Київський національний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Автоматизація обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі»

Студента 4 курсу, 10 групи,
спеціальності
122 «Комп'ютерні науки»

_____
підпис студента

Адаменко
Назар
Леонідович

Науковий керівник
доктор фізико-математичних наук,
професор

_____
підпис керівника

Демідов Павло
Георгійович

Гарант освітньої програми
кандидат технічних наук, доцент

_____
підпис керівника

Демідов Павло
Георгійович

Київ 2021

Київський національний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую
Зав. кафедри _____Пурський О.І.
«18» грудня 2020р.

Завдання
на випускну кваліфікаційну роботу (проект) студенту

Адаменко Назар Леонідович
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)
«Автоматизація обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі».
Затверджена наказом ректора від *«15» грудня 2020 р. № 3780*
2. Строк здачі студентом закінченої роботи *31 травня 2021 року*
3. Цільова установка та вихідні дані до роботи.
Мета роботи: автоматизація обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі.
Об'єкт дослідження: процес автоматизації обліку замовлень у сервісному центрі.
Предмет дослідження: інформаційні системи і технології для створення автоматизації обліку замовлень на діагностику та ремонт побутової техніки.
4. Перелік графічного матеріалу_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Демідов П.Г.	22.12.2020 р.	22.12.2020 р.
2	Демідов П.Г.	22.12.2020 р.	22.12.2020 р.
3	Демідов П.Г.	22.12.2020 р.	22.12.2020 р.

6. Зміст випускної кваліфікаційної роботи (проекту) (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. Дослідження та аналіз автоматизованого обліку замовлень у сервісному центрі.

1.1. Особливості та принципи організації обліку замовлень.

1.2. Огляд і аналіз існуючих аналогів, що реалізовують автоматизований облік замовлень на діагностику у сервісних центрах.

1.3. Характеристика та вибір програмних засобів розробки комп'ютерної систем управління діяльністю сервісного центру.

РОЗДІЛ 2. Проектування системи автоматизованого обліку замовлень у сервісному центрі.

2.1. Постановка задачі на моделювання ведення замовлень та обліку їх у сервісному центрі.

2.2. Математична модель розрахунку показників ведення замовлень у сервісному центрі.

2.3. Імітаційна модель ведення замовлень в сервісному центрі.

2.4. Програмна система обліку замовлень у сервісному центрі.

РОЗДІЛ 3. Розробка імітаційної моделі ведення та обліку замовлень та її програмна реалізація.

3.1. Об'єкти та елементи моделі ведення замовлень в середовищі системи

AnyLogic.

3.2. Рішення задачі ведення замовлень у середовищі системи AnyLogic.

3.3. Рішення задачі обліку замовлень у середовищі С++.

3.3. Тестування системи моделювання ведення та обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі.

РЕЗУЛЬТАТИ ТА ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*05.10.2019*	*05.10.2019*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*18.12.2020*	*18.12.2020*
3	*Вступ*	*03.02.2021*	*03.02.2021*
4	*РОЗДІЛ 1. Аналітичне дослідження специфіки діяльності закладів вищої освіти в Україні*	*26.02.2021*	*26.02.2021*
5	*РОЗДІЛ 2. Розробка моделі перевірки та оцінювання знань студентів*	*06.04.2021*	*06.04.2021*
6	*РОЗДІЛ 3. Інформаційна система перевірки та оцінювання знань студентів*	*12.05.2021*	*12.05.2021*
7	*Висновки*	*14.05.2021*	*14.05.2021*
8	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*20.05.2021*	*20.05.2021*
9	*Попередній захист випускної кваліфікаційної роботи*	*26.05.2021*	*26.05.2021*
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*27.05.2021*	*27.05.2021*
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*31.05.2021*	*31.05.2021*
13	*Публічний захист випускної кваліфікаційної роботи*	*15.05.2021*	*15.05.2021*

8. Дата видачі завдання «22» грудня 2020 р.

9. Керівник випускної кваліфікаційної роботи (проекту)
Демідов П.Г.
(прізвище, ініціали, підпис)

10. Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник
Адаменко Н.Л.
(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи (проекту)

Керівник випускної кваліфікаційної роботи (проекту)
_____ 31.05.2021 р.
(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу (проект)

Випускна кваліфікаційна робота (проект) студента Адаменко Н. Л.
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_____Демідов П.Г.____
(підпис, прізвище, ініціали)

Завідувач кафедри _____Пурський О.І.___
(підпис, прізвище, ініціали)
«_____»_____2021 р.

Анотація

У випускній кваліфікаційній роботі здійснено розробку імітаційної моделі ведення замовлень та комп'ютерну програму обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі з метою підвищення ефективності керування сервісним центром. Проаналізовано особливості функціонування сервісної галузі та основні потреби в обліку замовлень. Створено імітаційну модель ведення замовлень та написано програмний код для обліку замовлень.

Ключові слова: сервісний центр, ведення замовлень, облік замовлень, імітаційна модель, комп'ютерна програма.

Annotation

In the final qualification work, a simulation model of order management and a computer program for accounting orders for diagnostics and repair of household appliances in the service center were developed in order to increase the efficiency of service center management. The peculiarities of the service industry functioning and the main needs in order accounting are analyzed. An imitation model of order management has been created and a program code for order accounting has been written.

Keywords: service center, order management, order accounting, simulation model, computer program.

ЗМІСТ

ВСТУП

Актуальність теми дипломної роботи «Автоматизація обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі» обумовлена тим, що ведення обліку ручним способом в умовах швидких темпів технологічного прогресу вважається не тільки неефективним і архаїчним, але й безперспективним, через те, що всі сучасні сервісні центри являють собою складну систему складів, що вимагає оперативного обліку замовлень та надання відповідних послуг клієнтам. Без автоматизації обліку досягти успіху у цьому бізнесі досить складно. Тому дослідження автоматизації обліку замовлень дасть змогу довести, що:

- використання комп'ютерної техніки дає змогу знижувати трудомісткість роботи;
- контролювати правильність операцій;
- прискорювати оброблення облікової інформації;
- зменшувати обсяги документообігу.

З розвитком технологій, люди купують все більше техніки, яку потрібно встановлювати, обслуговувати та ремонтувати. У порівнянні з минулими роками, попит на діагностику, обслуговування та ремонт побутової техніки стрімко збільшується. Тому сервісні центри, які надають ці послуги, повинні мати саме автоматизований облік замовлень, також автоматизована система обліку дозволяє своєчасно отримувати необхідну інформацію для прийняття ефективних управлінських рішень. За допомогою якої буде набагато легше:

- вести облік і керувати замовленнями;
- оцінювати якість роботи менеджерів;
- гарантувати дотримання термінів.

Мета роботи: розробка моделі ведення замовлень та створення комп'ютерної програми обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі.

Досягнення мети обумовило необхідність вирішення таких завдань:

- аналітичне дослідження способів обліку замовлень у сервісних центрах;
- аналіз існуючих програмних засобів у сфері сервісного обслуговування;
- розробка функціональної імітаційної моделі ведення замовлень;
- розробка комп’ютерної програми для обліку замовлень;
- програмна реалізація.

Об'єкт дослідження: процеси ведення та обліку замовлень.

Предмет дослідження: інформаційні системи і технології в системі ведення та обліку замовлень у сервісному центрі.

Методи дослідження: загальнонауковий аналітичний метод, метод моделювання, метод програмування.

РОЗДІЛ 1.
ДОСЛІДЖЕННЯ ТА АНАЛІЗ АВТОМАТИЗОВАНОГО ОБЛІКУ ЗАМОВЛЕНЬ У СЕРВІСНОМУ ЦЕНТРІ

1.1. Особливості та принципи організації обліку замовлень.

Сервісний центр по ремонту побутової техніки - організація, яка займається усуненням неполадок електроніки та побутової техніки. Головне завдання такої організації - усунути неполадки у роботі обладнання і не допустити нових, забезпечити все необхідне для того, щоб обладнання виконувало свою функцію у власника якомога довше. Надає гарантійний та післягарантійний ремонт електричних пристроїв, забезпечує підтримку користувачів.

Для оптимізації роботи збутових структур в частині оперативної і якісного відпрацювання звернень клієнтів необхідно вирішити ряд завдань. Це можливо за допомогою автоматизації процесів прийняття, відпрацювання та управління зверненнями клієнтів.

Найчастішим факторами необхідності автоматизації ділових процесів сервісних центрах виступають:

- значне підвищення продуктивності праці і зниження трудовитрат;
- підвищення швидкості виконання завдань, пов'язаних з різного роду урахуванням;
- зниження кількості помилок в документації, звітах і т. д.;
- підвищення ефективності обліку, що приводить до збільшення рентабельності підприємства;
- зручність здійснення ділових процесів і документообігу.

Кожен з перерахованих вище пунктів – є вагомим аргументом на користь того, щоб звернути увагу на можливість автоматизації бізнес-процесів компанії.

Завдання оптимізації в цій ситуації є:

1. Оптимізувати прийом і обробку вхідних звернень, стабільне зростання бази клієнтів.

2. Регламентувати фіксацію всіх вхідних звернень клієнтів та інформації по роботі з ними, щоб забезпечити збереження історії взаємин.
3. Розробити механізми та впровадити інструменти контролю над менеджерами, їх оптимальним завантаженням і виконанням плану.
4. Запровадити інструмент аналізу продажів, оцінки результативності кожного етапу.
5. Забезпечити керівництво компанії легкодоступною оперативної звітністю і можливість бути в курсі всіх змін за ключовими показниками ефективності.

Результати автоматизації:

1. Оптимізація пошуку та залучення потенційних клієнтів, стабільне зростання клієнтської бази.
2. Коректний облік замовлень, виключення втрат цільових інтересів від потенційних замовників.
3. Результативна обробка кожного звернення і грамотне ведення обліку замовлень (дзвінка, замовлення з сайту, електронного листа), доведення більшості з них до укладення угоди.
4. Зростання показників конверсії менеджерів і, як наслідок, - обсягів продажів.
5. Підвищення ефективності маркетингових заходів і більш доцільне розподіл бюджету на рекламу.

Для підприємств і організацій, активно розширюють свою діяльність, не тільки за рахунок появи нових клієнтів і партнерів, а й за рахунок збільшення оборотів і надання більш широкого спектру послуг і варіантів співпраці вже наявним клієнтам і партнерам, для підвищення ефективності своєї діяльності особливо важлива автоматизація реєстрації та моніторингу замовлень від таких контрагентів. Ця процедура, в кінцевому підсумку, не тільки істотно полегшує роботу заявника, але також дозволяє виконавцю замовлення заощадити час на отримання та обробку запитів і мінімізувати вплив помилок «людського фактору». Адже якою б високою кваліфікацією

не був фахівець, потрібно віддати належне технічному прогресу: машинна пам'ять і швидкість обробки даних перевершують аналогічні людські показники.

Типова структура системи обробки замовлень та схема передачі замовлень наведені нижче (Рис. 1.1. та Рис. 1.2.).

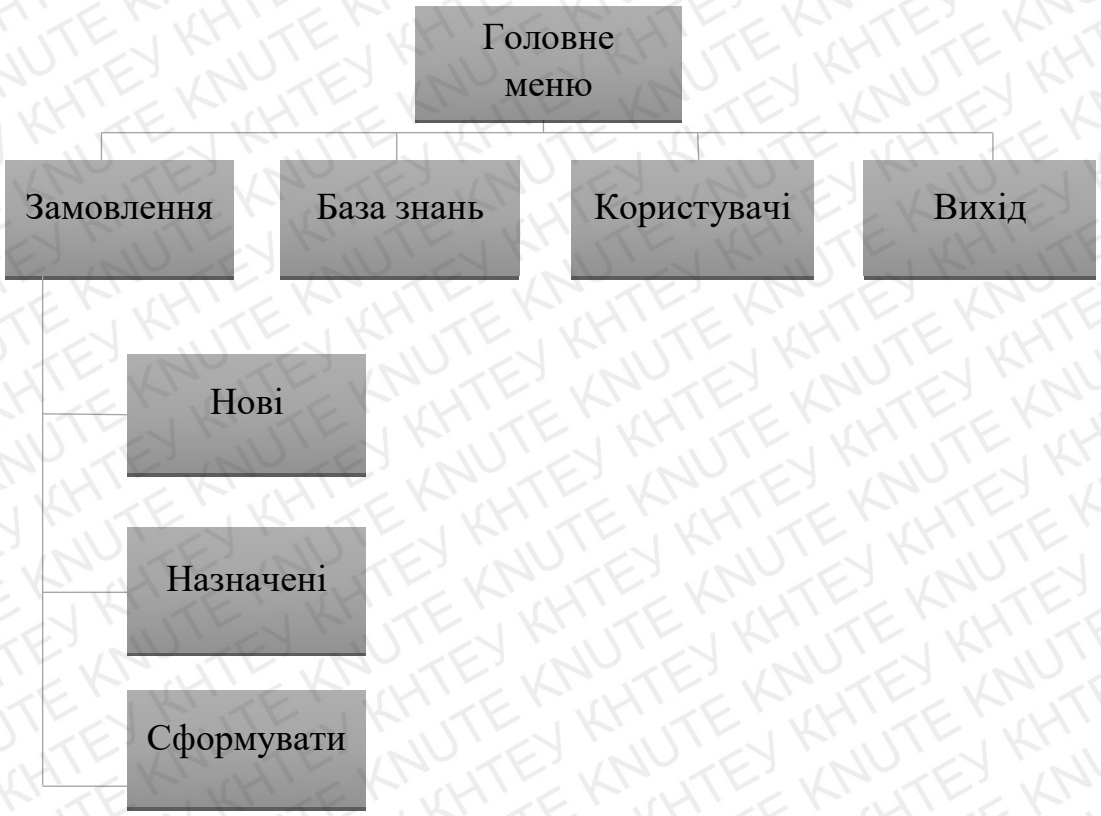

Рис. 1.1. Типова структура системи обробки замовлень.

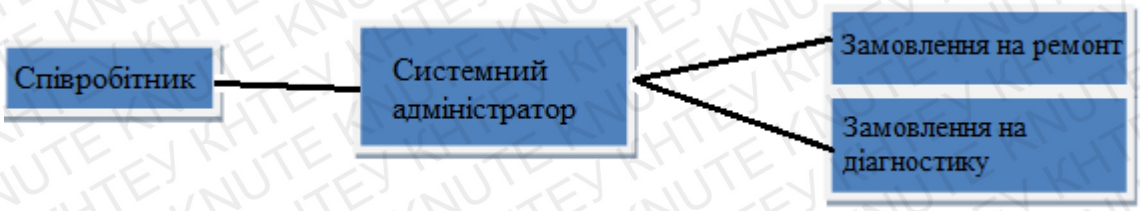

Рис. 1.2. Схема передачі замовлення.

Результатом роботи буде висновок звітів по рахунках діагностики обладнання, робота обладнання, рахунок з надання послуг на обслуговування побутової техніки. Звіти будуть виводитися на друк, за допомогою додатка

автоматизованої системи обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі.

1.2. Огляд і аналіз існуючих аналогів, що реалізовують автоматизований облік замовлень на діагностику у сервісних центрах.

Сучасний ринок інформаційних технологій (IT) пропонує безліч систем обліку замовлень та управління клієнтською базою. Однак за допомогою подібного роду програм можлива автоматизація лише деяких бізнес-завдань. Ведення і управління базою клієнтів вимагає системності та комплексності. Необхідно враховувати всі можливі групи потенційних замовників, всі види і джерела звернень і всі способи отримання прибутку на кожному етапі продажу. Для цього потрібна повна автоматизація роботи відділу продажів, комплексна автоматизована багатофункціональна система.

В результаті впровадження подібного програмно-апаратного комплексу будуть формалізовані бізнес-процеси пошуку, залучення й утримання покупців. Все це повинно бути реалізовано в єдиному інформаційному просторі, що значно підвищить керованість збутових процесів і прибутковість компанії в цілому. До складу таких автоматизованих рішень, як правило, входить:

- налаштування ключових процесів відділу продажів;
- підбір і впровадження комп'ютеризованої програми для більш якісної роботи з інтересами;
- консалтинг і комплексна підтримка.

Далі наведемо найбільш відомі системи та проаналізуємо переваги кожної для автоматизації обліку замовлень для сервісного центру.

1. Система автоматизованого обліку замовлень Okdesk.

Система Okdesk - функціональна і зручна система в проведенні обліку замовлень та в автоматизації всіх процесів b2b, включаючи виїзне та внутрішнє обслуговування послуг компанії - (field service). Головне вікно системи Okdesk наведено на Рис. 1.3.

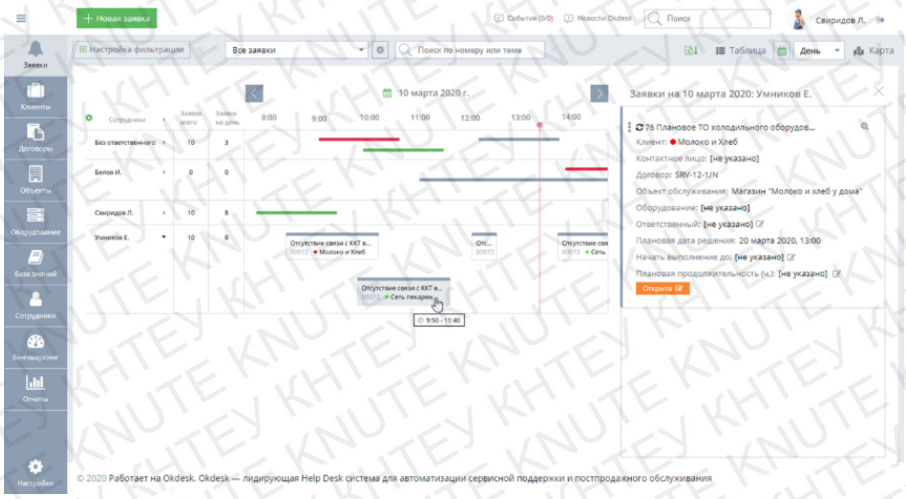

Рис. 1.3. Головне вікно системи Okdesk.

Система Okdesk є хмарним рішенням для автоматизації процесів підтримки та взаємодії з особами в малих і середніх сервісних компаніях включає наступну функціональність:

- Help Desk + CRM;
- вбудований клієнтський портал;
- реєстрацію звернень поштою і з сайту;
- листування з клієнтами;
- функціональність звітів і дашборда для керівників;
- відображення замовлень на карті.

Ця система розроблена на підставі кращих практик більш 10000 компаній в Росії і СНД. Допомагає не втрачати жодного звернення, контролювати виконання замовлень, стежити за дисципліною співробітників та клієнтів (замовників).

Ключові переваги Okdesk:

- облік і автоматизація рішення клієнтських замовлень;
- мобільний додаток для заявників і виконавців (iOS і Android);
- telegram бот для заявників;
- єдине Help Desk + CRM рішення (база клієнтів, контактних осіб);
- геолокація і замовлення на карті (field service management);

- сервісна специфіка (календарне планування, прайс-лист робіт, облік обладнання, договорів, сервісних періодів, абонентських платежів, об'єкти обслуговування;
- друковані форми;
- «Клієнтський портал» для реєстрації замовлень від клієнтів;
- вбудовані дашборда по KPI і трирівнева панель звітів для контролю SLA. Готові інтеграції з BI системами і шаблони звітів в MS Power BI;
- галузеві інтеграції;
- зручний і простий інтерфейс;
- швидка технічна підтримка.

Платформа знадобиться компаніям, обслуговуючим бізнес-центри, ресторани і підприємства. З її допомогою клієнти і підрядники можуть залишити повідомлення про неполадки, співробітники - прийняти їх та вирішити, а керівники - бути в курсі поточних процесів і отримати доступ до аналітики про ефективність персоналу.

2. Система автоматизованого обліку замовлень РемОнлайн.

Система РемОнлайн - це програма для різних типів бізнесу послуг: майстерень, автосервісів, ательє, ремонту електроніки, друкарень, хімчисток, прокату і оренди, ремонту та заправки картриджів, управління автомобільної СТО. Вікно статистики системи РемОнлайн наведено в Рис. 1.4.

Рис. 1.4. Вікно статистики системи РемОнлайн.

Доступна єдина база замовлень і клієнтів, фінансовий облік, магазин, склад, SMS повідомлення клієнтам, більше 200 операторів IP-телефонії та інструменти аналітики для керівника.

Можливості сервісу:

- Доступність з будь-якого пристрою. Для роботи сервісу необхідний лише браузер.
- Управління замовленнями. Замовлення виводяться в єдину таблицю і мають статуси (в ремонті, готовий, виданий, в очікуванні).
- Доступний пошук замовлень за різними параметрами: номер, найменування, ім'я або телефон клієнта.
- Складський облік всього обсягу товарів і запчастин майстерні: застосування, списання, перенесення запчастин між складами майстернями, у кожного товару доступні категорія і артикул.
- Модуль магазину для реєстрації роздрібних продажів. Після завершення угоди є опція друку «Товарного чека» для звітності.
- Створення звітів з продажу та накопиченим залишкам.
- Каса - контроль потоків грошових коштів бізнесу. Автореєстрація всіх операцій по прийому і видачі грошей, всі дані можна вивести в звітах.
- Розрахунок заробітної плати. Якщо застосовується відрядна оплата - вказівка відраховувати співробітнику відсотка від вартості робіт. Звіт «по майстрам» допоможе розрахувати зарплату за будь-який час.
- Друк квитанцій, актів, чеків та інших документів з браузера.
- Автоматичне резервне копіювання даних.

На даний момент цей сервіс дуже знаменитий, тому ним користується більшість сервісних центрів самої різної спеціалізації. Але не дивлячись на таку популярність цього сервісу, він має низку проблем, помилок, та мінусів у використанні в порівнянні з Okdesk, наприклад:

- недостатня функціональність;
- немає взаєморозрахунків;
- немає вбудованої комунікації з клієнтом;
- крім СМС-повідомлень, немає продуманої системи завдань і нагадувань;

- неможливо відредагувати замовлення;

3. Онлайн-система обслуговування HubEx.

Система HubEx - це онлайн-система для автоматизації процесів сервісного і гарантійного обслуговування на всіх етапах. Єдина платформа для всіх бізнес-процесів компанії (Smart desk + FSM + BPM). Вікно замовлень системи HubEx наведено на Рис. 1.4.

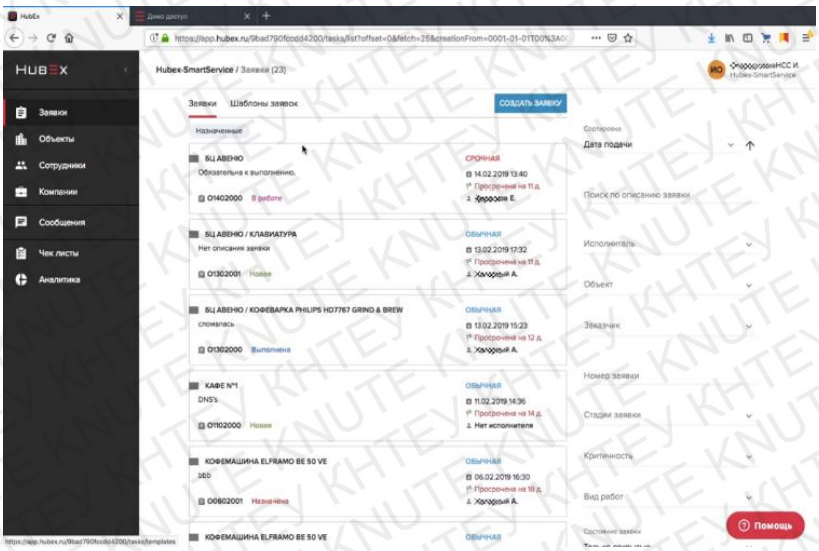

Рис. 1.4. Вікно замовлень системи HubEx.

HubEx дозволяє налаштувати процес диспетчеризації замовлення до виконавця, систему контролю переміщень виконавців, стандартизувати процеси виконання робіт та перейти на електронну звітність по актам виконаних робіт. Платформа управління мобільними співробітниками і автоматизації сервісного обслуговування: Управління замовленнями, GPS-контроль мобільного персоналу, автоматизація диспетчерської, електронні паспорти обладнання, управління сервісним обслуговуванням в єдиній системі

Основні можливості системи:

- керування замовленнями (створення, редагування, видалення, пошук);
- налаштування життєвого циклу замовлення (стадії замовлення і переходи між ними);
- подача замовлень скануванням QR-коду;
- додавання файлів і документів до замовлення;

- обмін повідомленнями по замовленню;
- налаштування push-повідомлень про зміни замовлення і нові повідомлення;
- додавання фотографій і файлів до замовлення;
- геолокація мобільних співробітників;
- електронний паспорт обладнання;
- довідник об'єктів / обладнання;
- довідник видів робіт;
- BI-система, що відображає основні показники роботи сервісної служби;
- API для інтеграції з зовнішніми системами;
- стандартна підтримка і оновлення на нові версії;
- чек-листи для виконавців;
- автоматичне створення планових замовлень;
- узгодження умов виконання замовлення з замовником в системі;
- управління доступом до файлів в заявці;
- можливість подати замовлення по QR-коду незареєстрованому користувачеві;
- налаштування графіка робіт співробітників;
- налаштування типів замовлень;
- відстеження завдань;
- інтеграція зі складськими системами;
- облік матеріалів за замовленнями;
- SMS повідомлення клієнтів про стан замовлення.

Як ви можете помітили, цей сервіс має розширений функціонал та доволі приємний та інтуїтивно зрозумілий інтерфейс, що не заважає цьому сервісу, на відміну від попереднього «РемОнлайн» працювати якісно, без помилок та бути надійним сервісом.

Провівши аналіз та дослідивши цей сервіс, помилок не було виявлено, єдиний, замічений мінус, на даний момент – це поки що відсутність API з 1С.

1.3. Характеристика та вибір програмних засобів розробки комп'ютерної систем управління діяльністю сервісного центру.

Для розробки комп'ютерної системи керування діяльністю сервісного центру я буду використовувати таке програмне забезпечення як AnyLogic та мову програмування C++.

1. Програмне забезпечення AnyLogic.

Програма AnyLogic – це лідируючий інструмент імітаційного моделювання для бізнесу. Він допомагає аналітикам, інженерам і керівникам з різних галузей одержувати детальне уявлення про бізнес-системах і процесах і оптимізувати їх. Використовується в 40% компаній з рейтингу Fortune 100.

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

Також програмне забезпечення AnyLogic – має єдине ПО для агентного моделювання професійного класу:

- Завдяки Агентному підходу стало можливо застосовувати імітаційне моделювання в областях, де раніше це було недоступно: в маркетингу, соціальних процесах, епідеміології.
- Агентне моделювання дозволяє використовувати big data, щоб наповнювати моделі вхідними даними з реального світу. Використовуйте бази даних організації, щоб відбити в моделі поведінку споживачів, особисті навички персоналу, розкладу, час виконання бізнес-процесів і медичну статистику.

2. Мова програмування C++.

Мова програмування С++ - це компільована строго типізована мова програмування загального призначення. Підтримує різні парадигми програмування: процедурну, узагальнену, функціональну; найбільшу увагу приділено підтримці об'єктно-орієнтованого програмування.

Нововведеннями С ++ в порівнянні з С є:

- підтримка об'єктно-орієнтованого програмування через класи. С ++ надає всі чотири можливості ООП - абстракцію, інкапсуляцію, успадкування (в тому числі і множинне) і поліморфізм;
- підтримка узагальненого програмування через шаблони функцій і класів;
- стандартна бібліотека С ++ складається зі стандартної бібліотеки С (з деякими модифікаціями) і бібліотеки шаблонів (Standard Template Library, STL), яка надає широкий набір узагальнених контейнерів і алгоритмів;
- додаткові типи даних;
- обробка виключень;
- віртуальні функції;
- простору імен;
- вбудовуються (inline) функції;
- перевантаження (overloading) операторів;
- перевантаження імен функцій;
- посилання і оператори управління вільно розподіленою пам'яттю.

РОЗДІЛ 2.
ПРОЕКТУВАННЯ СИСТЕМИ АВТОМАТИЗОВАНОГО ОБЛІКУ ЗАМОВЛЕНЬ У СЕРВІСНОМУ ЦЕНТРІ

2.1. Постановка задачі на моделювання ведення замовлень та обліку їх у сервісному центрі.

Задачею дипломної роботи є розробка імітаційної моделі ведення замовлень та програмної системи обліку замовлень.

Для ведення замовлень необхідно дослідити залежність кількості виконаних замовлень всіх типів в заданому інтервалі (T_n) від ймовірності їх надходження: p_1, p_2, p_3, … p_n (задачі визначено $_n$ типів замовлень). Результати моделювання необхідно отримати з точністю до 0.01 і ймовірністю 0.95. проаналізувати завантаженість кожної групи.

До сервісного центру надходять замовлення $_n$ типів с ймовірністю p_1-p_n. Кожен тип замовлення вимагає одного із декількох видів ремонтів. Замовлення надходять до майстрів, для кожного майстра є свій вид роботи. Прийом і розподіл замовлень відбувається за участі диспетчерів.

Реалізація роботи буде здійснена за допомогою спеціального програмного забезпечення призначеного для створення імітаційних моделей - а саме AnyLogic, та мови програмування для написання коду – а саме C++.

Головними цілями дипломної роботи у розробці ведення та обліку замовлень на діагностику та ремонт у сервісному центрі є:

1. Побудова імітаційної моделі яка дасть змогу побачити та проаналізувати те, як ведення замовлень буде працювати в реальному часі.
2. Написати програму для обліку замовлень на ремонт. Кожне замовлення містить:
    1. Тип поломки.
    2. Номер замовлення.
    3. Прізвище та ініціали клієнта.
    4. Бажану дату виконання замовлення.

3. Програма повинна забезпечувати вибір за допомогою меню і виконання однієї з наступних функцій:

1. Додавання замовлень в список.
2. Видалення замовлень.
3. Висновок замовлень по заданому номеру замовлення і даті отримання замовлення.
4. Висновок всіх замовлень.

Для зберігання даних використовувати клас "колекція ключ-значення" в якості ключа використовувати "тип поломки". Передбачити збереження всіх даних при виході в файл і відновлення при повторному запуску програми.

Щоб реалізувати додання замовлень, огляд всіх створених замовлень керівником, та підключення бібліотек для роботи програми необхідно розробити комп'ютерну програму на С++, де вхідними даними будуть:

- "Додання нового замовлення:";
- "Вкажіть тип поломки - ";
- "Опис технічного пристрою - ";
- "Вкажіть бажану дату отримання замовлення (приклад: 22.05.2021) - ";
- "1. Додання нового замовлення ";
- "2. Видалити дані з інформаційної системи";
- "3. Висновок відомостей по поломці з запитаним номером";
- "4. Висновок відомостей по всіх поломках";
- "0. Вихід з програми ";
- "Вводьте - ";

А на виході отримаємо такі дані:

- "Замовлення сформоване.";
- " Висновок відомостей по всіх поломках:";
- " Висновок замовлень по фільтру ".

2.2. Математична модель розрахунку показників ведення замовлень у сервісному центрі.

Математична модель визначається замовленнями, законами, часом зберігання, надходженнями. В даній моделі ми використовуємо «Експоненціальний закон», формула наведена нижче (2.1).

$$f(t) = \begin{cases} 0, & t < 0 \\ \lambda e^{-\lambda t}, & t \geq 0 \end{cases}, \quad (2.1)$$

де λ — інтенсивність подій, тобто кількість подій в одиницю часу, експоненціальний закон розподілу має тільки один параметр λ. Таким чином, якщо випадкова величина X має показниковий закон розподілу з параметром λ>0, це можна записати у вигляді $X \in E_\lambda$.

Для моделювання ведення замовлень використовуються агенти, майстри.

В агенті «Джерело заявок» в діях при виході було налаштовано потік замовлень що в результаті будуть надходити до різних блоків, таким чином встановлюємо час між прибуттям exponential$(1/(Tp/n))$, та дію при виході:

agent.типЗ=uniform();

agent.видР=uniform();

Для розподілення замовлень, використовується номер виходу uniform_discr(1,4), а також такі чотири дії при виході:

1. Перша дія при виході: agent.типЗ=1; ЗаявкаТип1++; ЗаявкаТип++.
2. Друга дія при виході: agent.типЗ=2; ЗаявкаТип2++; ЗаявкаТип++.
3. Третя дія при виході: agent.типЗ=3; ЗаявкаТип3++; ЗаявкаТип++.
4. Четверта дія при виході: agent.типЗ=4; ЗаявкаТип4++; ЗаявкаТип++.

Після розподілу замовлення проходять чергу та направляються до диспетчера, диспетчер затримує агентів. Час затримки та дія при виході:

1. normal(To1,T1) – час затримки;
2. коефВикорДисп=Диспетчер.statsUtilization.mean() – дія при виході.

У розподілі замовлень за видами робіт використовуються чотири умови, а саме:

agent.типЗ==1;

agent.типЗ==2;

agent.типЗ==3;

agent.типЗ==4.

У властивості для розподілу робіт та час, також використовуються наведені нижче умови та дії при виході:

1. agent.видР<=p11;

   agent.видР<=p12;

   agent.видР<=p13;

   randomTrue(0.5).

2. agent.видР=1;

   agent.часР=exponential(1/T11);

   agent.видР=2;

   agent.часР=exponential(1/T12);

   agent.видР=3;

   agent.часР=exponential(1/T13);

Для розподілу та порівняння зайнятості кожного майстра та направлення до його, якщо він звільнився, використовується єдина умова:

(очМайстрів1.size()==0)&&;

(майстри1.size()<кількМайстрів1)&&;

(майстри3.size()!=0).

Для створення моделі ведення замовлень, було побудовано схему алгоритму генерування замовлень: 1-м майстром замовлення черги, а 2-м майстром отримання замовлення та їх обробка. Схема алгоритму наведена на Рис. 2.1.

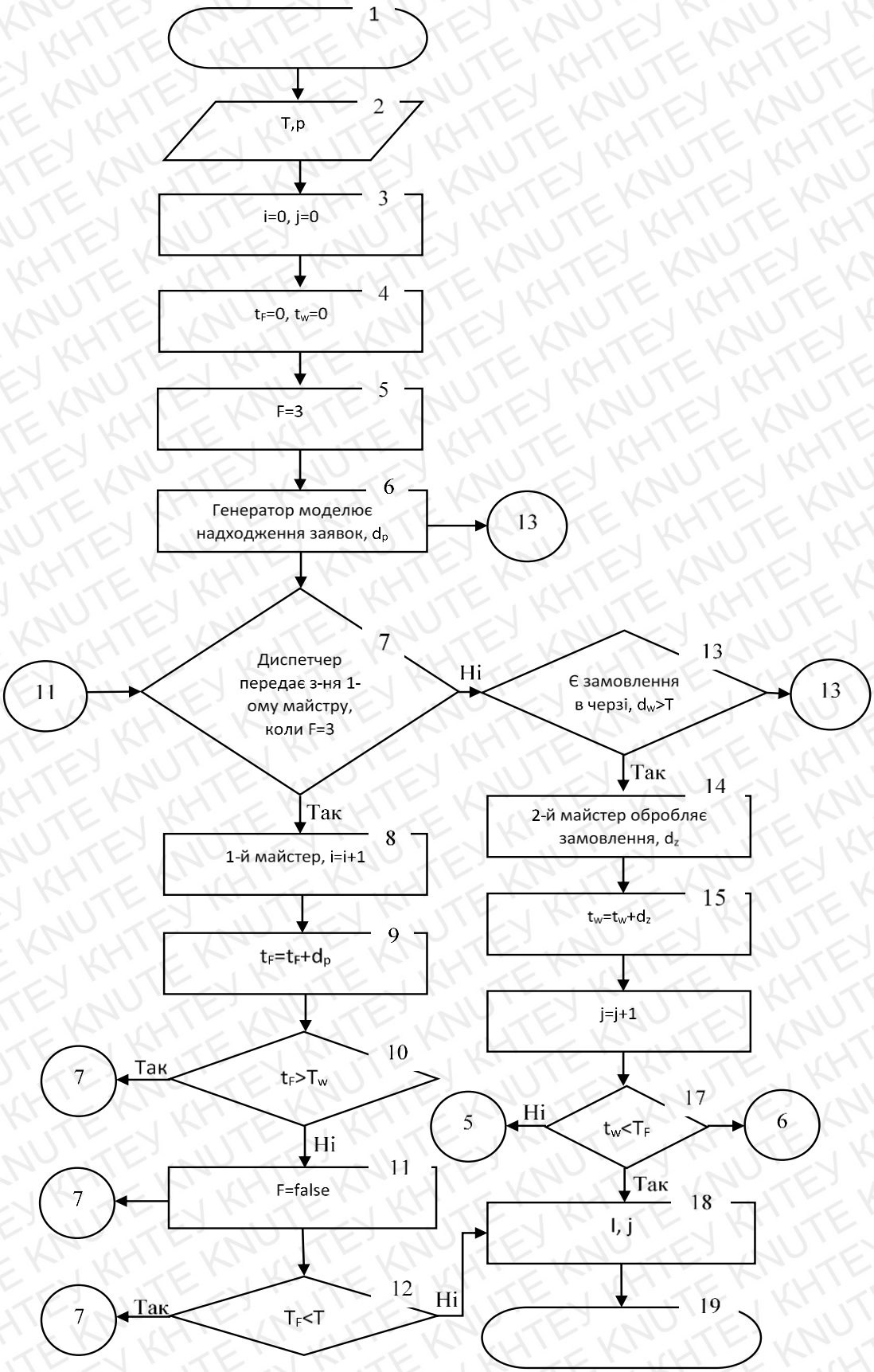

Рис. 2.1. Схема алгоритму генерування замовлень та їх обробка.

Позначення схеми алгоритму:

- T – період моделювання;
- p – ймовірність надходження замовлення;
- i – кількість замовлень надійшло до черги;
- j – кількість замовлень оброблено з черги;
- t_f – поточний час надходження замовлення до черги;
- t_w – поточний час обробки замовлень з черги;
- d_p – проміжок часу необхідний для формування поточного замовлення;
- d_z – час очікування поточного замовлення;
- z – час необхідний для обробки замовлення.

2.3. Імітаційна модель ведення замовлень у сервісному центрі.

Імітаційне моделювання - метод дослідження, при якому вивчається система та замінюється моделлю, з достатньою точністю описує реальну систему, з якої проводяться експерименти з метою отримання інформації про цю системі. Таку модель можна «програти» в часі, як для одного випробування, так і заданого їх безлічі.

Середовище моделювання включає в себе: низько рівневі конструкції моделювання (змінні, рівняння, параметри, події тощо), форми подання (лінії, квадрати, овали і т.д), елементи аналізу (бази даних, гістограми, графіки), стандартні картинки і форми експериментів.

Графічне середовище моделювання AnyLogic включає в себе наступні елементи :

- Stock & Flow Diagrams (діаграма потоків і накопичувачів) застосовується при розробці моделей, використовуючи метод системної динаміки.
- Statecharts (карти станів) в основному використовуються в агентних моделях для визначення поведінки агентів. Вони також часто використовуються в дискретно-подієвому моделюванні, наприклад для симуляції машинних збоїв.

- Action charts (блок-схеми) використовуються для побудови алгоритмів. Застосовується в дискретно-подієвому моделюванні (маршрутизація дзвінків) і Агентному моделюванні (для логіки рішень агента).
- Process flowcharts (процесні діаграми) - основна конструкція, яка використовується для визначення процесів в дискретно-подієвому моделюванні.

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

В цій дипломній роботі для дослідження автоматизації обліку замовлень в сервісному центрі я буду використовувати «Агентний вид імітаційного моделювання».

Агентний вид моделювання - це напрямок в імітаційному моделюванні, яке використовується для дослідження децентралізованих систем, динаміка функціонування яких визначається не глобальними правилами і законами, а навпаки, коли ці глобальні правила і закони є результатом індивідуальної активності членів групи.

Також агентне моделювання дозволяє використовувати big data, щоб наповнювати моделі вхідними даними з реального світу. Використовуйте бази даних організації, щоб відбити в моделі поведінку споживачів, особисті навички персоналу, розкладу, час виконання бізнес-процесів і медичну статистику.

Модель відтворює повне функціювання автоматизації обліку, та його аспекти. Також зможемо контролювати навантаження сервісного центру у визначений час і дату, та інтенсивність прибуття замовлень.

2.4. Програмна система обліку замовлень у сервісному центрі.

Програмна система – це група інтегрованих програмних засобів, які підтримують певний діловий процес споживача (або його частину) і спільно використовують бази даних. Прикладом програмної системи може бути

група програмних застосувань, що складають автоматизоване робоче місце фахівця в предметній області.

Для створення програмної системи обліку замовлень у сервісному центрі буде використовуватися концептуальне проектування.

Концептуальне проектування полягає в описі і синтезі інформаційних вимог користувачів у початковий проект БД. Вихідними даними буде сукупність даних про замовника та опис причини звернення при класичному підході. Результатом цього етапу є високо-рівневі подання (у вигляді системи таблиць БД) інформаційних вимог користувачів на основі різних підходів. Спочатку вибирається модель БД. Потім створюється структура БД, яка заповнюється даними за допомогою систем меню, екранних форм.

У кожну створювану модель даних входять наступні компоненти:

- типи сутностей;
- типи зв’язків;
- атрибути;
- домени атрибутів;
- потенційні ключі;
- первинні ключі.

Для зберігання даних використовувати клас "колекція ключ-значення" в якості ключа використовувати "Тип поломки".

Програма повинна забезпечувати вибір за допомогою меню і виконання однієї з наступних функцій:

- введення даних в інформаційну систему;
- висновок відомостей по всім замовленням;
- висновок відомостей по замовленню з запитаним номером.

База даних "колекція ключ-значення" – це простий магазин ключів/значень. Для його написання використовується C++, та використовує лише стандартний рядок та функції вводу-виводу.

Він зберігає ключі та значення фіксованої довжини у простому форматі файлу на основі хеш-таблиць фіксованого розміру. Якщо відбувається зіткнення хешу, до бази даних додається нова "сторінка" хеш-таблиці. Формат - лише додаток.

Розмір хеш-таблиці - це параметр компромісу пробіл/швидкість. Більші хеш-таблиці зменшать зіткнення і трохи прискорять процес, за рахунок пам'яті та дискового простору. Хороший розмір зазвичай становить приблизно 1/2 середньої кількості записів.

РОЗДІЛ 3.
РОЗРОБКА ІМІТАЦІЙНОЇ МОДЕЛІ ВЕДЕННЯ ТА ОБЛІКУ ЗАМОВЛЕНЬ ТА ЇЇ ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1. Об’єкти та елементи моделі ведення замовлень в середовищі системи AnyLogic.

У робочому просторі вікна AnyLogic за замовчуванням відображаються наступні основні компоненти:

- Графічний редактор - Місце для візуального редагування діаграми типу агентів (або експерименту).
- Панель «Проекти» - Забезпечує легку навігацію по моделям, відкритим в поточний момент часу. Кожна модель представлена в панелі у вигляді ієрархічного дерева.
- Панель «Палітра» - Містить список всіх елементів, які можуть бути додані на діаграму агента (експерименту). Елементи логічно розбиті за категоріями на кілька закладок (палітр). А також відображаються і палітри бібліотек AnyLogic.
- Панель «Властивості» - Використовується для перегляду і зміни властивостей обраного в даний момент елемента (або елементів) моделі.
- Панель «Помилки» - Показує виявлені на етапі компіляції і побудови моделі помилки.

У побудові моделі ведення замовлень на діагностику та ремонт у сервісному центрі я використовував такі елементи:

1) Елемент «Параметр». Параметри зазвичай використовуються для завдання статичних характеристик агента. Ви можете задати різні значення параметрів для різних агентів одного і того ж типу, що потрібно в тих випадках, коли агенти мають однакову поведінку, але у них відрізняються деякі характеристики.
2) Елемент «Змінні». Змінні зазвичай використовуються для моделювання змінюються характеристик агента або для

зберігання результатів роботи моделі. AnyLogic підтримує два типи змінних - прості змінні і колекції.

3) Елемент «Агент». Під агентом в Агентному моделюванні розуміється елемент моделі, який може мати поведінку, пам'ять (історію), контакти і т.д. Агенти можуть моделювати людей, компанії, проекти, автомобілі, міста, тварин, кораблі, товари і т.д.
4) Блок «Source». Створює агентів. Зазвичай використовується в якості початкової точки потоку агентів.
5) Блок «SelectOutput5». Блок направляє входять агентів в один з п'яти вихідних портів в залежності від виконання заданих (детермінованих або заданих за допомогою ймовірностей) умов.
6) Блок «Queue». Блок Queue моделює чергу агентів, які очікують прийому блоками, наступними за даними в потокової діаграмі, або ж сховище агентів загального призначення.
7) Блок «Delay». Затримує агентів на заданий період часу. Час затримки обчислюється динамічно, може бути випадковим, залежатиме від поточного агента або від якихось інших умов.
8) Блок «SelectOutput». Блок направляє входять агентів в один з двох вихідних портів в залежності від виконання заданого (детермінованого або заданого за допомогою ймовірностей) умови. Умова може залежати як від агента, так і від якихось зовнішніх чинників. Поступив агент залишає блок SelectOutput в той же момент часу.
9) Блок «Sink». Знищує надійшли агентів. Зазвичай використовується в якості кінцевої точки потоку агентів.

3.2. Рішення задачі ведення замовлень у середовищі системи AnyLogic.

На першому етапі створення моделі було виділена область для розміщення параметрів з вихідними даними для моделі розробки. Перший етап створення моделі наведено в Рис. 3.1.

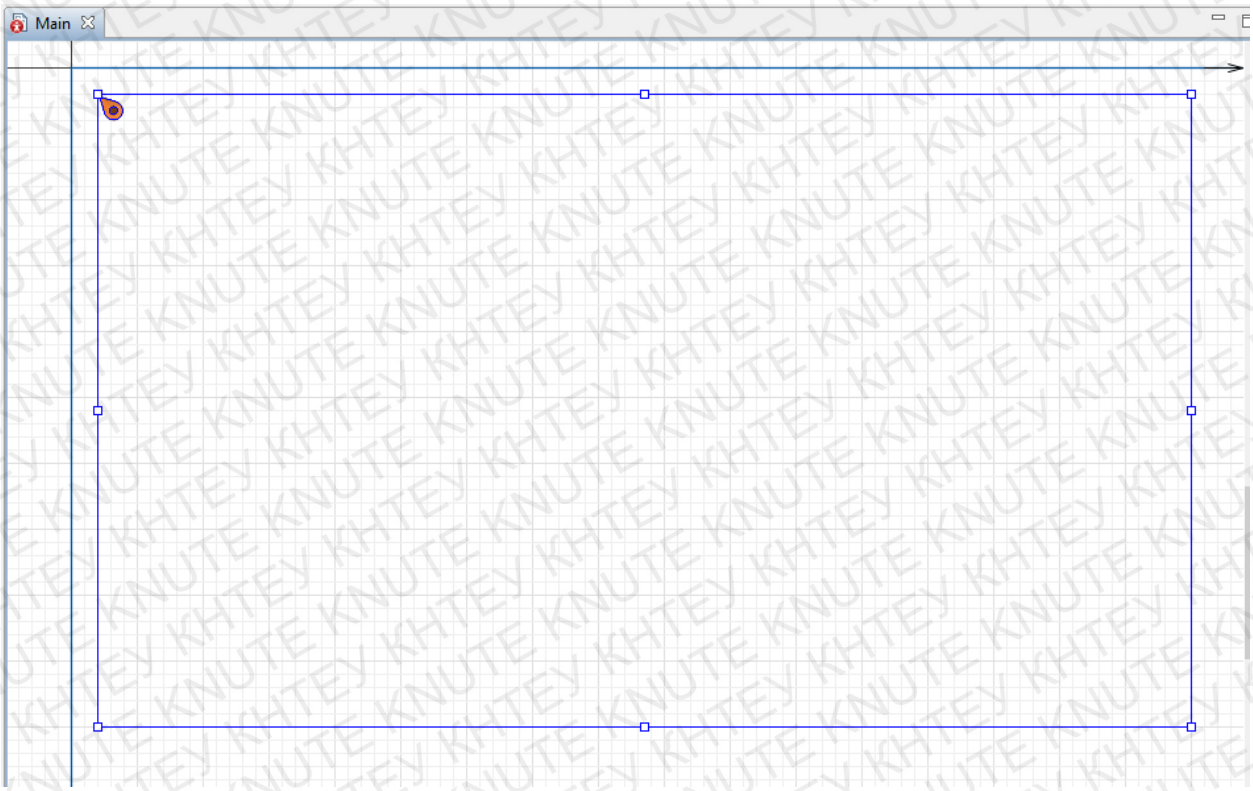

Рис 3.1. Виділення області для розміщення параметрів.

В наступному кроці були створені параметри з відповідними даними. Для початку була задана ймовірність надходження замовлень (параметр p1-pn) та інтервали часу між двома черговими надходженнями одного типу. За умовою було задано 4 типи майстри (n), для кожного були створені параметри та параметри для його робіт(p11…p43) . Також був доданий параметр з інтервалом часу між двома черговими надходженнями одного типу замовлення (Tn). Для кожного виду ремонту заданий окремий інтервал часу(T11…T43). Параметри T1 і To1 це інтервали часу обробки замовлень диспетчерами. Створення параметрів наведено в Рис. 3.2.

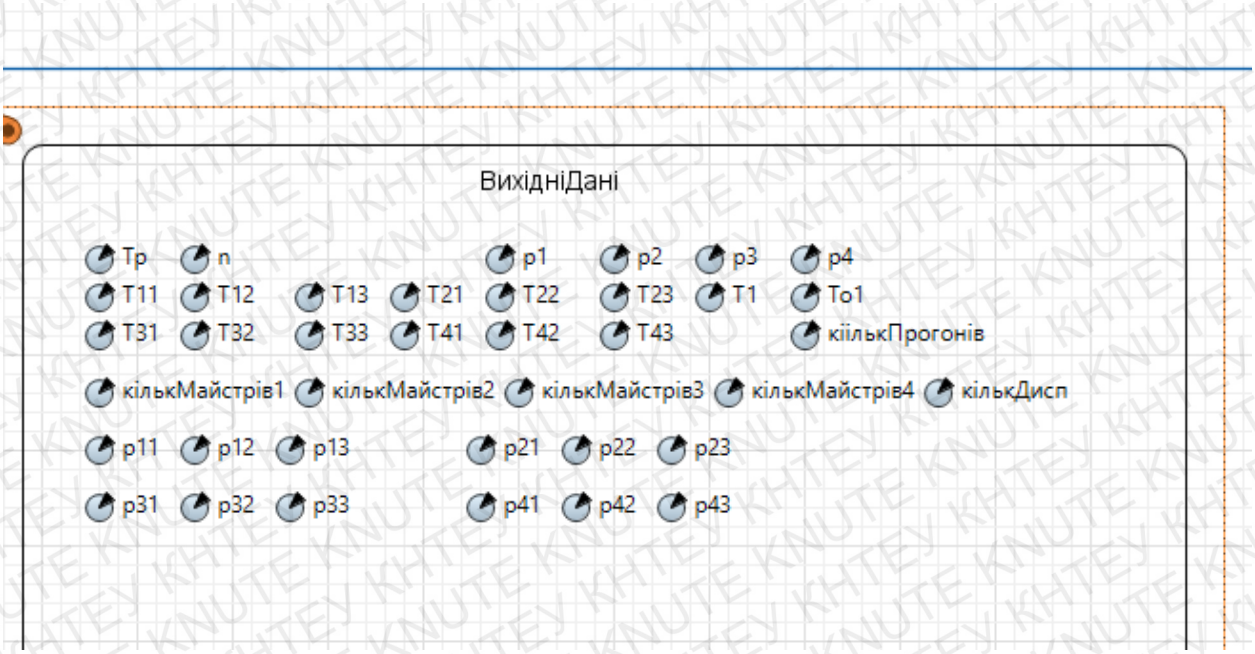

Рис. 3.2. Створення параметрів.

Для параметрів, які відповідають за персонал був встановлений тип (int), так як це цілі числа. В Рис. 3.3. наведено параметр, що відповідає за персонал.

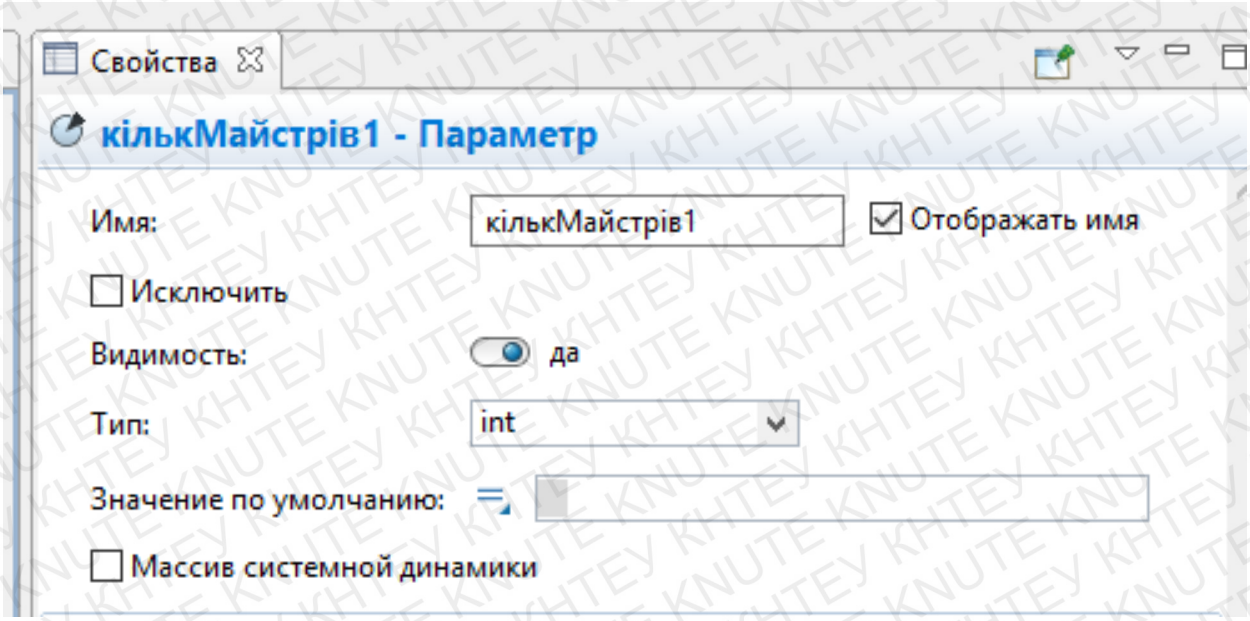

Рис. 3.3. Параметр відповідний за персонал.

Для інших параметрів був заданий тип (double). Параметри типу (double) наведено в Рис. 3.4.

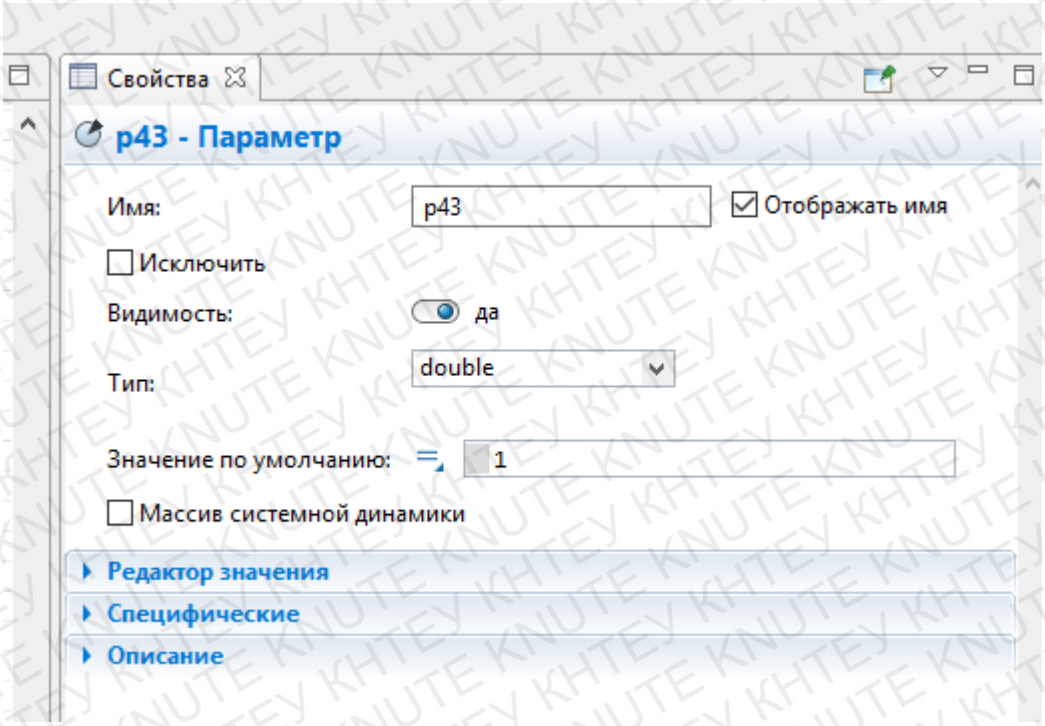

Рис. 3.4. Параметри типу (double).

В наступному кроці були додані змінні що допоможуть нам записати отримані результати для аналізу моделі. Це дозволить нам дізнатися кількість створених замовлень різних типів та навантаження майстрів. Додання змін зображено на Рис. 3.5.

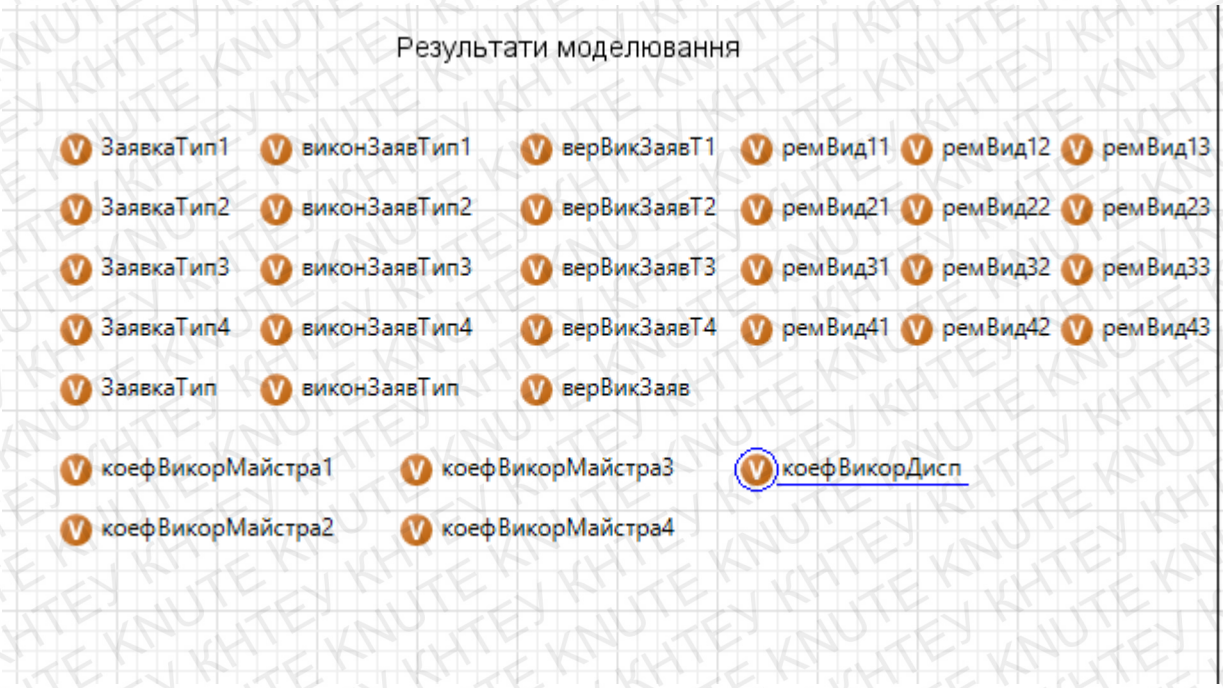

Рис. 3.5. Додання змін.

Далі було створено потік деталей та створено Java клас для створення та розподілу різних типів замовлень. У властивостях були задані параметри з типом даних double, тип (З) (тип замовлень), вид (Р) (вид робіт) та час (Р) (час робіт).

Для подачі агентів перетворюємо цей клас в тип агента, та додаємо до блоку Source. Створення Java класу та подача агента наведені нижче (Рис. 3.6. та Рис. 3.7.)

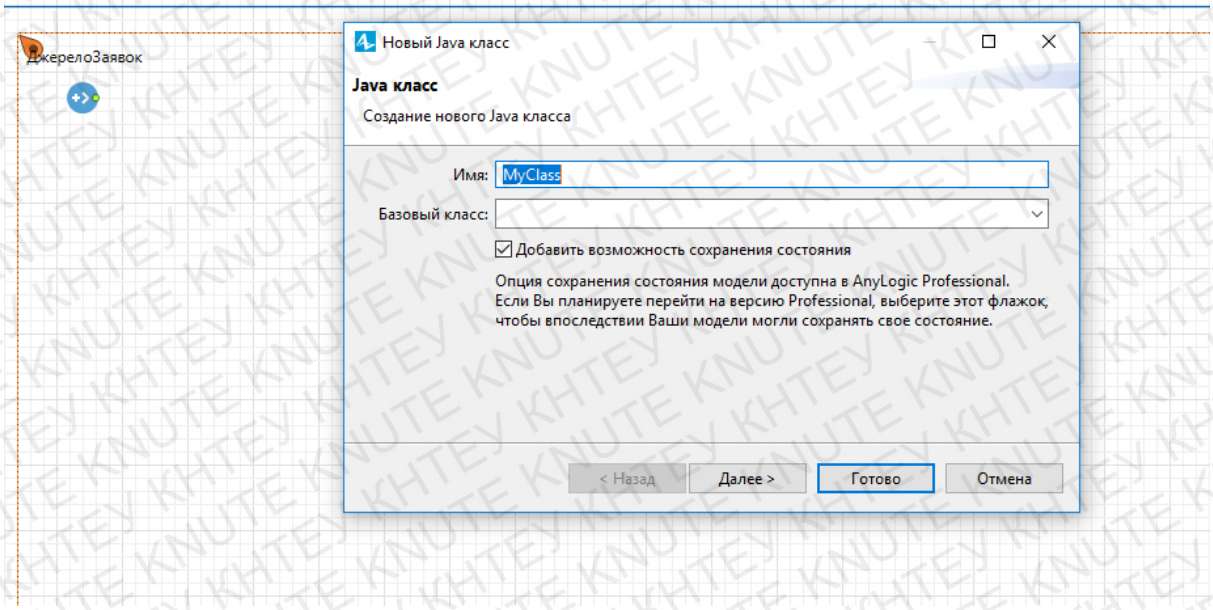

Рис. 3.6. Створення Java класу.

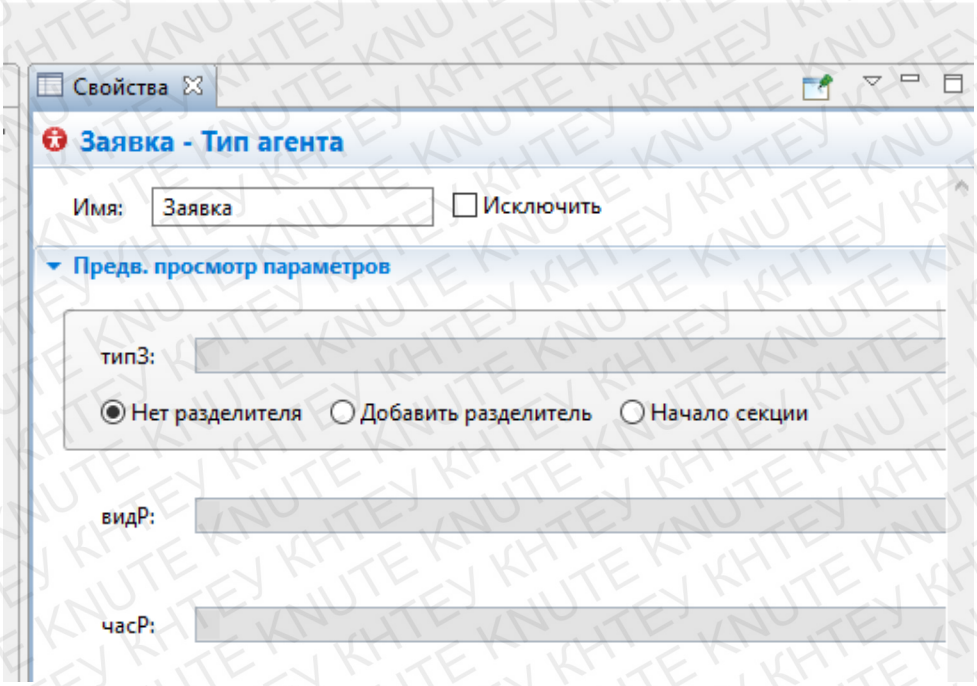

Рис. 3.7. Подача агента.

В діях при виході було налаштовано потік замовлень що в результаті будуть надходити до різних блоків. Налаштування потоку замовлень наведено нижче на Рис. 3.8.

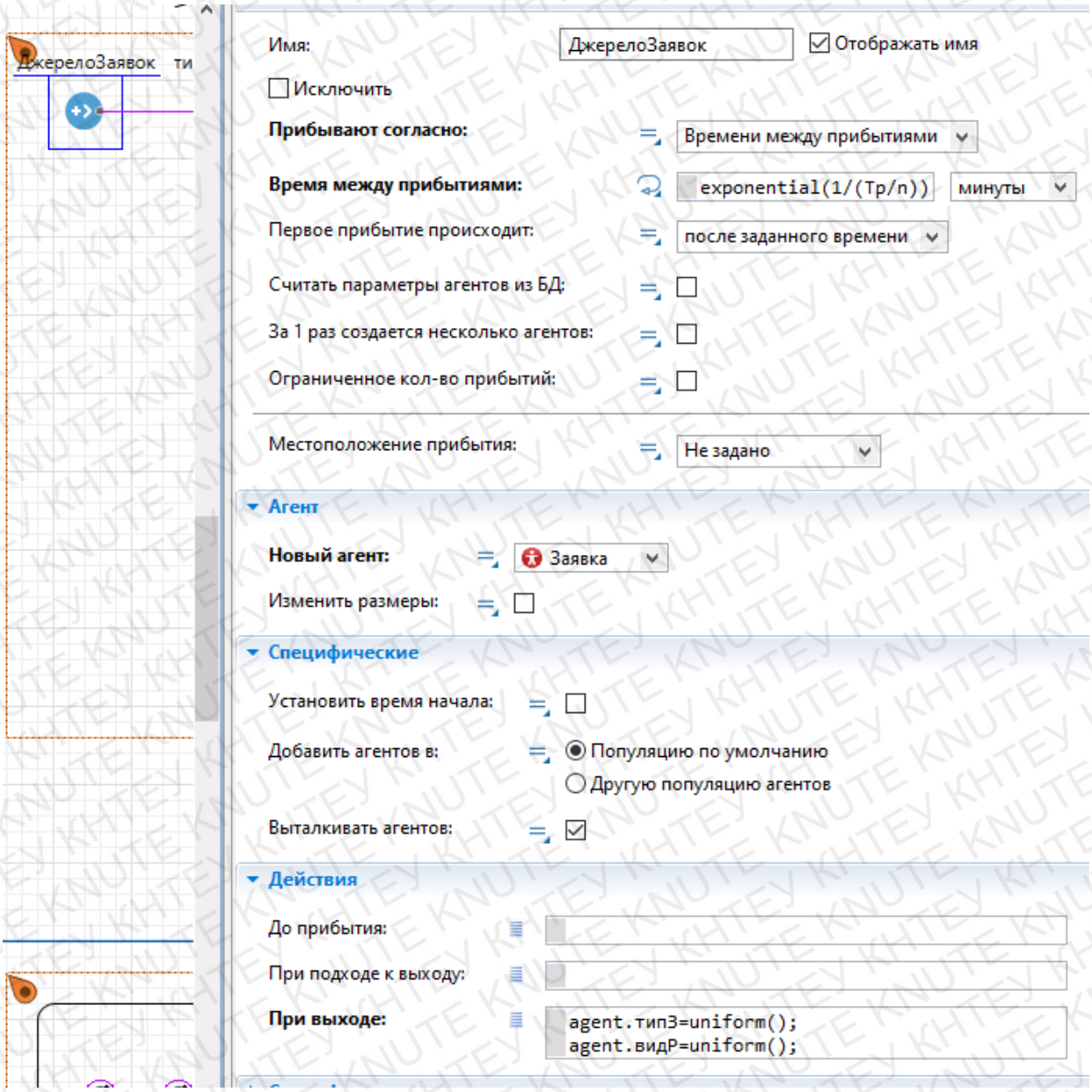

Рис. 3.8. Налаштування блоку «Джерело заявок».

На наступному кроці замовлення розділяються за допомогою блоку (SelectOutput5). Для кожного виходу вказана відповідна умова що відповідає типу замовлення. Розділення замовлень наведено нижче в Рис. 3.9.

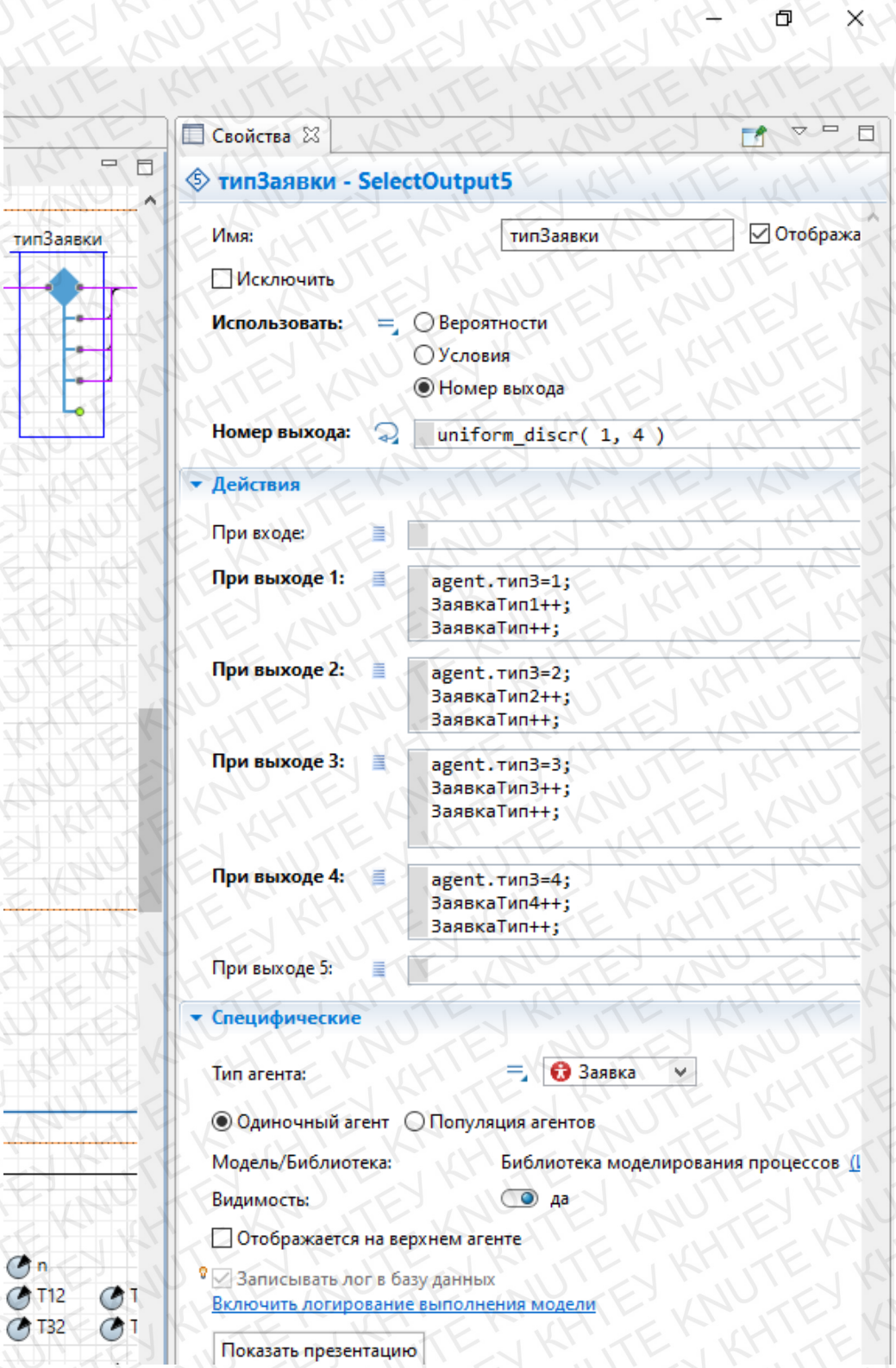

Рис. 3.9. Розділення замовлень за допомогою блоку (SelectOutput5).

Після розподілу замовлення проходять чергу та направляються до диспетчера (блок Delay). Диспетчер затримує агентів на заданий час відповідними параметрами. Властивості диспетчера зображені на Рис. 3.10.

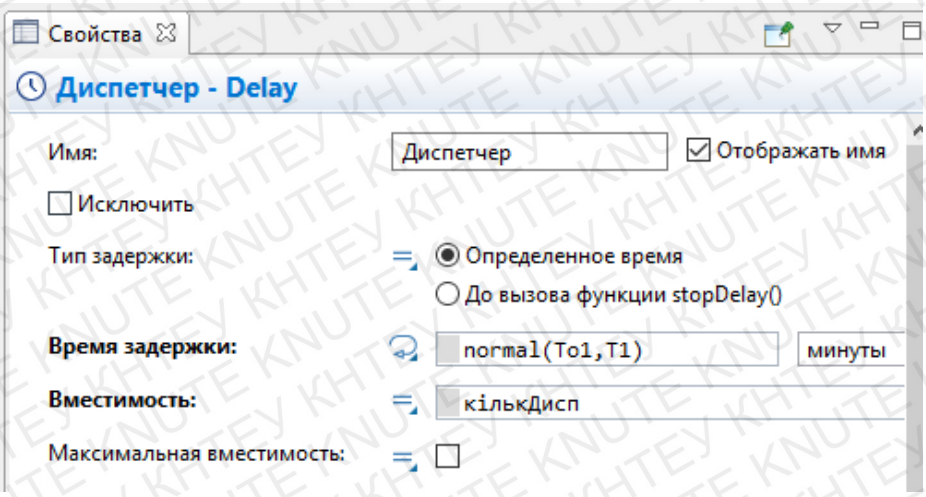

Рис. 3.10. Властивості диспетчера (блок Delay).

Агенти які проходять диспетчера роблять це з ймовірністю 0.98. Властивості диспетчера зображені на Рис. 3.11.

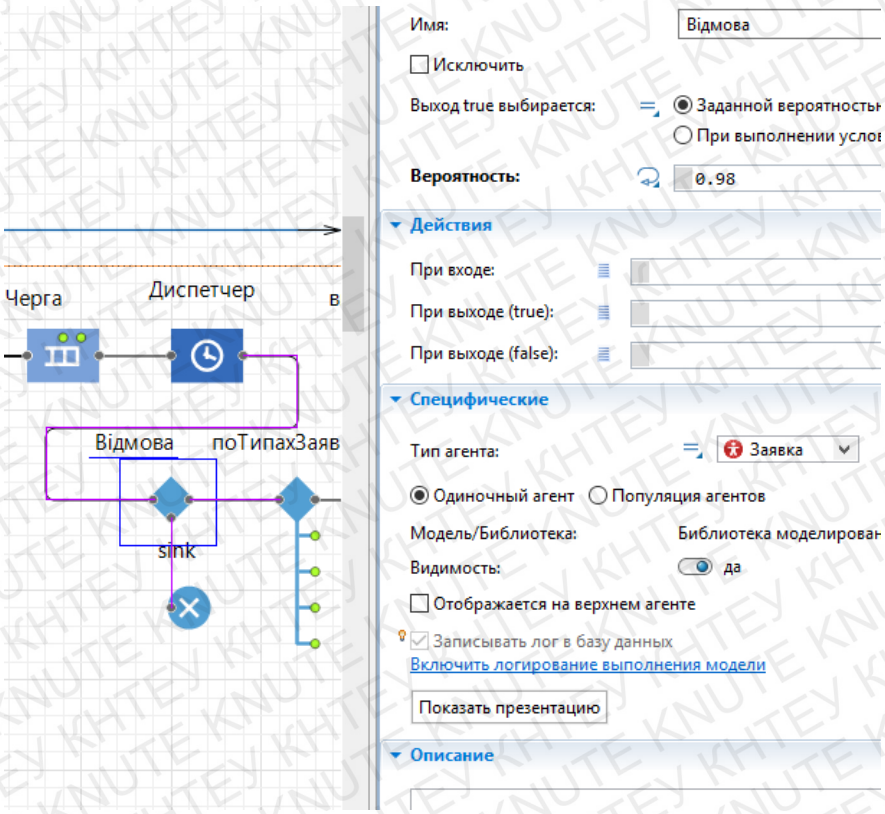

Рис. 3.11. Властивості диспетчера.

Далі йде ще один розподіл що безпосередньо розподіляє замовлення за видами ремонтних робіт. Розподіл замовлень за видами робіт наведено на Рис. 3.12.

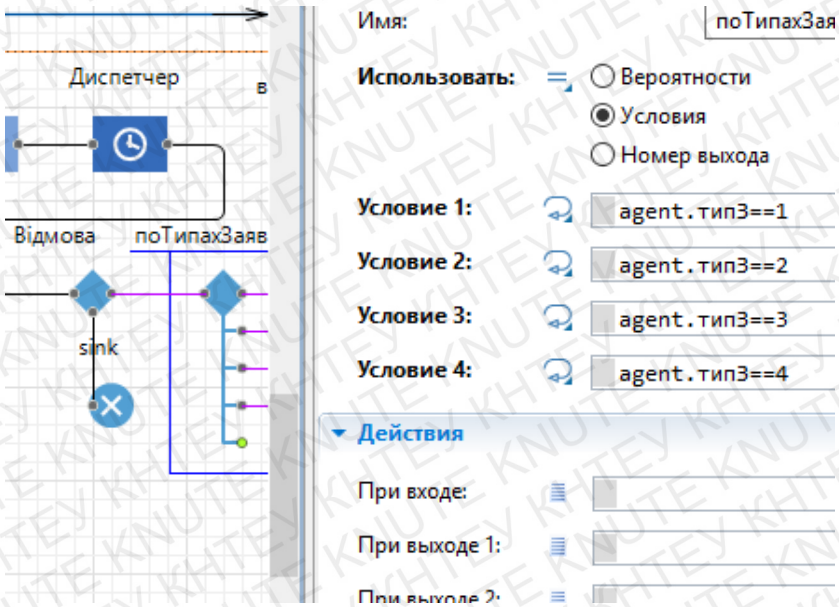

Рис. 3.12. Розподіл замовлень за видами робіт.

Для кожного виду замовлень є декілька типів ремонтних робіт, для цього задається відповідна умова. Також вказується час для кожного виду робіт, що відбувається за допомогою експоненціального розподілу. В Рис. 3.13. зображено властивості для розподілу робіт та вказаний час.

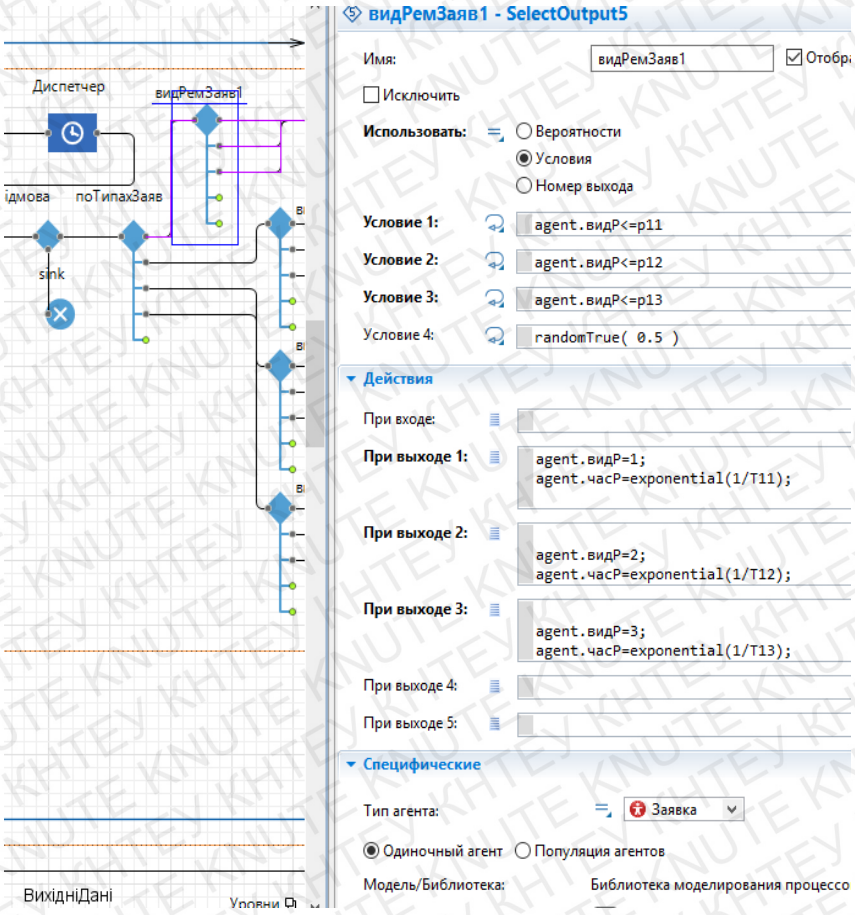

Рис. 3.13. Властивості для розподілу робіт та час.

Наступний розподіл порівнює зайнятість кожного майстра та направляє до його, якщо він звільнився. Розподіл та порівняння зайнятості наведено на Рис. 3.14.

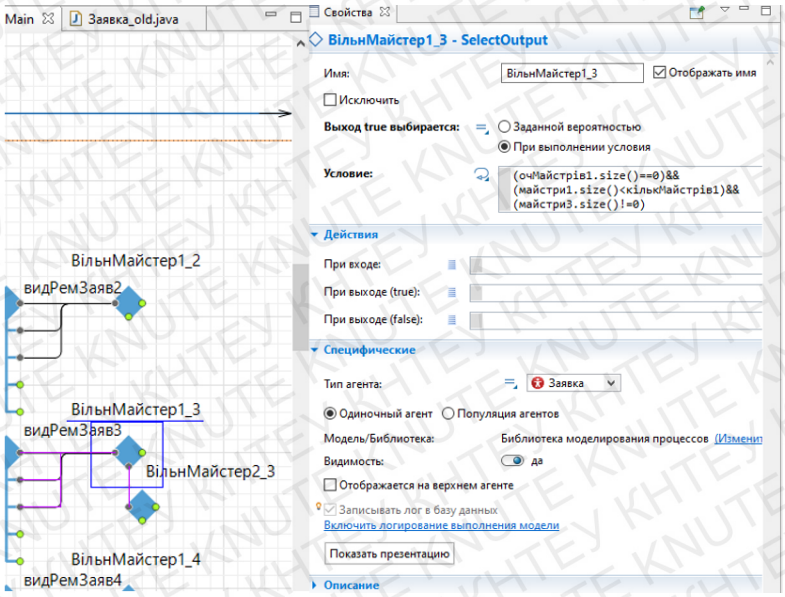

Рис. 3.14. Розподіл та порівняння зайнятості

Блоки (Delay) будуть імітувати роботу майстрів. Час затримки відповідає значенню час (Р). В діях умову при виході вказуємо для дослідження завантаженості кожного майстра. Властивість блоків (Delay) нижче на Рис. 3.15.

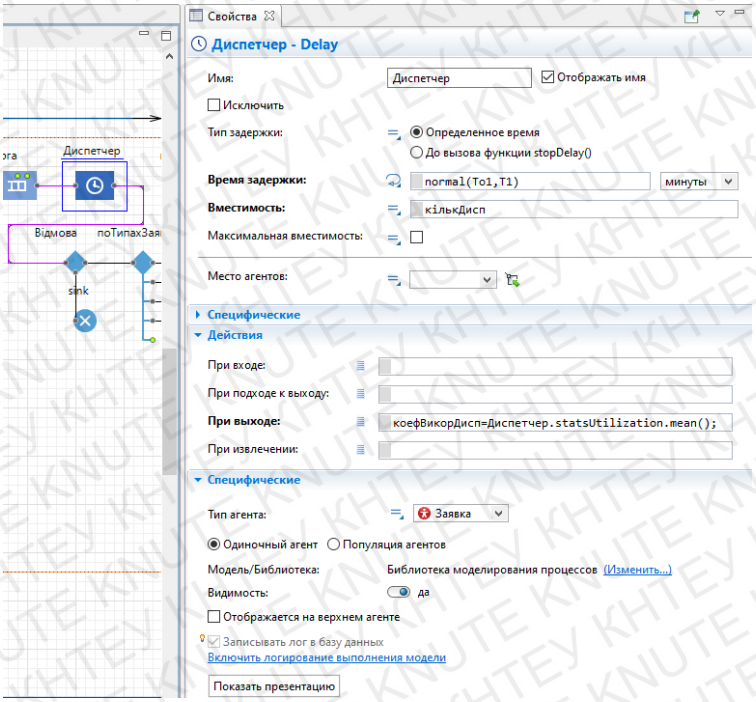

Рис. 3.15. Властивість блоків (Delay)

Далі іде розподіл виконаних замовлень. Він також відбувається як і попередні за типами замовлень. В діях при виході вказуємо умову для підрахунку виконаних замовлень. замовлення проходять верифікацію(виконані замовлення /тип замовлення). Розподіл виконаних замовлень та замовлень за видами робіт наведені нижче (Рис. 3.16. та Рис. 3.17.)

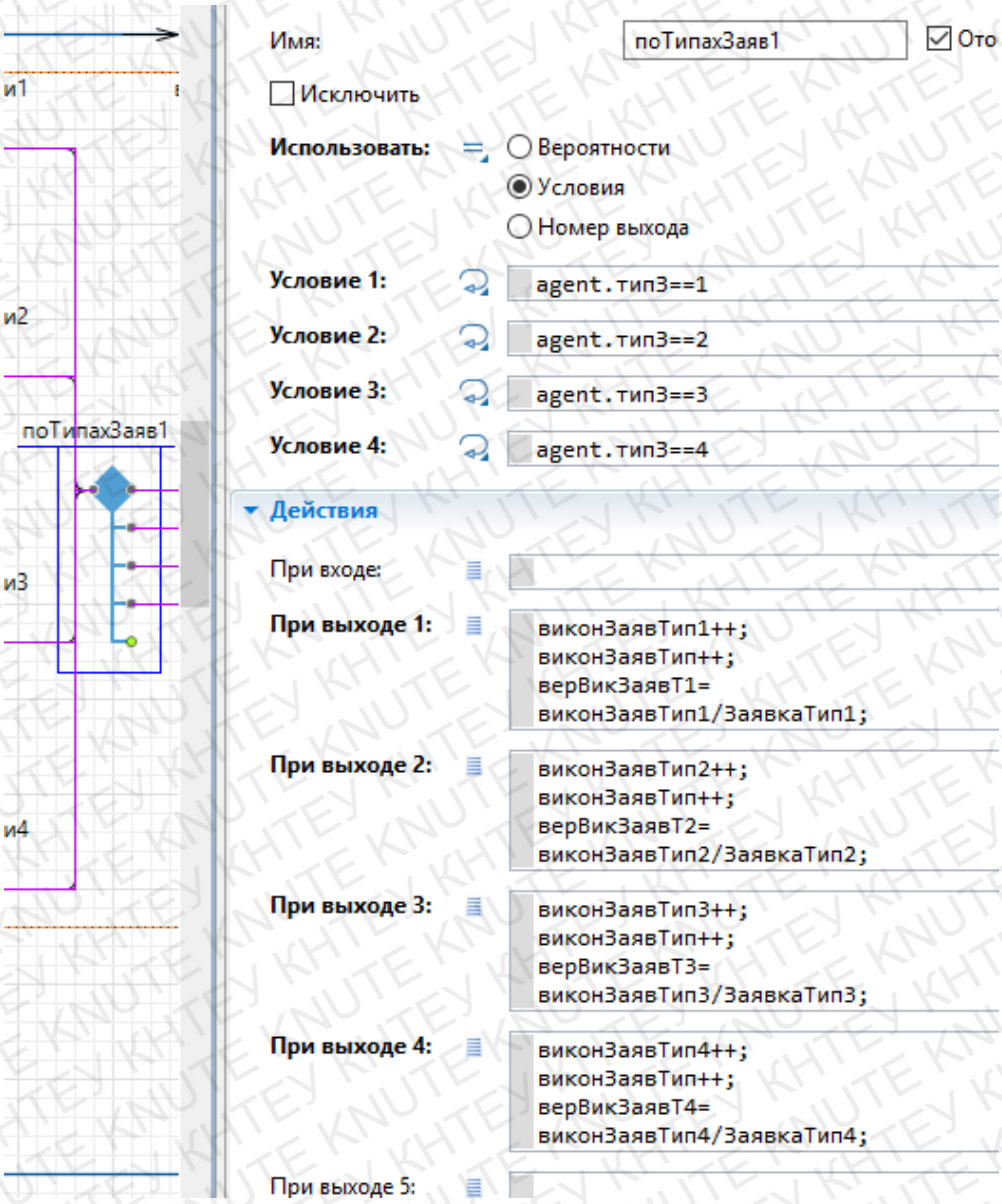

Рис. 3.16. Розподіл виконаних замовлень.

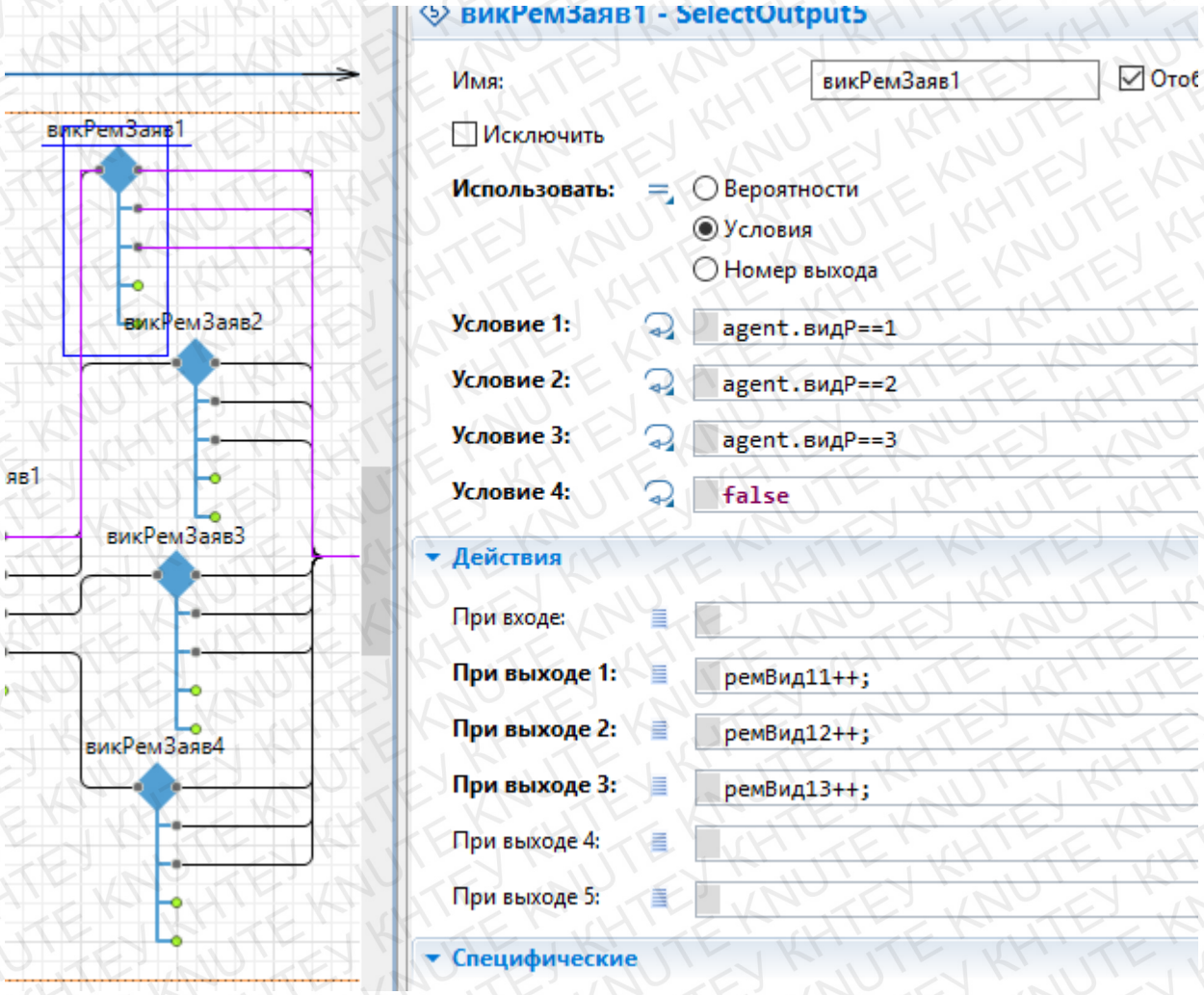

Рис. 3.17. Розподіл замовлень за видами робіт.

Для вивільнення замовлень використовується блок (Sink). Блок для вивільнення замовлень зображено нижче на Рис. 3.18.

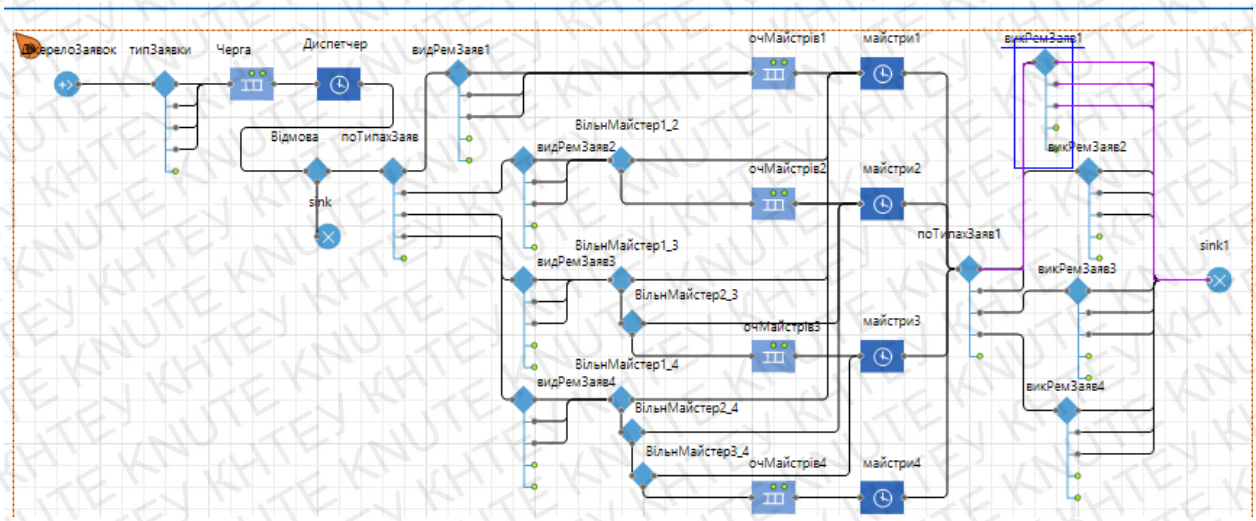

Рис. 3.18. Вивільнення замовлень блок (Sink).

У завершення виведемо кругову і стовпчикову діаграму, для спостереження та аналізу продуктивності ведення замовлень на діагностику та ремонт у сервісному центрі. Діаграма результатів зображена на Рис. 3.19.

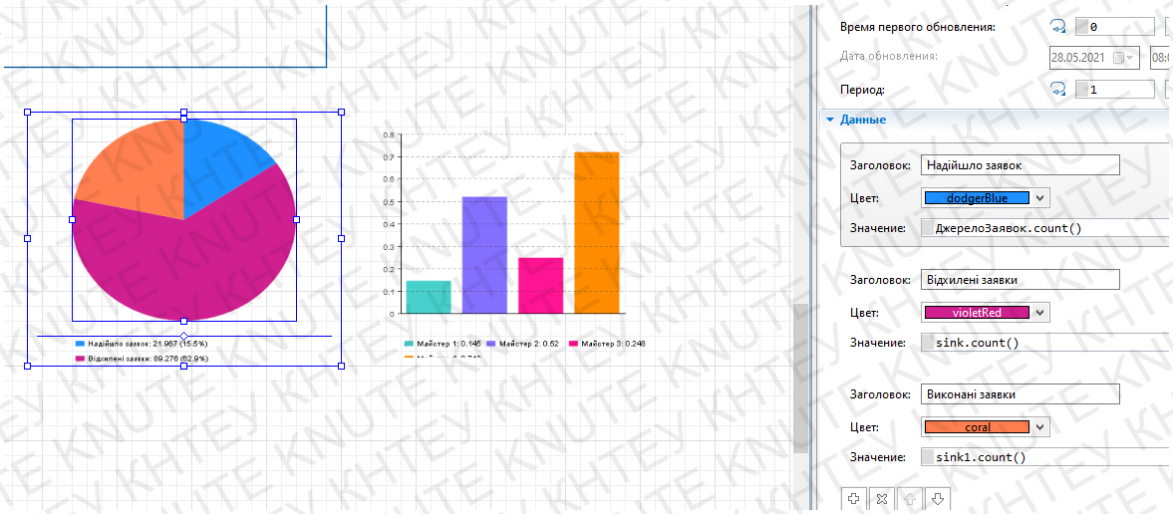

Рис. 3.19. Кругова і стовпчикова діаграма продуктивності.

3.3. Рішення задачі обліку замовлень у середовищі C++.

Для вирішення задачі обліку замовлень, необхідно створити комплекс комп'ютерних програм на C++, а саме:

1) Написати програмний код довідкової інформації для користувача про використання програми.
2) Написати програмний код для формування замовлення.
3) Написати програмний код реалізації класу двох-зв'язного елемента для двох-зв'язного списку.

1. Програмний код довідкової інформації для користувача про використання програми та реалізація консольного призначеного для користувача інтерфейсу через цикл:

```
#include "RegBids.h"
 // Точка входу в програму
int main()
{
   // Міняємо "locale", для коректного виведення українських символів
   setlocale(LC_ALL, "Ukraine");
   RegBids RB;
```

```
    RB.LoadBids();
 // Реалізація консольного призначеного для користувача інтерфейсу через цикл
    bool loop = true;
    int user;
    while(loop)
    {
// Очистка екрана
    system("cls");
 // Довідкова інформація для користувача про використання програми
    cout << "1. Додання нового замовлення " << endl;
    cout << "2. Видалити дані з інформаційної системи" << endl;
        cout << "3. Висновок відомостей по поломці із запитаним номером" << endl;
    cout << "4. Висновок відомостей по всіх поломках" << endl;
    cout << "0. Вихід з програми " << endl;
    cout << " Вводьте - ";
// Отримання даних від користувача з наступною обробкою цього введення
    cin.clear();
    cin.sync();
    cin >> user;
    switch(user)
    {
    case 1:
      RB.AddBid();
      break;
    case 2:
      RB.Delete();
      system("pause");
```

```
            break;
        case 3:
            RB.PrintNumber();
            break;
        case 4:
            system("cls");
            cout << " Висновок відомостей по всіх поломках:" << endl;
            RB.PrintBase();
            system("pause");
            break;
    // Реалізація виходу з циклу
        case 0:
        loop = false;
        default: break;
        }
    }
    return 0;
}
```

2. Програмний код комп'ютерної системи для формування замовлення на ремонт або діагностику у сервісному центрі та реалізація пошуку і видалення замовлень:

```
#include "RegBids.h"
 string GetString(string invite)
{
    string result;
    bool loop = true;
    while(loop)
        {
            cout << invite;
            cin.clear();
```

```
            cin.sync();
            getline(cin, result);
            if(!cin.fail())
            loop = false;
            else
            cout << "Некоректне введення" << endl;
        }
    return result;
}
 int GetInt(string invite)
{
    int result;
    bool loop = true;
    while(loop)
        {
            cout << invite;
            cin.clear();
            cin.sync();
            cin >> result;
            if(!cin.fail())
            loop = false;
            else
            cout << "Некоректне введення" << endl;
        }
    return result;
}
 void RegBids::push(string _stat, int _numb, string _time)
{
    Node *newNode = new Node(_stat, _numb, _time);
    if(start == 0)
```

```
    {
        start = end = newNode;
    } else {
        start->prevNode = newNode;
        newNode->nextNode = start;
        start = newNode;
    }
    count++;
}
void RegBids::LoadBids()
{
    ifstream file("Base.txt");
    if(!file.is_open())
        return;
    string newStat, newTime;
    int newNumber;
    int count;
    file >> count;
    int i = 0;
    while(i < count)
    {
        file >> newStat;
        file >> newNumber;
        file >> newTime;
        push(newStat, newNumber, newTime);
        i++;
    }
    file.close();
}
void RegBids::SaveBids()
```

```
{
    ofstream file("Base.txt", ios::out | ios::trunc);
    file << count << endl;
    Node *node = end;
    while(node)
    {
        file << node->station << endl;
        file << node->number << endl;
        file << node->time << endl;
        node = node->prevNode;
    }
    file.close();
}
void RegBids::AddBid()
{
    string newStat, newTime;
    int newNumber;
    system("cls");
    cout << "Додання нового замовлення:" << endl;
    newStat = GetString("Вкажіть тип поломки  - ");
    newNumber = GetInt("Опис технічного пристрою - ");
     newTime = GetString("Вкажіть бажану дату отримання замовлення
(приклад: 22.05.2021) - ");
    push(newStat, newNumber, newTime);
     cout << "Замовлення сформоване." << endl;
    SaveBids();
    system("pause");
}
void RegBids::Delete()
{
```

```
system("cls");
if(!start)
{
   cout << "У вас немає замовлень" << endl;
   return;
}
PrintBase();
int del = GetInt("Введіть № видаляємого замовлення - ");
int i = 1;
bool status = false;
Node *node = start;
while(node)
{
   if(i == del)
   {
      if(start != node && end != node)
      {
         node->nextNode->prevNode = node->prevNode;
         node->prevNode->nextNode = node->nextNode;
      }
      if(start == end)
         start = end = 0;
      if(start != end && start == node && node->nextNode)
      {
         node->nextNode->prevNode = 0;
         start = node->nextNode;
      }
      if(start != end && end == node && node->prevNode)
      {
         node->prevNode->nextNode = 0;
```

```
            end = node->prevNode;
        }
        status = true;
        delete node;
        break;
      }
      node = node->nextNode;
      i++;
    }
    if(status)
      cout << "Замовлення № - " << i  << " видалена" << endl;
    else
      cout << "Такого замовлення не існує" << endl;
    SaveBids();
  }
  void RegBids::PrintNumber()
  {
    system("cls");
    if(!start)
    {
      cout << "У вас немає замовлень" << endl;
      return;
    }
    int wantNumber = GetInt("Введіть номер замовлення - ");
    cout << " Висновок замовлень по фільтру " << endl;
     cout << "№ Замовлення " << setw(18) << "Тип поломки" << " " <<
setw(18) << " Дата подання завяки" << endl;
    Node *node = start;
    int i = 1;
    while(node)
```

```
        {
            if(node->number == wantNumber)
            {
                cout << i << " " << setw(4) << node->number << " " << setw(18) <<
node->station << " " << setw(18) << node->time << " "  << endl;
                i++;
            }
            node = node->nextNode;
        }
        system("pause");
    }
    void RegBids::PrintBase()
    {
        if(!start)
        {
            cout << " У вас немає Замовлень " << endl;
            return;
        }
         cout << "№ Замовлення " << setw(18) << "Тип поломки" << " " <<
setw(18) << " Дата подання" << endl;
        cout << setfill(' ');
        Node *node = start;
        int i = 1;
        while(node)
        {
            cout << i << " " << setw(4) << node->number << " " << setw(18) <<
    node->station << " " << setw(18) << node->time << " "  << endl;
            node = node->nextNode;
            i++;
        }
```

}

Запустивши програмний код маємо дану форму для формування нового замовлення. Вікно для формування нового замовлення наведено на Рис. 3.20.

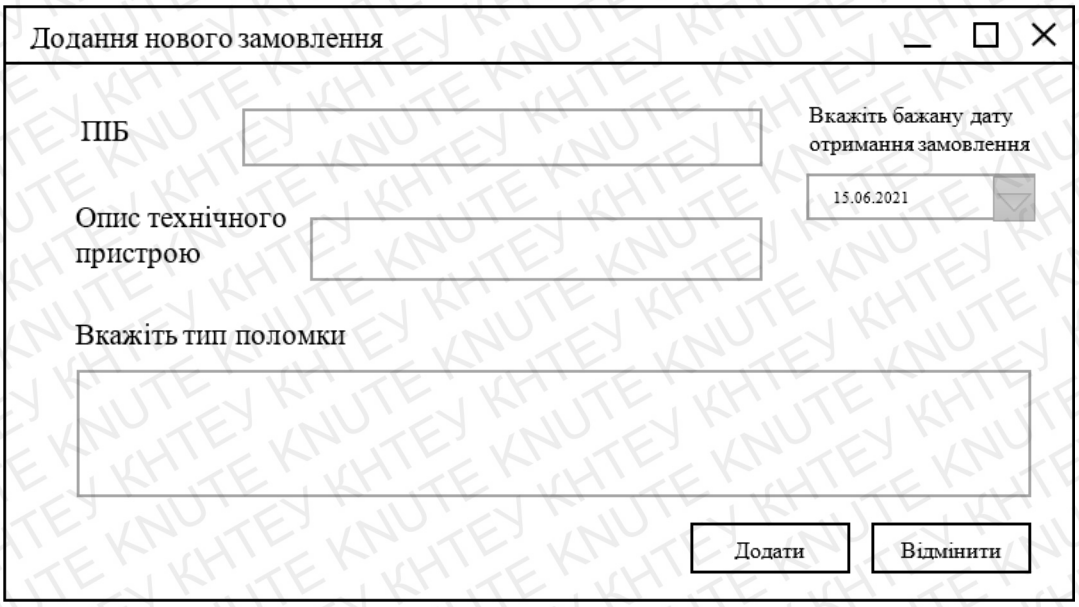

Рис. 3.20. Вікно формування нового замовлення.

3. Програмний код з реалізацією класу двох-зв'язного елемента для двох-зв'язного списку, оголошення використовуваних змінних та конструктор і деструктор, з ініціалізацією і коректним звільненням:

```
// Підключення бібліотек для роботи програми
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <time.h>
#include <locale.h>
using namespace std;
 // Реалізація класу двусвязного елемента для двусвязного списку
class Node {
public:
   // Оголошення використовуваних змінних
   Node *prevNode, *nextNode;
```

```
    string station, time;
    int number;
    // Конструктор і деструктор, з ініціалізацією і коректним звільненням
динамічної пам'яті
    Node() { prevNode = nextNode = 0; }
    Node(string _stat, int _numb, string _time) : station(_stat),
number(_numb), time(_time) { prevNode = nextNode = 0; }
};
 class RegBids {
    int count;
    Node *start, *end;
public:
    RegBids() { start = end = 0; count = 0;}
     void push(string _stat, int _numb, string _time);
     void LoadBids();
    void SaveBids();
     void AddBid();
    void Delete();
    void PrintNumber();
    void PrintStation();
    void PrintBase();
}
```

3.4 Тестування системи моделювання ведення та обліку замовлень на діагностику та ремонт побутової техніки у сервісному центрі.

Після запуску моделі та аналізу роботи ми можемо спостерігати як чітко і злагоджено працює модель ведення замовлень, тобто, як тільки поступає замовлення в чергу за нього відразу береться диспетчер, який перенаправляє замовлення до вільного, або майстра, котрий завершує попереднє замовлення. Модель ведення замовлень в дії зображено на Рис.

3.20. а також зображення статистики по замовленням та завантаженості кожного з майстра наведено на Рис. 3.21.

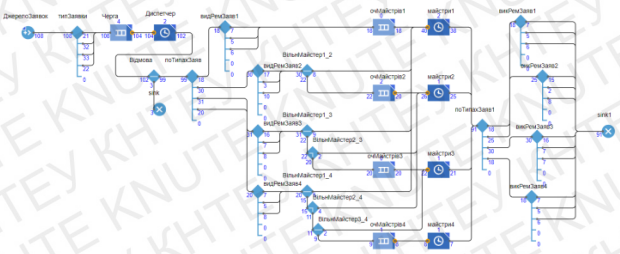

Рис. 3.21. Модель автоматизації обліку в дії.

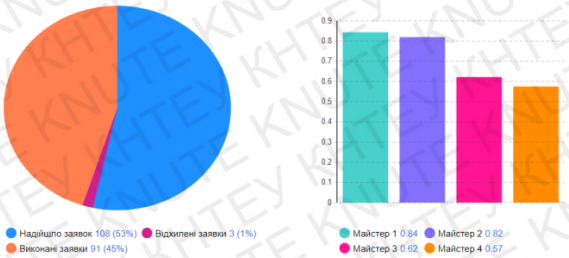

Рис. 3.22. Ведення замовлень та завантаженість кожного з майстрів.

Проаналізувавши результати ведення замовлень, можемо зробити висновок, що завдяки моделі ведення замовлень на ремонт та діагностику у сервісному центрі, відхилені замовлення становлять лиш 1% від всіх замовлень, також завантаженість кожного з майстрів не перевищує норму.

Також запуск комплексу комп'ютерних програм для обліку замовлень дає змогу побачити як працює програмний код та який вид має комп'ютерна програма для формування замовлень на ремонт або діагностику новими клієнтами у сервісному центрі. Вікно створення нового замовлення наведено на Рис. 3.23.

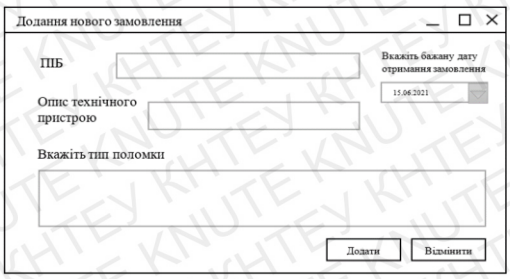

Рис. 3.23. Вікно комп'ютерної програми для створення нового замовлення

РЕЗУЛЬТАТИ ТА ВИСНОВКИ

У випускній кваліфікаційній роботі представлено результати теоретичних і прикладних досліджень, що полягають у розробці імітаційної моделі та написанню програмного коду для автоматизації обліку замовлень на діагностику у сервісному центрі.

В результаті проведених досліджень були отримані такі висновки:

1) Сервісні центри з часом стають все більш і більш затребувані, особливо в наш час, а тому його коректна робота залежить від чіткого ведення та обліку замовлень, що з часом стає все тяжче приймати замовлення в ручному стані, тому в сервісному центрі необхідне ведення та облік замовлень.
2) Впровадження і використання сучасних засобів ведення і обліку замовлень в управління сервісним центром забезпечує чітке та ефективне функціонування.
3) Розроблено імітаційну модель яка дає змогу проаналізувати роботу ведення замовлень у сервісному центрі в реальному часі.
4) Здійснено програмну реалізацію для обліку замовлень на діагностику та ремонт у сервісному центрі, довідкової інформації для користувача про використання програми та реалізації класу двох-зв’язного елемента для двох-зв’язного списку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Автоматизація процесів обліку і контролю у торгових мережах - https://www.vostok.dp.ua/ukr/infa1/Avtomatizatsiya/avtomatizaciya/ .
2) Автоматизація обліку від компанії СОБІ - http://sobigroup.com/products/avtomatizaciya-obliku .
3) Довідка - Програмне забезпечення для моделювання AnyLogic - https://help.anylogic.ru/index.jsp?topic=%2Fcom.anylogic.help%2Fhtml%2Fui%2Fpalette-view.html .
4) Облік і управління замовленнями. Оптимізуємо прийом замовлень на обслуговування - https://hubex.ru/upravlenie-zayavkami-i-urovnem-servisa/rabota-s-zayavkami/ .
5) Огляд сервісів з автоматизацією обліку замовлень - https://startpack.ru/application/.
6) Програма для сервісного центру - https://remonline.ua/service-center/ .
7) AnyLogic: імітаційне моделювання для бізнесу - https://www.anylogic.ru/ .
8) Consuting – рекрутивна компанія. Взаємодія з клієнтми та замовленнями - http://guverina.org.ua/news/uk/vedenie-klienti-zamovniki-posuk-zalucenna-oblik-vedenna-klientski-programi-bazi-avtomatizacia-posuku-zalucenna-obliku-vedenna-klientiv-zamovnikiv-kontragentiv-vartist-vprovadzenna-avtomatizovanih-program-sistem-stvorenna-ta-upravlinna-klientskou-bazou/ .
9) C ++ - Енциклопедія мови програмування - http://progopedia.ru/language/c-plus-plus/.
10) Help Desk система обліку та керування замовленнями - https://okdesk.ru/ .