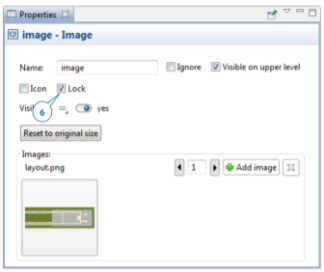



Рисунок 10.4 – Перегляд властивостей

Замкові форми:

– ви можете заблокувати фігуру, щоб вона не реагувала на клацання мишею, і не можете вибрати її в графічному редакторі. Це значно полегшує малювання фігури поверх макетів, які становлять такі об’єкти, як заводи чи лікарні;

– якщо вам потрібно розблокувати фігуру, клацніть правою кнопкою миші на графічному редакторі та виберіть у меню *Unlock All*.

Елементи розмітки простору «Наш наступний крок» – використання палітри розмітки простору для розміщення форм розмітки простору поверх макета робочої майстерні. Палітра містить елемент *Path*, три елементи *Node*, елемент *Attractor* і фігури *Pallet Rack*.

Створення мережі «Шляхи та вузли». Це елементи розмітки простору, які визначають розташування агентів:

– вузол – це місце, де агенти можуть перебувати або виконувати операції;

– шлях – це маршрут, який агенти можуть використовувати для переміщення між вузлами.

Вузли та шляхи утворюють мережу, яку агенти моделі можуть використовувати для пересування по найкоротшим шляхам між вузлами відправлення та вузлами призначення. Зазвичай ви створюєте мережу, коли процеси вашої моделі відбуваються у визначеному фізичному просторі та в ній є рухомі агенти та ресурси. Передбачається, що сегменти мережі мають необмежену пропускну здатність і агенти не заважають один одному. Тепер, коли ви розглянули мережі та їх складники, ми готові створити мережу, яка визначатиме шляхи руху для піддонів моделі. Першим кроком є використання прямокутних вузлів для визначення конкретних областей на макеті робочої майстерні. Намалюйте прямокутний вузол над входом у робочу майстерню, як це показано на рисунку 10.5, щоб представити док для прийому піддонів нашої моделі.

7. Відкрийте палітру *Space Markup* і перетягніть елемент *Rectangular Node* на головну діаграму. Змініть розмір вузла. Вузол повинен виглядати так (рис. 10.5).

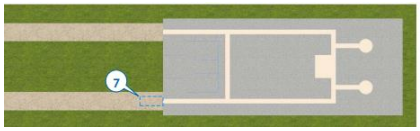



Рисунок 10.5 – Палітра *Space Markup*

8. Назвіть створений вузол *receiveDock*.

9. Намалюйте вузол, щоб визначити місце, де агенти моделі паркуватимуть вилкові навантажувачі, коли вантажівки простоюють або більше не потрібні агентам для виконання завдання. Використовуйте інший прямокутний вузол, щоб намалювати зону паркування (рис. 10.6), а потім назвіть цей вузол *forkliftParking*.

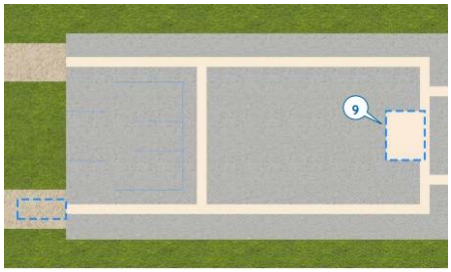



Рисунок 10.6 – Прямокутний вузол

Намалюємо траєкторію руху, щоб керувати вилковими навантажувачами нашої моделі.

10. Щоб намалювати траєкторію руху, яка спрямовуватиме вилкові навантажувачі нашої моделі, зробіть таке:

– на палітрі *Space Markup* двічі клацніть по елементу *Path*, щоб активувати режим малювання;

– намалюйте шлях, як показано на рисунку 10.7, клацнувши на межі *receiveDock* на схемі, щоб додати точку повороту шляху, а потім клацніть на межі

вузла *forkliftParking*. Якщо ви успішно з'єднали вузли, точки з'єднання шляху відображатимуться блакитним підсвічуванням кожного разу, коли ви вибиратимете шлях (рис. 10.7).

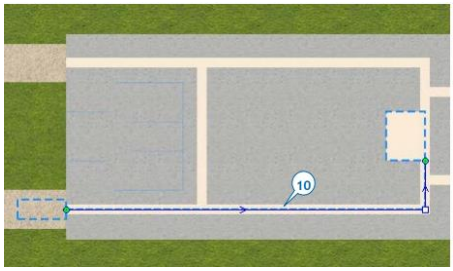



Рисунок 10.7 – Межа *ReceivingDock*

За замовчуванням шляхи в *AnyLogic* є двонаправленими, однак ви можете обмежити рух уздовж вибраного шляху в одному напрямі, очистивши властивість «Двонаправлений», а потім визначивши напрям руху. Ви можете переглянути напрям заданого шляху, вибравши шлях, а потім переглянувши стрілку напряму, яка відображається в графічному редакторі.

11. Визначте складське сховище вашої моделі, перетягнувши елемент *Pallet Rack* з палітри *Space Markup* на макет і розмістивши його проходження на шляху. Правильно розміщений палетний стелаж підсвічується зеленим кольором, який показує, що він підімкнений до мережі (рис. 10.8).

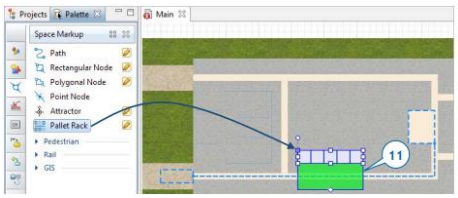



Рисунок 10.8 – Палітра *Space Markup*

*Pallet* rack *The Pallet Rack*простору Pallet Rack графічно становить стелажі для піддонів, які ви часто бачите на складах і в зонах зберігання.

Як ви можете бачити нижче, елемент має три альтернативні конфігурації (рис. 10.9).

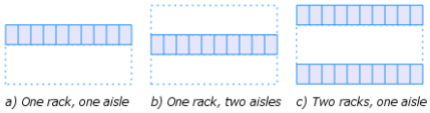



Рисунок 10.9 – Стійка для піддонів

Під час виконання елемент *Pallet Rack* керує агентами, які модель зберігає в однорівневих або багаторівневих комірках, доступних по сторонах проходу.

12. В області властивостей стелажа для піддонів виконайте такі дії:

– встановіть тип: дві стійки, один прохід;

– кількість комірок: 10;

– висота рівня: 10.

У розділі «Розташування та розмір»:

– довжина: 160;

– глибина лівої палетної стійки: 14;

– глибина правого палетного стелажа: 14;

– ширина проходу: 11.

13. Після внесення цих змін стелаж для піддонів має нагадувати стелаж для піддонів, показаний на рисунку 10.10. Якщо це необхідно, перемістіть стелаж для піддонів так, щоб його центральний прохід лежав на шляху. Переконайтеся, що палетний стелаж підімкнений до мережі, двічі клацнувши по ньому, щоб вибрати його. Ваш перший клік вибере всю мережу, а другий вибере палетний стелаж. Палетний стелаж має відображатися зеленим кольором, що вказує на підімкнення до мережі (рис. 10.10).

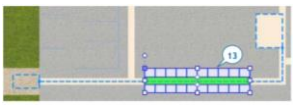


Рисунок 10.10 – Стелаж для палет

Ми розмітили простір нашої моделі, намалювавши важливі місця та шляхи поверх макета, і тепер ми будемо використовувати бібліотеку моделювання процесів *AnyLogic* для моделювання процесів.

Бібліотека моделювання процесів.

Блоки в бібліотеці моделювання процесів *AnyLogic* дозволяють використовувати комбінації агентів, ресурсів і процесів для створення систем реального світу, зорієнтованих на процеси моделей. Раніше ви дізналися про агентів і ресурси, і ми будемо використовувати ці відомості, визначаючи процеси, як послідовності операцій, що включають черги, затримки та використання ресурсів.

Процеси вашої моделі визначаються блок-схемами, графічними представленнями процесів, які ви створюєте з блоків бібліотеки моделювання процесів. На наступних кроках ви створите блок-схему процесу (рис. 10.11).

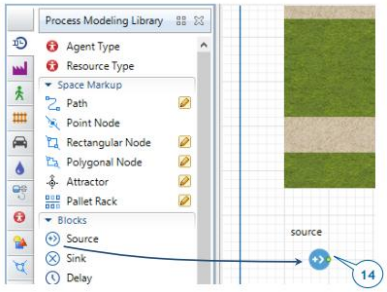



Рисунок 10.11 – Блоки бібліотеки моделювання процесів

14. Перетягніть елемент *Source* з палітри *Process Modeling Library* на графічну діаграму та назвіть блок *sourcePallets*. Тоді як блок *Source* зазвичай діє, як початкова точка процесу, наша модель використовуватиме його для генерації палет.

15. В області властивостей блока *sourcePallets* виконайте таке, щоб переконатися, що піддони моделі надходять кожні п’ять хвилин і з’являються у вузлі *receiveDock* (рис. 10.12):

– в області «Прибуття визначено» клацніть на *Interarrival time;*

– у полі між прибуттями введіть 5 і виберіть хвилини зі списку праворуч, щоб позначити, що піддони надходять кожні п’ять хвилин;

– в області розташування прибуття клацніть *Network / GIS* у списку;

– в області *Node* клацніть на списку *receiveDock.*

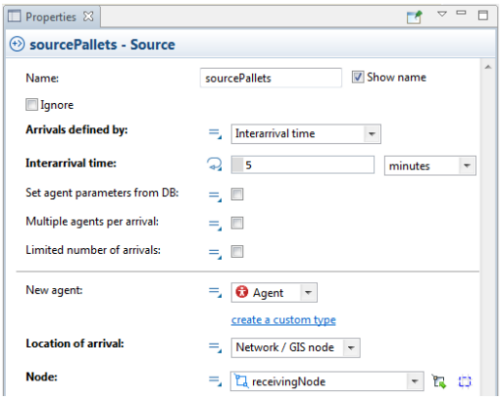



Рисунок 10.12 – Мережа

Посилання на елементи моделі з параметрів блоку.

Параметри блоку пропонують два способи вибору графічного елемента:

– можна вибрати графічний елемент зі списку доступних і дійсних елементів, який відображається поруч із параметром (рис. 10.13);

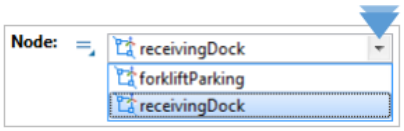



Рисунок 10.13 – Графічний елемент

– можна вибрати графічний елемент, натиснувши кнопку вибору, яка відображається поруч зі списком. Якщо ви натиснете кнопку вибору, ваш вибір буде обмежений доступними та дійсними елементами, які ви можете вибрати, клацнувши на графічному редакторі (рис.  10.14).

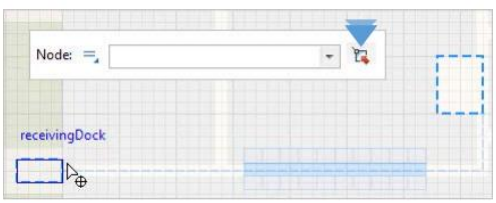



Рисунок 10.14 – Графічний редактор

Продовжуйте побудову блок-схеми, додаючи інші блоки бібліотеки моделювання процесів.

16. Перетягніть блок *RackStore* з палітри *Process Modeling Library* на діаграму та розмістіть його поряд із блоком *sourcePallets*, щоб вони автоматично з'єднувалися, як показано на діаграмі рисунка 10.15. Блок *RackStore* розміщує піддони в комірки заданого палетного стелажа.

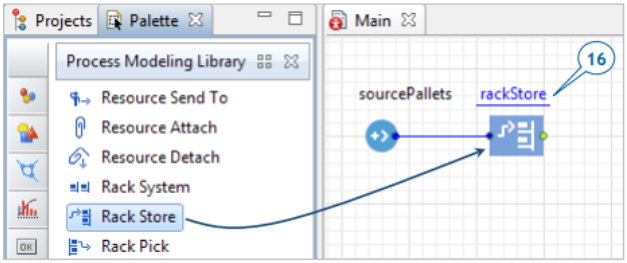



Рисунок 10.15 – Блок *RackStore*, що розміщує піддони

17. В області властивостей блока *rackStore* виконайте такі дії:

– у полі «Ім'я» введіть *storerawmaterial*;

– у списку *pallet rack / Rack System* натисніть *palletRack*. в; у списку розташування (черги) агента клацніть *receiveDock*, щоб вказати місце, де агенти очікують на збереження (рис. 10.16).

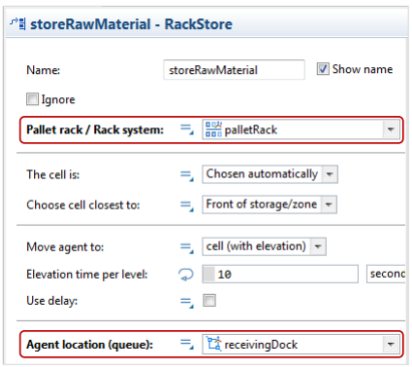



Рисунок 10.16 – Список стелажної системи

18. Додайте блок *Delay*, щоб імітувати, як піддони очікують у стелажі, а потім назвіть блок *rawMaterialInStorage* (рис. 10.17).

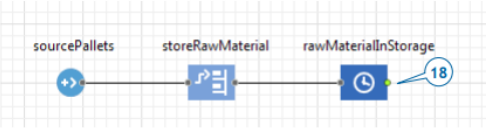



Рисунок 10.17 – Блок *rawMaterialInStorage*

Ви, мабуть, помітили, що *AnyLogic* автоматично підмикає правий порт блоку до лівого порту наступного блока. Кожен блок бібліотеки моделювання процесів має лівий вхідний порт і правий вихідний порт, але ви повинні з'єднувати вхідні порти лише з вихідними портами.

19. В області властивостей блока *rawMaterialInStorage* виконайте такі дії:

– у полі «Час затримки» введіть *triangular* (15, 20, 30) і виберіть хвилини зі списку;

– установіть прапорець *Maximum capacity*, щоб гарантувати, що агенти не застрягнуть, якщо вони чекатимуть, коли їх заберуть зі сховища (рис. 10.18).

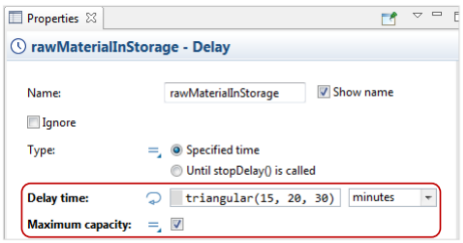



Рисунок 10.18 – Прапорець «Максимальна ємність»

20. Додайте блок *RackPick*, підімкніть його до блок-схеми, а потім назвіть його *pickRawMaterial*.

У нашій моделі блок *RackPick* видаляє піддон із комірки в стелажі для піддонів, а потім переміщує його до вказаного місця призначення (рис. 10.19).

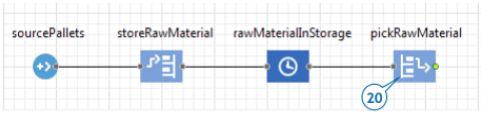



Рисунок 10.19 – Видалення блока *RackPick*

21. В області властивостей блока *pickRawMaterial* виконайте такі дії:

– у списку *Pallet Rack / Rack System* клацніть на *palletRack*, щоб вибрати палетний стелаж, який надаватиме піддони агентам;

– у списку *Node* клацніть на *forkliftParking*, щоб вказати, де агенти мають паркувати навантажувачі (рис. 10.20).

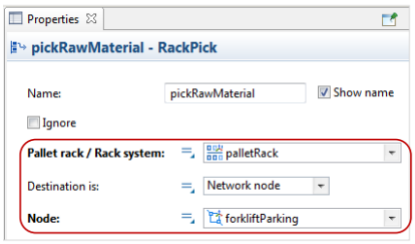



Рисунок 10.20 – Стоянка навантажувача

22. Додайте блок раковини. Блок приймача розпоряджається агентами і зазвичай є кінцевою точкою блок-схеми (рис. 10.21).


Рисунок 10.21 – Блок раковини, що утилізує агенти

23. Ми закінчили створення цієї простої моделі, і тепер ви можете запустити її та спостерігати за її поведінкою. Запустіть модель (*Job Shop / Simulation Experiment)* (рис. 10.22).

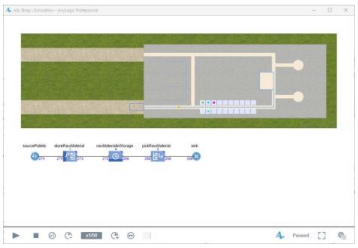


Рисунок 10.22 – Імітаційний експеримент

Відображення повідомлення про помилку. Для виконання дискретної події необхідно під’єднати палетну стійку до мережі. Потрібно вибрати форму стелажа для піддонів у графічному редакторі, перемістити його, допоки на проході стелажа для піддонів не відобразиться зелене виділення, яке вказує на підімкнення до мережі, а потім повторно запустити модель (рис. 10.23).

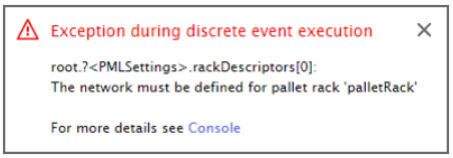



Рисунок 10.23 – Відображення повідомлення про помилку
(виняток під час виконання дискретної події)

### Питання для самоконтролю

1. Яку команду потрібно застосувати, щоб агенти не очікували, коли їх заберуть зі сховища?
2. Що потрібно зробити, щоб зімітувати піддони?
3. Що потрібно зробити в разі помилки в роботі простої моделі?

## Практична робота № 11
## ДОДАВАННЯ РЕСУРСІВ

**Мета:** набути практичних навичок щодо додавання ресурсів.

### Хід виконання

Продовжимо вдосконалення нашої моделі, додавши ресурси – вилкові навантажувачі для зберігання піддонів у стелажі, а потім перемістимо їх у виробничу зону. Ресурси – це об’єкти, які агенти використовують для виконання

певної дії. Агент повинен отримати ресурс, виконати дію, а потім звільнити ресурс. Деякі приклади ресурсів:

– лікарі, медсестри, обладнання та інвалідні візки моделі для лікарні;

– транспортні засоби та контейнери моделі ланцюга постачання;

– вилкові навантажувачі та робітники складської моделі трьох типів ресурсів – статичні, рухомі та переносні;

– статичні ресурси, прив'язані до певного місця, які не можна перемістити;

– рухомі ресурси, які можуть переміщуватися незалежно;

– портативні ресурси, які можуть бути переміщені агентами або шляхом переміщення ресурсів.

В *AnyLogic* блок *resourcepool* бібліотеки моделювання процесів визначає кожен набір або пул ресурсів. Одиниці ресурсу можуть мати індивідуальні атрибути, і кожен ресурс має графічну діаграму, до якої можна додавати такі елементи, як діаграми станів, параметри та функції. ресурси нашої моделі – це вилкові навантажувачі, які переміщують піддони із зони розвантаження на палетний стелаж, а потім доставляють піддони зі стелажа у виробничу зону:

1. На палітрі *Process Modeling Library* перетягніть блок *ResourcePool* на головну діаграму. Вам не потрібно під'єднувати блок до блок-схеми (рис. 11.1).

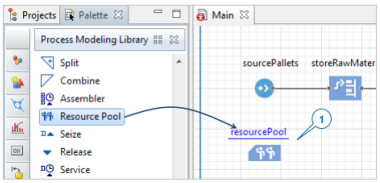



Рисунок 11.1 – Палітра бібліотеки моделювання процесу

2. Назвіть блокові навантажувачі (рис. 11.2).

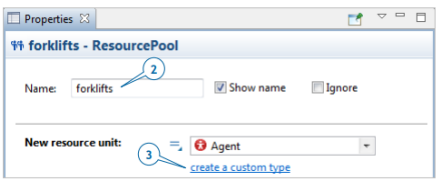



Рисунок 11.2 – Блок навантажувачів

3. В області властивостей блоку навантажувачів натисніть кнопку для створення власного типу. Таким чином ми створюємо новий тип ресурсу.

4. У майстрі нового агента:

– у полі імені агента введіть *forklifttruck;*

– натисніть *next* (рис. 11.3).

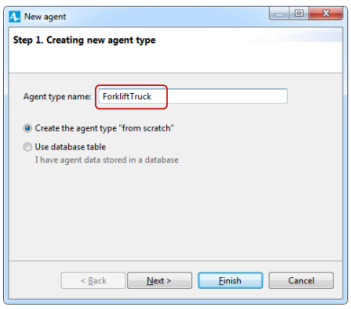



Рисунок 11.3 – Автонавантажувач

– на наступній сторінці у списку в лівій частині майстра розгорніть область «Склади та контейнерні термінали», а потім клацніть на тривимірній анімаційній фігурі вилковий навантажувач;

– натисніть кнопку *finish* (рис. 11.4).

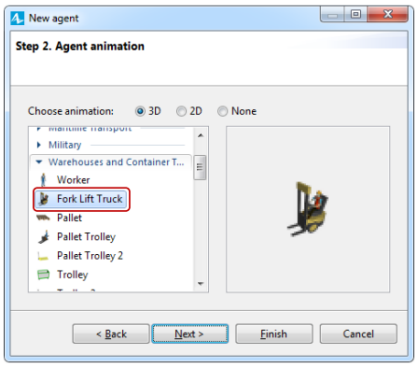



Рисунок 11.4 – Вилковий навантажувач

Відкриється діаграма типу агента *ForkliftTruck* і відобразить форму анімації, яку ви вибрали в майстрі.

5. Клацніть на вкладці *Main*, щоб відкрити головну діаграму (рис. 11.5).

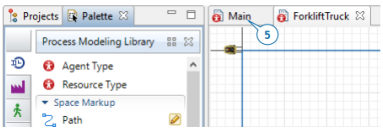



Рисунок 11.5 – Основна діаграма

Ви побачите, що тип ресурсу *ForkliftTruck* вибрано в параметрі *New resource unit* блока *ResourcePool*.

6. Змініть інші параметри типу ресурсу навантажувачів:

– у полі *capacity box* 5, щоб установити кількість вилкових навантажувачів у нашій моделі;

– у полі *speed box* 1, щоб виберати метри за секунду зі списку праворуч;

– в області «Домашнє розташування» (вузли) виберіть вузол «*plus button and then click forkliftparking*» і натисніть кнопку з плюсом, а потім *forkliftparking* у списку вузлів моделі (рис. 11.6; 11.7).

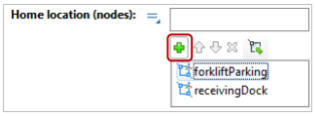



Рисунок 11.6 – Паркування навантажувача

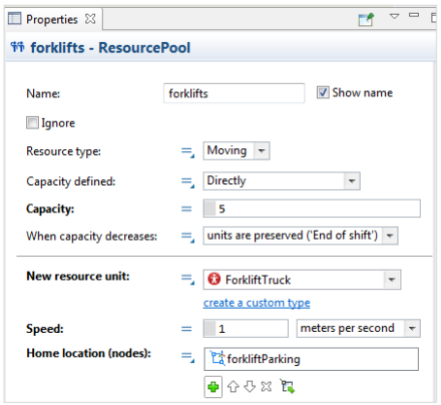



Рисунок 11.7 – Вузли моделі

Ми визначили наші ресурси, але нам все ще потрібно переконатися, що блоки блок-схеми нашої моделі використовуватимуть їх під час змодельованих процесів.

7. В області властивостей блоку *storeRawMaterial* виконайте такі дії:

– натисніть на стрілку, щоб розгорнути область ресурсів;

– установіть прапорець «Використовувати ресурси для переміщення»;

– у списку наборів ресурсів (альтернатив) виберіть вилкові навантажувачі, щоб переконатися, що блок блок-схеми використовує вибрані ресурси (у нашому випадку вилкові навантажувачі) для переміщення агентів. Натисніть кнопку з плюсом, а потім виберіть вилкові навантажувачі у списку ресурсів моделі;

– виберіть параметр переміщення зі швидкістю ресурсу;

– у списку ресурсів для переміщення виберіть вилкові навантажувачі. це змусить агентів (піддони) рухатися зі швидкістю навантажувача під час транспортування до зони зберігання;

– в області *return* home area виберіть (якщо немає інших завдань) навантажувачі, які мають повернутися на вихідне місце після виконання завдань (рис. 11.8).

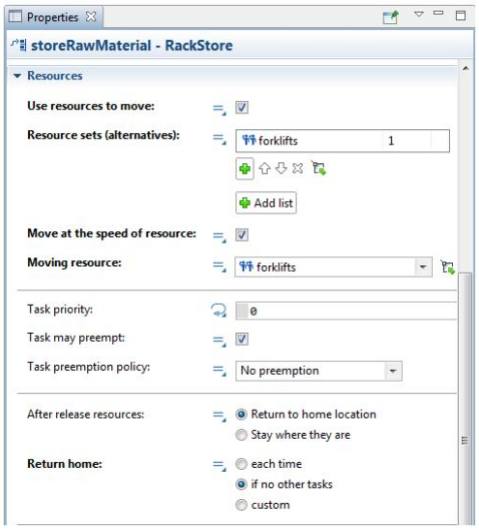



Рисунок 11.8 – Область повернення додому

8. В області властивостей блока *pickRawMaterial* виконайте такі дії:

– натисніть стрілку, щоб розгорнути область ресурсів;

– установіть прапорець *use resources to move check box;*

– у списку наборів ресурсів (альтернативи) клацніть вилкові навантажувачі, щоб переконатися, що блок блок-схеми використовує вилкові навантажувачі для переміщення агентів;

– виберіть параметр *move at the speed of resource option;*

– у списку ресурсів для переміщення виберіть вилкові навантажувачі;

– клацніть у зоні *return home area*, якщо немає інших завдань, щоб гарантувати, що навантажувачі повернуться на вихідне місце після виконання завдань. Якщо ресурси нашої моделі переміщують агента, блок *rackstore* (або *rackpick*) захоплює їх, переносить до розташування агента, приєднується до нього й переміщує агента в комірку, а потім звільняє ресурси.

9. Запустіть модель (рис. 11.9).

Рисунок 11.9 – Запуск моделі

Ви побачите, як вилкові навантажувачі забирають піддони та зберігають їх у стелажі для піддонів. Після короткої затримки вони переміщують піддони на стоянку навантажувача, де піддони зникають (рис. 11.10).

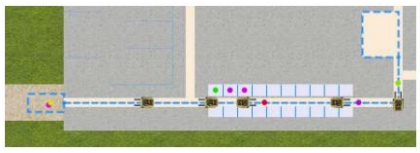


Рисунок 11.10 – Коротка затримка

**Питання для самоконтролю**

1. В яких сферах можна застосовувати розглянуту модель?
2. Яким є алгоритм дій для створення нового ресурсу?
3. Як змінити параметри типу ресурсу навантажувача?

## Практична робота № 12
## СТВОРЕННЯ 3D-АНІМАЦІЇ

**Мета:** набути практичних навичок щодо створення анімації у тривимірному форматі.

### Хід виконання

Ми розглянули низку функцій, завдяки яким *AnyLogic* став таким потужним інструментом моделювання. Але є ще й інші, і одна з найцікавіших – це 3D-анімація. Ознайомлення з «Об'єктами камери» *AnyLogic* дозволяють визначити вигляд, який відображається у 3D-вікні. По суті, об'єкт камери «знімає» картинку, яку ви бачите. Ви також можете створити кілька об'єктів камери, щоб показати різні області однієї 3D-сцени або один об'єкт із різних точок зору. Якщо ви використовуєте більше одного об'єкта камери, ви можете легко переходити від одного перегляду до іншого під час виконання:

1. На палітрі *Presentation* перетягніть об'єкт *Camera* на головну діаграму так, щоб він спрямовувався на макет майстерні.

2. Перетягніть елемент *3D Window* на головну діаграму, а потім розмістіть його під блок-схемою процесу (рис. 12.1).

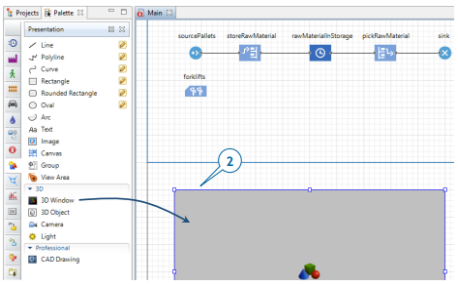



Рисунок 12.1 – Основна діаграма

3D-вікно.

Окрім можливості додати кілька камер до вашої моделі, ви також можете додати кілька 3D-вікон, кожне з яких відображатиме одну й ту саму 3D-сцену з іншої точки зору.

3. Дозвольте камері *shoot* зробити картинку для 3D-вікна. В області властивостей 3D-вікна клацніть на камері у списку камер.

4. Запобігти зйомці камерою зображення з-під підлоги, вибравши опцію «Обмежено Z вище 0» зі списку «Тип навігації» (рис. 12.2).

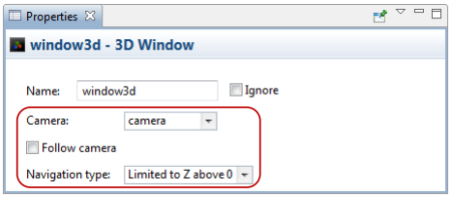



Рисунок 12.2 – Список типів навігації

5. Запустіть модель. Коли ви створюєте 3D-вікно, *AnyLogic* додає область перегляду, яка дозволяє легко переходити до 3D-вигляду під час виконання. Щоб перейти на цей 3D-вид, клацніть на крайньому правому елементі керування *Toggle Developer panel*, а потім виберіть [*window3d*] в області вибору перегляду для навігації списком. Зона перегляду розширює тривимірне вікно до повного розміру вікна моделі (рис. 12.3).

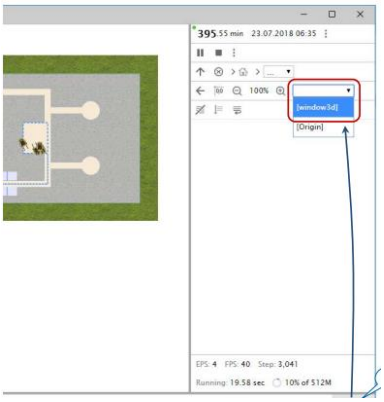


Рисунок 12.3 – Перехід до панелі розробника

6. Виконайте одну або декілька з наведених нижче дій для навігації в 3D під час виконання:

– щоб перемістити камеру вліво, вправо, вперед або назад, перетягніть мишу у вибраному напрямі;

– щоб перемістити камеру ближче або далі від центру сцени, обертайте коліщатко миші;

– щоб повернути сцену відносно камери, перетягніть мишу, утримуючи *Alt* і ліву кнопку миші.

7. Виберіть подання, яке потрібно відображати під час виконання, клацніть правою кнопкою миші (*Mac OS: CTRL* + клацання) всередині 3D-сцени, а потім клацніть на «Копіювати розташування камери» (рис. 12.4).

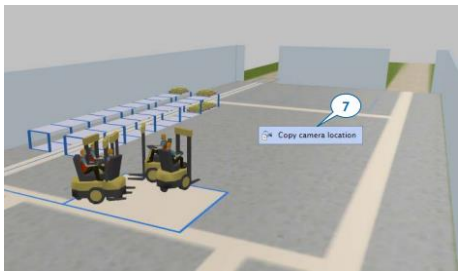



Рисунок 12.4 – Копіювання розташування камери

8. Закрийте вікно моделі.

9. В області властивостей камери застосуйте розташування камери, вибране на попередньому кроці, натиснувши «Вставити координати з буфера обміну» (рис. 12.5).

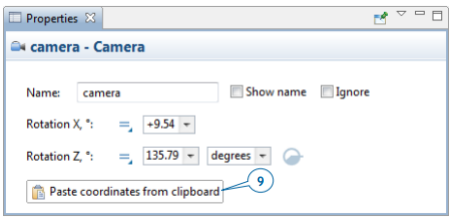



Рисунок 12.5 – Область властивостей

ПРИМІТКА. Якщо ви не можете знайти камеру, ви можете скористатися деревом проєктів. Камера відображатиметься під гілкою «Презентація головного агента».

10. Запустіть модель, щоб переглянути 3D-вид з нової позиції камери, а потім закрийте вікно моделі.

11. Розгорніть пішохідну зону палітри *Space Markup* palette's *Pedestrian* area, а потім двічі клацніть на піктограмі елемента *Wall*, щоб увімкнути режим малювання стіни (рис. 12.6).

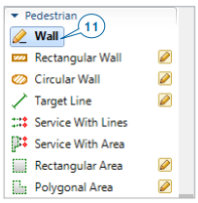



Рисунок 12.6 – Пішохідна зона палітри *Space Markup*

12. Виконайте такі дії, щоб намалювати стіни навколо робочої зони макета майстерні:

– клацніть на позицію в графічному редакторі, де ви хочете почати малювати стіну;

– перемістіть вказівник у будь-якому напрямі, щоб намалювати пряму лінію, а потім клацніть на будь-якій точці для зміни напряму;

– двічі клацніть на точці, де потрібно припинити малювання стіни (рис. 12.7).

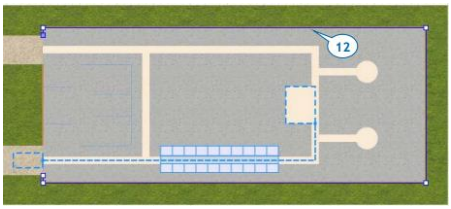



Рисунок 12.7 – Графічний редактор

13. Щоб змінити колір і текстуру заливки стіни, зробіть таке:

– в області *properties* стіни розгорніть розділ *appearance;*

– у меню кольорів виберіть *other colors;*

– у діалоговому вікні *colors* виберіть колір, який потрібно застосувати до стіни з палітри або спектру.

Ви також можете встановити рівень прозорості (використовуйте повзунок *Transparency slider* у діалоговому вікні *Colors*) або налаштувати стіну за допомогою будь-якої наданої текстури (клацніть на пункті *Textures…* у меню кольорів). У цьому розділі ми використовуємо стіни, щоб прикрасити нашу модель. Стіни також можуть бути перешкодами.

14. Перейдіть до розділу *Position and size section* стіни та змініть *Z-Height* на 40. *AnyLogic* автоматично встановлює висоту фігури у 20 пікселів, щоб забезпечити її об'єм у 3D-перегляді, але тепер ми збільшили її висоту до 40 пікселів.

15. Намалюйте ще одну стіну між виходами, а потім змініть параметри в області властивостей другої стіни, щоб вони відповідали першій стіні (рис. 12.8).

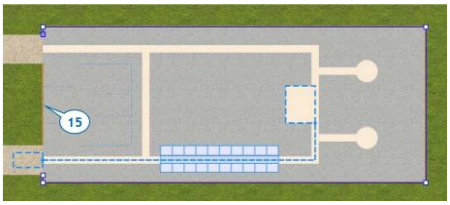



Рисунок 12.8 – Область властивостей

16. Запустіть модель і перегляньте 3D-анімацію. Ви побачите, що анімація нашої моделі використовує форми циліндра для представлення піддонів, але ми виправимо проблему, створивши тип агента, який визначає спеціальну анімацію для піддонів.

17. В області властивостей блока *sourcePallets* у списку *New agent* клацніть на посиланні «Створити настроюваний тип» (рис. 12.9).

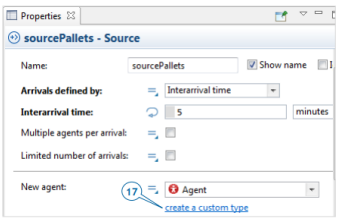



Рисунок 12.9 – Новий список агентів

18. У Майстрі нового агента виконайте такі дії (рис. 12.10):

– у полі «Ім’я типу агента» введіть *pallet*;

– натисніть «Далі»;

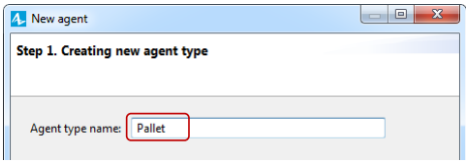



Рисунок 12.10 – Поле імені типу агента

– на наступній сторінці майстра розгорніть розділ «Склади та контейнерні термінали» у списку ліворуч, а потім клацніть на 3d-анімаційній фігурі палета;

– натисніть кнопку «Готово» (рис. 12.11).

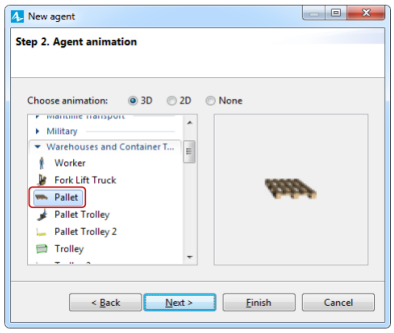



Рисунок 12.11 – Піддон

*AnyLogic* створює тип агента *Pallet* і відкриває діаграму *Pallet*, яка відображає анімацію, вибрану в майстрі. Нашим наступним кроком буде додавання анімації продукту поверх анімації піддона, але спочатку ми збільшимо зображення, щоб побачити піддон ближче.

19. Використовуючи панель інструментів *Zoom*, збільшіть діаграму *Pallet* до 500 %, а потім перемістіть полотно праворуч і вниз, щоб переглянути точку відліку осі та форму анімації палети.

Збільшення або зменшення перегляду панелі інструментів *Zoom AnyLogic* дозволяє збільшувати або зменшувати вигляд графічної діаграми (рис. 12.12).

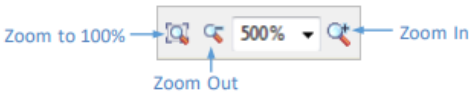



Рисунок 12.12 – *Zoom AnyLogic*

20. Щоб почати додавати анімацію продукту поверх анімації піддона, виконайте таке:

– на палітрі 3d-об’єктів розгорніть область «Бокси»;

– перетягніть об’єкт *Box 1 Closed* у верхній лівий куток палети (рис. 12.13).

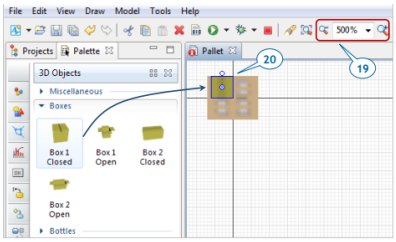



Рисунок 12.13 – Закритий об’єкт

21. Оскільки ця коробка здається занадто великою порівняно з піддоном, змінимо масштаб коробки на 75 % (рис. 12.14).

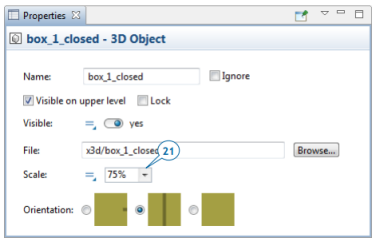



Рисунок 12.14 – Масштаб коробки до 75 %

22. В області *Properties* коробки розгорніть розділ *Position*, а потім змініть координату *Z* коробки на 2. Наша зміна відображає той факт, що ми хочемо розмістити коробки на піддонах, а висота кожного піддона становить близько двох пікселів.

23. Додайте три коробки, скопіювавши першу коробку тричі. Щоб скопіювати поле, виділіть його, а потім натисніть і утримуйте *CTRL*, перетягнувши поле. У нашому піддоні тепер є чотири закриті ящики, і ви можете змінити рівень масштабування до 100 %, натиснувши на кнопку «Збільшити до 100 %» на панелі інструментів (рис. 12.15).


Рисунок 12.15 – Кнопка «Збільшити до 100 %» панелі інструментів

Якщо ви відкриєте область властивостей блока *sourcePallets*, то побачите, що піддон вибрано як нового агента. Цей блок буде генерувати агентів типу *Pallet*.

25. Запустіть модель. Ви побачите, що форми піддонів замінили різнокольорові циліндри. Однак, якщо ви збільшите 3D-сцену, то помітите, що навантажувачі не транспортують піддони. Ми виправимо цю помилку, перемістивши анімацію піддонів нашої моделі так, щоб навантажувачі могли забирати піддони (рис. 12.16).

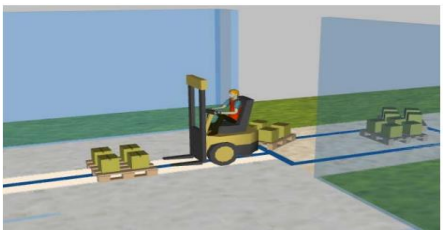


Рисунок 12.16 – Транспортування піддонів

26. У вікні «Проєкти» двічі клацніть по типу агента *ForkliftTruck*, щоб відкрити його діаграму, а потім перемістіть фігуру *forkliftWithWorker* на одну клітинку праворуч. Тепер фігура анімації розміщується в потрібному місці,

а піддони нашої моделі вирівняні відповідно до розміщення навантажувачів (рис. 12.17).

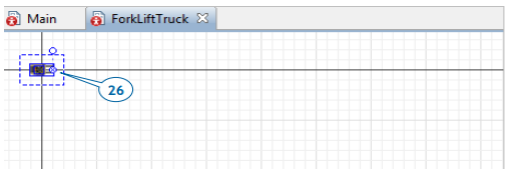



Рисунок 12.17 – Навантажувач з працівником

27. Відкрийте головну схему та в області властивостей стелажа для піддонів у полі «Кількість рівнів» введіть 2. ПОРАДА: пам’ятайте, що перше клацання вибирає мережу, а друге – мережевий елемент.

28. В області властивостей блока блок-схеми *storeRawMaterial* встановіть для параметра «Час підйому на рівень» значення 30 секунд.

29. В області властивостей блока *pickRawMaterial* встановіть параметр «Час падіння на рівень» 30 секунд.

30. Запустіть модель, і ви побачите палетну стійку з двома рівнями (рис. 12.18).

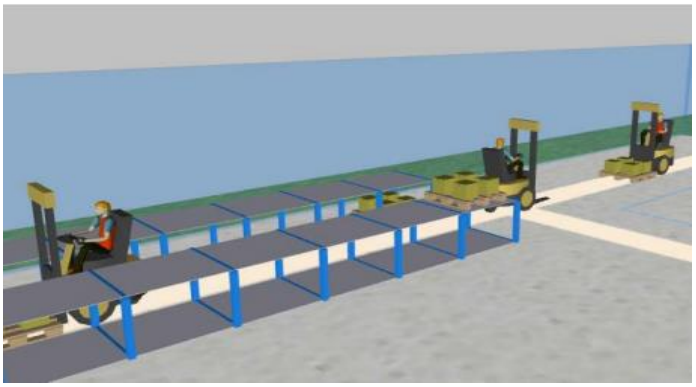


Рисунок 12.18 – Область властивостей

**Питання для самоконтролю**

1. Як можна додати декілька камер до моделі?
2. Як можна встановити рівень прозорості?
3. Як можна переглянути готову модель?

## Практична робота № 13
## МОДЕЛЮВАННЯ ДОСТАВКИ ПАЛЕТ ВАНТАЖНИМИ АВТОМОБІЛЯМИ

**Мета:** набути практичних навичок щодо моделювання доставки палет за допомогою навантажувальної техніки.

### Хід виконання

У цій частині нашого підручника ми додамо вантажівки, які доставляють піддони до майстерні. Почнемо зі створення типу агента для їх представлення.

1. На палітрі *Process Modeling Library* перетягніть елемент *Agent Type* на головну діаграму.

2. У майстрі нового агента виконайте такі дії:

– у полі *agent* введіть *truck;*

– натисніть *next;*

– на наступній сторінці майстра розгорніть розділ *road transport* у списку ліворуч, а потім клацніть на тривимірній анімаційній фігурці *truck;*

– натисніть «Готово». Додамо ще два елементи до нашої мережі: вузол, де будуть з’являтися вантажівки, і шлях, яким вони будуть переміщуватися до приймального пункту.

3. Відкрийте головну діаграму.

4. Під палітрою *Space Markup* клацніть на елементі *Point Node* та перетягніть його до входу під’їзду (рис. 13.1).

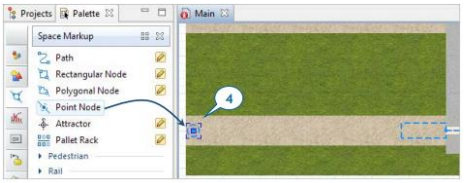



Рисунок 13.1 – Палітра *Space Markup*

5. Назвіть вузол *exitNode*.

6. Намалюйте шлях для з’єднання *exitNode* з приймальною станцією (рис. 13.2).

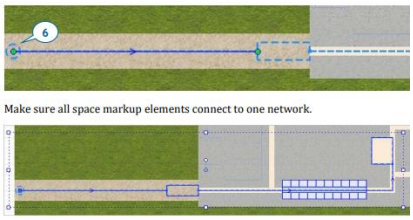



Рисунок 13.2 – Палітра *ReceivingDock*

7. Створіть іншу блок-схему процесу, щоб визначити логіку руху вантажівки, підімкнувши блоки бібліотеки моделювання процесів у такому порядку (рис. 13.3).

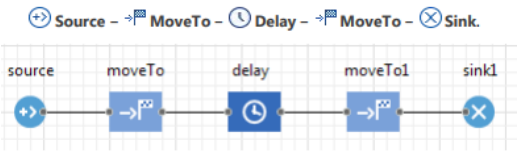



Рисунок 13.3 – Блоки бібліотеки моделювання процесу

1. Блок *Source* створює вантажівку.

2. Перший блок *MoveTo* веде вантажівку до входу в майстерню. Блок-схема *MoveTo* блокує переміщення агентів у нові місця в мережі. Якщо ресурси приєднані до агента, вони будуть рухатися разом з ним.

3. Блок *Delay* імітує розвантаження палет.

4. Другий блок *MoveTo* відганяє вантажівку.

5. Блок раковини знімає вантажівку з моделі.

8. Назвіть вихідний блок *sourceDeliveryTrucks*.

9. В області властивостей блока *sourceDeliveryTrucks* виконайте наведені нижче дії, щоб новий агент власного типу «Вантажівка» прибував до входу на під’їзд раз на годину з певною швидкістю: у списку «Прибуття визначено» клацніть на «Час між прибуттями»;

– у полі між прибуттями введіть 1 і виберіть години зі списку праворуч, у списку «Новий агент» натисніть «Вантажівка»;

– у списку «Розташування прибуття» клацніть на *network/gis node;*

– у списку *node* клацніть на  *exitnode;*

– у полі «Швидкість» введіть 40 і виберіть кілометри на годину зі списку праворуч (рис. 13.4).

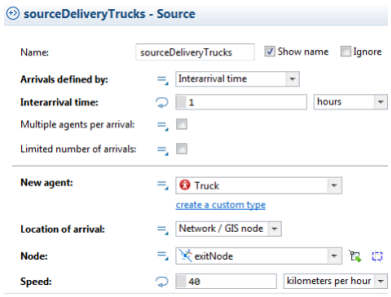



Рисунок 13.4 – Коробка швидкості

10. Назвіть перший блок *MoveTo driveToDock*.

11. В області властивостей блока *drivingToDock* у списку *Node* клацніть на *receiveDock*, щоб установити пункт призначення агента (рис. 13.5).

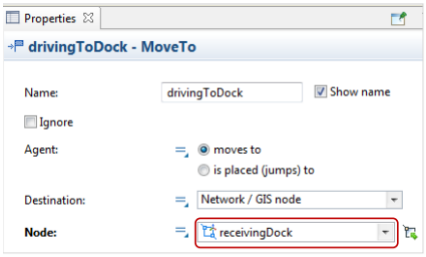



Рисунок 13.5 – Область властивостей

12. Перейменуйте блок *Rename* the *Delay*.

13. В області властивостей блока вивантаження виконайте такі дії:

– в області *type* клацніть на *until stopdelay();*

– у списку «Розташування агента» натисніть на *receiveDock* (рис. 13.6).

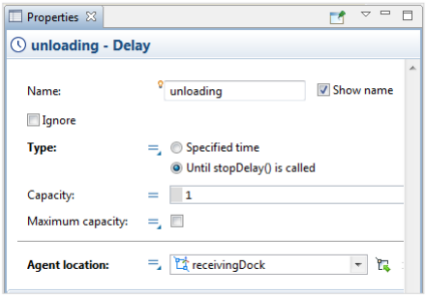



Рисунок 13.6 – Список розташування агента

Тривалість операції залежить від того, наскільки швидко піддони розвантажуються та вивозяться навантажувачами. Ми вважатимемо цю операцію завершеною, коли блок *RackStore* завершить зберігання піддонів і ми змоделюємо це, змінивши режим роботи блока *Delay*.

Програмне керування часом затримки. Зазвичай ви вказуєте час затримки для роботи блока затримки. Це може бути фіксована тривалість (наприклад п’ять хвилин) або стохастичний вираз, який створює час затримки (наприклад *triangular)* (1, 2, 6). Ви також можете програмно контролювати тривалість операції та, за необхідності, зупиняти затримку, викликаючи відповідну функцію блока. Якщо вам потрібно припинити очікування всіх агентів, які перебувають у затримці, викличте функцію блока *stopDelayForAll*(). Інша функція – *stopDelay(agent)* – завершує операцію та звільняє вказаного агента.

14. Назвіть другий блок *MoveTo driveToExit*.

15. У *drivingToExit* в області властивостей блока у списку вузлів клацніть на *exitNode*, щоб установити вузол призначення (рис. 13.7).

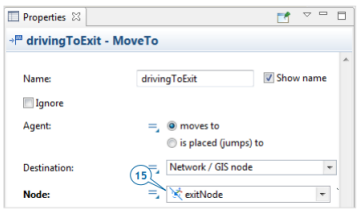



Рисунок 13.7 – *ExitNode* для встановлення вузла призначення

Два блоки *Source* нашої моделі генерують два типи агентів: вантажівки, які з’являються щогодини, і піддони, які генеруються кожні п’ять хвилин. Оскільки ми хочемо, щоб піддони з’являлися, коли вантажівка розвантажується, ми змінимо режим прибуття для вихідного блока, який їх генерує.

Управління генерацією агентів. Ви можете налаштувати вихідний блок на генерацію агентів через визначені проміжки часу, встановивши для параметра *Arrivals* блока значення *Calls of inject()*. Ви зможете керувати створенням агента під час виконання, викликавши функцію *inject(int n)* блока. Ця функція генерує задану кількість агентів під час виклику. Встановити кількість агентів, які згенерує блок, можна за допомогою аргументу функції, наприклад *sourcePalltets.inject(12)*.

16. В області властивостей блока *sourcePallets* у списку «Прибуття визначено» клацніть на «Виклики функції inject()» (рис. 13.8).

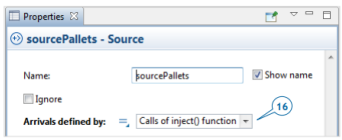



Рисунок 13.8 – Надходження, визначені списком

17. Щоб блок *sourcePallets* генерував піддони, коли вантажівка в’їжджає в блок розвантаження, потрібно зробити таке:

– в області властивостей блока вивантаження розгорніть розділ «Дії»;

– у полі *on enter* введіть таке: *sourcepallets.inject(16)*; функція *java* гарантує, що наша модель генерує 16 піддонів кожного разу, коли вантажівка починає розвантажуватися (рис. 13.9).

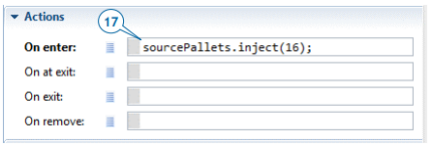



Рисунок 13.9 – Розділ «Дії»

Тепер, коли ми додали вантажівки до нашої моделі, зробимо так, щоб перша вантажівка з доставкою з'являлася під час запуску моделі і нам не доводилося чекати, поки промине година модельного часу.

18. В області «Властивості» типу *Main agent* розгорніть розділ *Agent actions*, а потім у полі введіть функцію *Java: sourceDeliveryTrucks.inject(1)* (рис. 13.10).

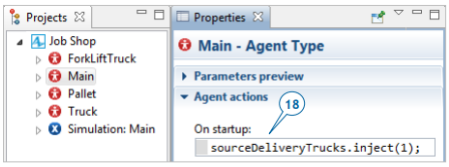



Рисунок 13.10 – Розділ «Дії агента»

Код запуску моделі. Код запуску моделі виконується на завершальній стадії ініціалізації моделі після того, як блоки моделі сконструйовано, підімкнено та ініціалізовано. Це місце для додаткової ініціалізації та запуску дій агента, таких як події.

19. В області властивостей блоку *storeRawMaterial* розгорніть розділ *Actions* та в полі «При виході» введіть таке: *if( self.queueSize() == 0 ) unloading.stopDelayForAll()*. У цьому прикладі *self* – це ярлик, який ми використовуємо для посилання на блок *storeRawMaterial* з його власної дії (рис. 13.11).

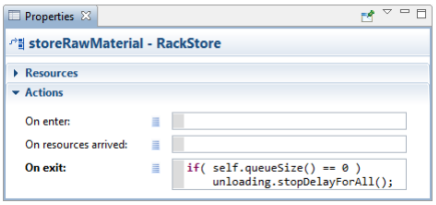



Рисунок 13.11 – Блок *StoreRawMaterial*

Якщо в черзі зберігання немає піддонів, час затримки блока розвантаження закінчується (тобто викликається *stopDelayForAll())* і вантажівка залишає блок розвантаження та входить до наступного блока блок-схеми *driveToExit*.

20. Запустіть модель.

21. Якщо вантажівка вирівняна неправильно (рис. 13.12), для корекції потрібно виконати такі дії (рис. 13.12):

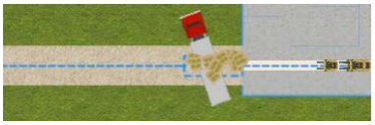


Рисунок 13.12 – Помилка у вирівнюванні вантажівки

– у дереві проєктів двічі клацніть по типу агента вантажівки, щоб відкрити його діаграму та переглянути анімаційну фігуру вантажівки;

– у графічному редакторі виберіть форму вантажівки, а потім використайте круглу ручку або властивість *Rotation Z,°* в області властивостей *Position* форми, щоб повернути вантажівку на 180° (рис. 13.13).

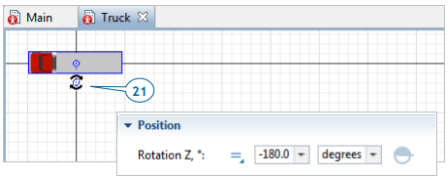



Рисунок 13.13 – Область властивостей позиції

Ми змінили положення фігури вантажівки, але нам також потрібно буде змінити налаштування *AnyLogic* за замовчуванням, щоб переконатися, що програма не обертає її вдруге.

22. Щоб змінити налаштування *AnyLogic* за замовчуванням, потрібно зробити таке:

– в області *Projects* натисніть на *Truck;*

– в області властивостей типу агента вантажівки клацніть на стрілку, щоб розгорнути область розмірів і переміщення;

– зніміть прапорець *Rotate animation towards movement check box* (рис. 13.14).

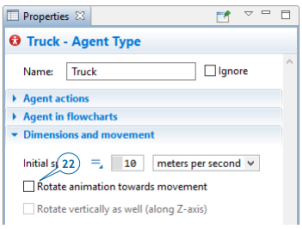



Рисунок 13.14 – Прапорець «Повернути анімацію до руху»

23. Відкрийте головну діаграму.

24. Щоб забезпечити правильне розташування піддонів у вузлі мережі *receiveDock*, відкрийте палітру *Space Markup* і перетягніть атрактор у *receiveDock*. Нехай він буде повернутий на вхід (рис. 13.15).


Рисунок 13.15 – Атрактор у приймальну станцію

Атрактори у вузлах дозволяють контролювати розташування агента всередині вузла:

1. Якщо вузол визначає пункт призначення, до якого рухаються агенти, атрактори визначають точні цільові точки всередині вузла.

2. Якщо вузол визначає місце очікування, атрактори визначають точні точки, коли агенти чекатимуть усередині вузла. Атрактори також визначають орієнтацію анімації агента, допоки агент очікує всередині вузла. У цьому випадку ми використовуємо атрактор для конкретної мети. Ви можете додати атрактори, перетягнувши їх по-одному на головну діаграму. Однак якщо атрактори утворюють регулярну структуру, варто скористатися спеціальним майстром, щоб додати кілька атракторів одночасно. Майстер пропонує кілька різних режимів створення, а також можливість очистити всі атрактори, і ви

можете відобразити все це, натиснувши на кнопку *Attractors* в області *Properties* вузла.

25. Запустіть модель, щоб перевірити поведінку вантажівки. Анімація має працювати так, як ми очікуємо (рис. 13.16).

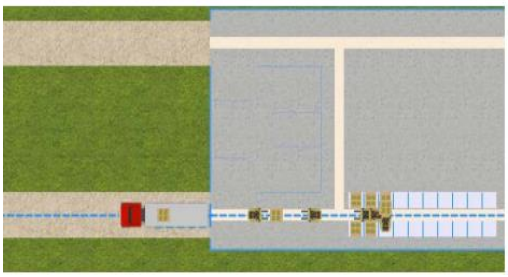


Рисунок 13.16 – Область властивостей

### Питання для самоконтролю

1. Від чого залежить тривалість операції?
2. Як можна забезпечити правильне розташування піддонів у вузлі мережі?
3. Чим дозволяють керувати атрактори у вузлах?

## Практична робота № 14
## МОДЕЛЮВАННЯ ВЕРСТАТІВ ІЗ ЧПК

**Мета:** набути практичних навичок щодо моделювання верстатів із ЧПК.

### Хід виконання

Змоделюємо верстати з ЧПК, які обробляють сировину. Ми почнемо з розмітки простору та використання точкових вузлів для визначення розташування верстатів із ЧПК:

1. На палітрі розмітки простору перетягніть елемент *Point Node* на макет робочої майстерні та назвіть його *nodeCNC1*.

2. Скопіюйте цей вузол, щоб позначити місце для другого верстата з ЧПК. *AnyLogic* назве другий вузол *nodeCNC2* (рис. 14.1).

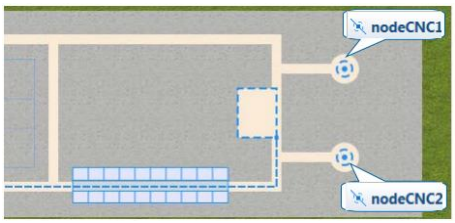



Рисунок 14.1 – Другий вузол *nodeCNC2*

Потрібно створити шляхи для підімкнення обох вузлів до мережі. Вилкові навантажувачі моделі знадобляться, щоб дістатися до верстатів із ЧПК.

3. На палітрі *Space Markup* клацніть по елементу *Path* і намалюйте контури, як показано на рисунку 14.2. Щоб підімкнути шлях до вузла точки, клацніть по центу вузла точки.

ПРИМІТКА. Переконайтеся, що шляхи, які ви малюєте, з'єднують *nodeCNC1* і *nodeCNC2* з мережею. Ви можете перевірити з'єднання шляху, клацнувши двічі, щоб вибрати його. Якщо шлях підімкнено до мережі, навколо його кінцевих точок з'являться блакитні підсвічування (рис. 14.2).

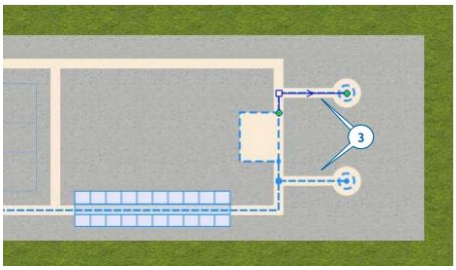



Рисунок 14.2 – Палітра *Space Markup*

Машина з ЧПК є ресурсною одиницею, і ми додамо її до нашої моделі, створивши тип ресурсу та використавши блок *ResourcePool* для визначення пулу ресурсів.

4. На палітрі *Process Modeling Library* клацніть і перетягніть блок *ResourcePool* на головну діаграму.

5. В області властивостей блока *ResourcePool* виконайте такі дії:

– у полі «Ім’я» введіть *cnc*;

– у списку «Тип ресурсу» виберіть статичний, щоб відобразити факт, що це статичний ресурс (рис. 14.3).

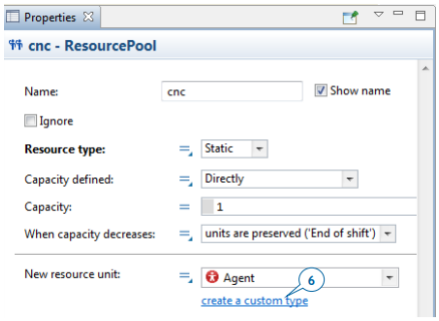



Рисунок 14.3 – Список типів ресурсів

Коли пул ресурсів завершено, можна створити новий тип ресурсу.

6. У списку «Новий блок ресурсу» клацніть на посиланні «Створити настроюваний тип».

7. У майстрі нового агента виконайте такі дії:

– у полі «Ім’я типу агента» введіть *cnc*;

– натисніть «Далі»;

– на наступній сторінці майстра розгорніть останній розділ (верстати з ЧПК) і виберіть вертикальний обробний центр з ЧПК 2, стан 1;

– натисніть «Готово».

8. Закрийте діаграму нового типу ЧПК і поверніться до головної діаграми.

9. В області властивостей блока *cnc ResourcePool*, щоб розмістити два верстати з ЧПК нашої моделі в місцях, визначених точковими вузлами – *nodeCNC1* і *nodeCNC2*, виконайте таке:

– у списку «Визначена потужність» клацніть на «За домашнім розташуванням». Параметр «За домашнім розташуванням» встановлює кількість ресурсів, що дорівнює кількості вузлів домашнього розташування, які ви встановили для цього пулу ресурсів;

– натисніть кнопку з плюсом, а потім додайте *nodeCNC1* і *nodeCNC2* до списку «Домашнє розташування» (вузлів). Після додавання вузлів список має нагадувати малюнок, зображений на рисунку 14.4.

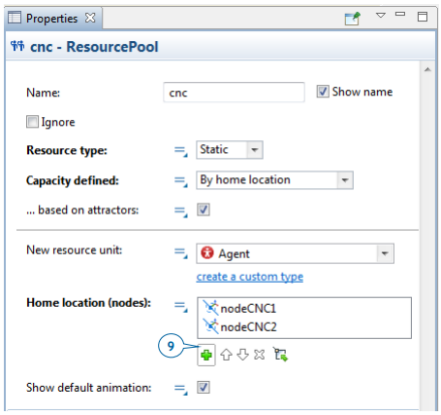



Рисунок 14.4 – Кнопка «плюс»

Ми готові змінити блок-схему, яку використовує наша модель для визначення поведінки піддонів, додавши блок *Seize*, який захопить верстат із ЧПК. Блок затримки буде імітувати обробку сировини верстатом із ЧПК, а блок випуску звільнить верстат із ЧПК, щоб він міг обробити сировину наступної палети.

ПРИМІТКА. Пам’ятайте, що блок-схема нашої моделі вже містить блок *pickRawMaterial*, який імітує рухомий ресурс (навантажувачі), який доставляє піддони до машини з ЧПК.

10. У блок-схемі, яка визначає поведінку піддонів, перетягніть блоки *pickRawMaterial* і *sink* праворуч, щоб звільнити місце для нового блока.

11. На палітрі *Process Modeling Library* перетягніть блок Seize і вставте його в блок-схему палет після блоку *rawMaterialinStorage* (рис. 14.5).

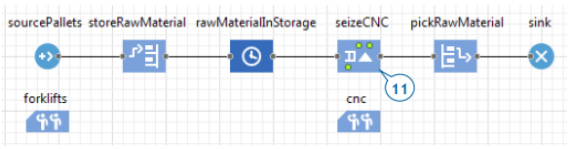



Рисунок 14.5 – Блок *RawMaterialinStorage*

12. В області властивостей блока захоплення виконайте такі дії:

– у полі «Ім’я» введіть *seizeCNC;*

– під параметром «Набори ресурсів» клацніть на кнопці з плюсом, а потім *cnc*. Виконання цього кроку гарантує, що блок Seize захопить один ресурс із пулу ресурсів *CNC*.

13. В області властивостей блока блок-схеми *pickRawMaterial* виконайте такі дії:

– у списку *Destinationis* клацніть на «Захоплена одиниця ресурсу»;

– у списку ресурсів клацніть на *cnc*. Цей блок моделюватиме транспортування піддонів до захопленого верстата з ЧПК, а не зону паркування навантажувачів.

14. Щоб імітувати обробку сировини верстатом з ЧПК, виконайте таке:

– додайте блок *Delay*, розмістіть його відразу після *pickRawMaterial* і назвіть його *Processing* (рис. 14.6);

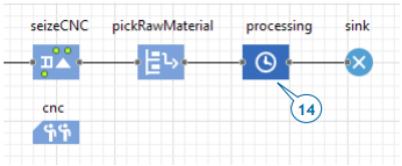



Рисунок 14.6 – Додавання блока затримки

– у полі «Час затримки» введіть *triangular* (2, 3, 4) і виберіть хвилини зі списку праворуч;

– установіть прапорець *Maximum capacity check box*, щоб дозволити машинам обробляти кілька палет. Кожен агент, який приходить до блока затримки, повинен мати один із двох верстатів з ЧПК нашої моделі.

16. На палітрі *Process Modeling Library* перетягніть блок *Release* на блок-схему палет і розмістіть його після блока обробки.

17. Назвіть цей блок *Release releaseCNC* (рис. 14.7).

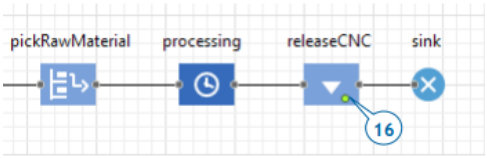



Рисунок 14.7 – Блок розблокування *CNC*

Якщо ви запустите модель, то побачите, що, хоча процеси моделюються правильно, 3D-анімація малює палету всередині форми верстата з ЧПК. Це відбувається, коли верстат з ЧПК, піддон, який він обробляє, і розташування анімації використовують один і той самий точковий вузол. Щоб вирішити проблему, нам потрібно перемістити верстат з ЧПК праворуч і повернути його лицевою стороною до палети (рис. 14.8).

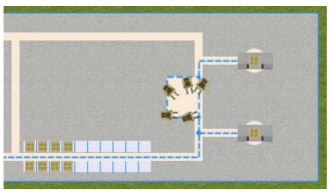


Рисунок 14.8 – Лицьова сторона до піддону

18. У вікні *Projects* двічі клацніть по типу агента ЧПК, щоб відкрити його діаграму.

19. Перемістіть анімацію вправо та поверніть форму верстата з ЧПК, використовуючи круглу ручку або встановивши властивість «Поворот фігури на 90°» (рис. 14.9).

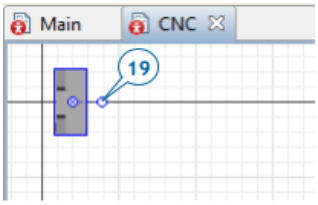



Рисунок 14.9 – Властивість повороту на 90°

Ми готові використати дві подібні фігури 3D-анімації для анімації верстата з ЧПК: одна фігура представлятиме непрацюючий верстат, а інша – верстат, який обробляє сировину. Ми визначимо динамічні значення для властивості *Visible* кожної форми, що дозволить нашій моделі використовувати стан верстата з ЧПК для визначення того, яку форму модель відображатиме під час виконання.

20. Щоб змінити налаштування видимості фігури CNC-анімації, зробіть таке:

– виберіть форму анімації ЧПК;

– наведіть вказівник миші на піктограму статичних параметрів, яка відображається поряд із видимою міткою, і натисніть на динамічному значенні (рис. 14.10).

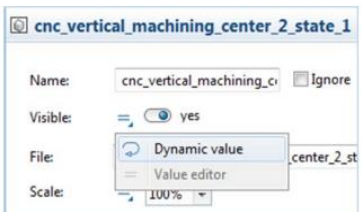



Рисунок 14.10 – Динамічне значення

Піктограма зміниться на піктограму динамічних властивостей і з’явиться поле, у якому можна визначити динамічний вираз значення. Ви можете використовувати поле для введення виразу *Java*, який повертає значення *true* або *false*. У полі введіть *isBusy()*.

Ця стандартна функція для ресурсу *AnyLogic* повертає значення *true*, якщо ресурс зайнятий. У нашому випадку функція зробить відображення форми 3D-анімації, коли верстат з ЧПК обробляє сировину (рис. 14.11).

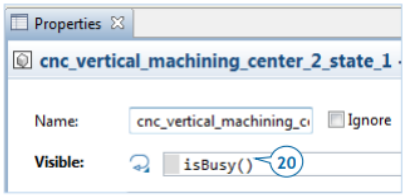



Рисунок 14.11 – Відображення фігури 3D-анімації

Динамічні властивості. Коли ви обираєте вираз для динамічного значення властивості, наша модель переоцінить вираз у кожному кадрі анімації під час виконання, а потім використає отримане значення, як поточне значення властивості. Надання динамічних значень для положення, висоти, ширини або кольору фігури дозволяє користувачам *AnyLogic* анімувати моделі. Якщо ви не введете динамічне значення, властивість зберігає стандартне статичне значення протягом усього моделювання:

1. Блоки блок-схеми можуть мати статичні параметри, які зберігають те саме значення протягом симуляції, якщо функція *set_parameterName* (нове значення) не змінить його. Динамічні властивості, значення яких переоцінюється кожного разу, коли новий агент входить до блока. Параметри коду, які дозволяють визначати дії, що виконуватимуться в ключовий момент у блоці блок-схеми, наприклад дії при вході або дії при виході. У більшості випадків ви знайдете параметри коду в області властивостей блока блок-схеми в розділі «Дії».

2. Маленький трикутник біля піктограми параметра показує, що ви можете клацнути на піктограмі та перейти від редактора статичного значення до поля, де можна ввести динамічно переоцінений вираз значення.

21. Виконайте наступне, Щоб додати ще одну форму анімації, яка буде видима лише тоді, коли верстат із ЧПК не оброблятиме сировину, потрібно зробити таке:

– відкрийте палітру 3D-об’єктів, яка містить готові до використання 3D-об’єкти *AnyLogic*;

– розгорніть область «Верстати з ЧПК» і перетягніть фігуру «Стан 2 вертикального обробного центру з ЧПК 2» на діаграму з ЧПК;

– поверніть фігуру та помістіть її над першою фігурою анімації;

– у вікні *Visible* перейдіть до редактора динамічних значень і введіть *isIdle()* як динамічний вираз для властивості *Visible* форми (рис. 14.12).

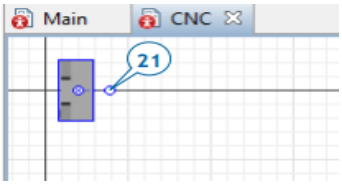



Рисунок 14.12 – Видиме поле

22. Розгорніть розділ *People* палітри 3D-об’єктів і перетягніть фігуру «Робітник» на діаграму ЧПК (рис. 14.13).

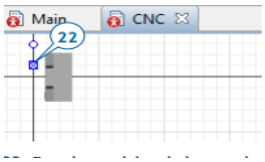



Рисунок 14.13 – Розділ «Люди»

23. Запустіть модель і поспостерігайте за процесом. Ви побачите, як вилкові навантажувачі транспортують піддони до верстатів із ЧПК для обробки. Ви також повинні побачити анімовані верстати з ЧПК, які змінюють тривимірні форми залежно від свого стану (рис. 14.14).

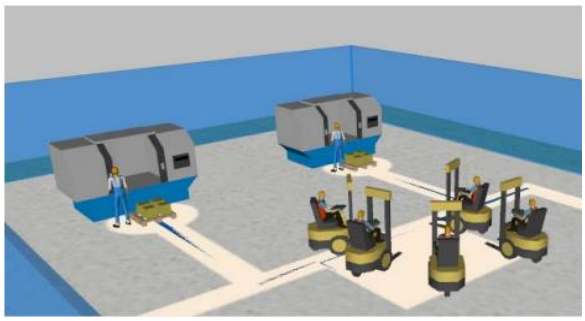


Рисунок 14.14 – 3D-форми залежно від їх стану

Ми закінчили нашу просту модель, яка імітує процес виробництва та доставки в невеликій майстерні. Ви отримали базові знання про ресурси *AnyLogic* і про те, як з ними потрібно працювати. Ви також знаєте, як використовувати блок-схему, створену з блоків бібліотеки моделювання процесів, для визначення логіки процесу. Вашим наступним кроком може бути моделювання того, як

піддони з готовими деталями переміщуються в іншу зону зберігання на транспортному доку, де вони залишаться до відправлення. Ви вже використовували блоки, які вам знадобляться для моделювання цієї частини процесу, тому можете спробувати застосувати ці знання самостійно. У наступному завданні також використовуватиметься блок-схема, зорієнтована на процес, але цього разу це буде модель пішохода.

Симуляція руху пішоходів є важливою частиною будівництва, розширення чи реконструкції таких об'єктів, як торгові центри, аеропорти, вокзали та стадіони. Ці аналізи можуть допомогти архітекторам поліпшити свої проєкти, власникам об'єктів переглянути можливі зміни в будівлі, а цивільним органам влади змоделювати можливі шляхи евакуації. Оскільки пішохідні потоки можуть бути складними, вони потребують масштабного моделювання. Пішоходи дотримуються основних правил, які були визначені шляхом детальних теоретичних досліджень; вони рухаються із заздалегідь визначеною швидкістю, уникають фізичних просторів, таких як стіни, а також інших людей, і вони використовують інформацію про натовпи, які їх оточують, щоб регулювати свою відстань і швидкість. Результати неодноразово підтверджувалися польовими дослідженнями та застосуваннями клієнтів. Ви можете створювати такі показники, як загальний час у дорозі між певними точками, і змінювати свої експерименти, щоб підкреслити ці показники під час пікових заторів. Нарешті, ви можете імпортувати фонові макети, плани поверхів і карти та створювати кілька 3D-видів, які полегшать аналіз пішохідного потоку.

*AnyLogic* може допомогти вам вирішити проблеми щодо руху пішоходів, а саме:

– розрахувати час та пропускну здатність. Припустимо, що ви проєктуєте супермаркет, метро, залізничну станцію чи будівлю аеропорту. Якщо ваша мета полягає в тому, щоб створити макет, який мінімізує час у дорозі та гарантує, що пішохідні потоки не заважатимуть один одному, симуляція *AnyLogic* може легко перевірити наявні умови – звичайні, спеціальні чи умови максимального обсягу;

– проаналізувати вплив на рух пішоходів. Керівники місць з інтенсивним рухом людей, таких як тематичні парки, музеї та спортивні стадіони, можуть використовувати симуляцію, щоб зрозуміти, як такі зміни, як новий кіоск або переміщення рекламної панелі, вплинуть на їхню роботу, тривалість руху пішоходів і досвід клієнтів;

– проаналізувати план евакуації. Збільшення частоти природних і антропогенних катастроф робить важливою оцінку та оптимізацію планів евакуації. Моделювання надзвичайних подій може допомогти агентствам з управління надзвичайними ситуаціями розробити ефективні плани евакуації, які рятують життя.

## Питання для самоконтролю

1. Які дії потрібно виконати, щоб підімкнути шлях до вузла точки?

2. Які дії потрібно виконати, щоб змінити налаштування видимості фігури *CNC*-анімації?

3. Як можна змінити положення 3D-анімації, яка малює палету всередині форми верстата з ЧПК?

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Моделювання ланцюгів постачання за допомогою AnyLogic [Електрон. ресурс] – Електрон. текст. дані. – Режим доступу: https://www.anylogic.com/supply-chains/, вільний (дата звернення: 06.09.2024). – Назва з екрана.

2. Ballou R. H. Business Logistics/Supply Chain Management / R. H. Ballou. – Upper Saddle River (New Jersey) : Pearson/Prentice Hall, 2004. – 816 p.

3. Christopher M. Logistics and Supply Chain Management / M. Christopher. – Harlow (England) : Pearson, 2016. – 328 p.

4. Chopra S. Supply Chain Management : Strategy, Planning, and Operation / S. Chopra, P. Meindl. – Harlow (England) : Pearson, 2020. – 640 p.

5. Kibira D. Modeling and Simulation of Supply Chains Using AnyLogic [Electronic resource] : / D. Kibira, C. McLean // Journal of Manufacturing Systems. – Electronic text data. – 2018. – Vol. 48. – P. 93–104. – . Regime of access: https://doi.org/10.1016/j.jmsy.2018.05.003, free (date of the application: 06.09.2024). – Header from the screen.

6. Mueller S. Simulation-Based Optimization of Logistic Networks Using AnyLogic [Electronic resource] / S. Mueller, M. Lütjen, J. C. Bendul // International Journal of Production Research. – Electronic text data. – 2017. – Vol. 55, is. 18. – P. 5359–5375. – Regime of access: https://doi.org/10.1080/00207543.2017.1308574, free (date of the application: 06.09.2024). – Header from the screen.

7. Rushton A. The Handbook of Logistics and Distribution Management / A. Rushton, P. Croucher, P. Baker. – London : Kogan Page, 2017. – 912 p.

8. Sokolowski J. A. Modeling and Simulation Fundamentals : Theoretical Underpinnings and Practical Domains / J. A. Sokolowski, C. M. Banks. – Hoboken (New Jersey) : John Wiley & Sons, 2010. – 466 p.

*Електронне навчальне видання*

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання практичних робіт та організації самостійної роботи
з навчальної дисципліни

**«АНАЛІЗ ЛОГІСТИЧНИХ ПРОЦЕСІВ»**

*(для здобувачів першого (бакалаврського) рівня вищої освіти
денної і заочної форм навчання зі спеціальності
073 – Менеджмент, освітня програма «Логістика»)*

Укладач **ГАЛКІН** Андрій Сергійович

Відповідальний за випуск *Т. В. Луценко*
*За авторською редакцією*
Комп’ютерне верстання *О. О. Грєкова*

План 2024, поз. 261М

---

Підп. до друку 31.10.2024. Формат 60 × 84/16.
Ум. друк. арк. 6,9.

Видавець і виготовлювач:
Харківський національний університет
міського господарства імені О. М. Бекетова,
вул. Чорноглазівська (Маршала Бажанова), 17, Харків, 61002.
Електронна адреса: office@kname.edu.ua
Свідоцтво суб’єкта видавничої справи:
ДК № 5328 від 11.04.2017.