Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

(тема)
Автоматизація управління технологічними процесами на основі мікросервісів

Виконав: студент 2 курсу, гр. ІТМРТм-19-1
Тютюнник Юлія Сергіївна
(прізвище, ініціали)

Спеціальність 172 Телекомунікації та радіотехніка

освітньої програми Інтелектуальні технології мікросистемної радіоелектронної техніки
(код і повна назва напряму)

Тип програми освітньо-професійна
(повна назва освітньої програми)

Керівник проф. каф. КІТАМ Новоселов С.П.
(посада, прізвище, ініціали)

Допускається до захисту
зав. кафедри _____ Невлюдов І.Ш.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет	Автоматики і комп'ютеризованих технологій
Кафедра	Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
Рівень вищої освіти	другий (магістерський)
Спеціальність	172 Телекомунікації та радіотехніка
Тип програми	освітньо-професійна
Освітня програма	Інтелектуальні технології мікросистемної радіоелектронної техніки

(код і повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____ 2020 р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Тютюнник Юлії Сергіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Автоматизація управління технологічними процесами на основі мікросервісів

затверджена наказом по університету від 02.11. 2020 р. № 1508 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 17. 12. 2020 р.

3. Вихідні дані до роботи Розподілені системи управління технологічними процесами; Автоматизоване управління процесом виробництва; Мікросервіси у виробництві; Технології керування системами за допомогою мікросервісів . Централізовані системи управління технологічними процесами.

4. Перелік питань, що потрібно опрацювати в роботі
4.1 Аналіз предметної області; 4.2 Аналіз технології керування складними об'єктами із застосуванням мікросервісів 4.3 Розробка методу взаємодії мікросервісів 4.4 Експериментальні дослідження

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Демонстраційний матеріал представлений у форматі презентації PowerPoint (*.ppt) – 15 с. формату А4

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Керівник (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз технічного завдання	18.10.2020	Виконано
2	Аналіз предметної області	01.11.2020	Виконано
3	Аналіз технології керування складними об'єктами із застосуванням мікросервісів	10.11.2020	Виконано
4	Розробка методу взаємодії мікросервісів	20.11.2020	Виконано
5	Експериментальні дослідження	01.12.2020	Виконано
6	Оформлення пояснювальної записки	07.12.2020	Виконано
7	Подання роботи до ЕК	17.12.2020	Виконано
8			

Дата видачі завдання 16.10.2020

Студент _____ Тютюнник Ю.С.

(підпис) (прізвище, ініціали)

Керівник роботи _____ проф. Новоселов С.П.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 71 с., 2 табл., 28 рис., 1 дод., 17 джерел.

МІКРОСЕРВІСИ, АВТОМАТИЗАЦІЯ, АСУТП, ПРОГРАМА УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ, C#, РОЗРАХУНКИ, MONODEVELOP, ПРОГРАМА КЕРУВАННЯ.

Об'єкт дослідження – розподілені системи управління технологічними процесами.

Предмет дослідження – автоматизоване управління процесом виробництва на основі мікросервісів.

Мета магістерської атестаційної роботи – розробка методу побудови та керуванням автоматизованої системи управління технологічними процесами з використанням мікросервісів.

В даній атестаційній роботі проведено аналіз систем управління ТП, розглянуті типи систем, та їх особливості побудови. Проведено аналіз технології керування складними об’єктами та описанні методики обмену повідомленнями між мікросервісами в системі.

Виконана розробка архітектури поєднання виробничих сервісів в моделі віртуального виробництва. Розроблено алгоритм роботы віртуальної виробничої дільниці та написано програму для моделювання процесу управління віртуальним виробництвом з використанням мікросервісної технології. Виявлені недоліки та переваги роботи з мікросервісами.

ABSTRACT

The explanatory note contains 71 p., 2 tables, 28 drawings, 1 pp., 17 sources.

MICROSERVES, AUTOMATION, ASMTP, TECHNOLOGICAL PROCESS MANAGEMENT PROGRAM, C#, CALCULATIONS, MONODEVELOP, CONTROL PROGRAM.

The object of study – distributed process control systems.

The subject of research – automated control of the production process based on microservices.

The purpose of the certification work is to develop a method of building an automated process control system using microservices.

In this attestation work the analysis of TP control systems is carried out, the types of systems and their features of construction are considered. An analysis of complex object management technology and a description of the method of messaging between microservices in the system.

The development of the architecture of the combination of production services in the model of virtual production is performed. The algorithm of work of a virtual production site is developed and the program for modeling of process of management of virtual production with use of microservice technology is written. Disadvantages and advantages of working with microservices are revealed.

ЗМІСТ

Зміст 6

Скорочення та умовні познаки 8

Вступ 9

1 Аналіз систем управління технологічними процесами 11

1.1 Аналіз особливостей побудови розподілених систем управління технологічними процесами 11

1.2 Порівняння централізованої і розподіленої архітектури АСУ ТП 16

1.3 Збір інформації для автоматизації промислових процесів 20

1.4 Мультиагентне управління розподіленим технологічним устаткуванням 21

1.5 Принцип проектування автоматизованої системи управління технологічними процесами в розподілених виробничих системах. 23

1.6 Висновки по першому розділу роботи 33

2 Аналіз технології керування складними об’єктами із застосуванням мікросервісів 34

2.1 Аналіз можливостей застосування мікросервісів в розподілених системах управління технологічними процесами 34

2.2 Відмінні ознаки мікросервісів 35

2.3 Опис методики обміну повідомленями між мікросервісами в технологічній системі 37

3 Розробка методу взаємодії мікросервісів 40

3.1 Розробка архітектури поєднання виробничих сервісів в моделі віртуального виробництва 40

3.2 Розробка алгоритму роботи віртуальної виробничої дільниці 46

3.3 Висновки по третьому розділу роботи 49
4 Експериментальні дослідження 50
4.1 Опис методу віртуалізації обчислень 50
4.2 Розробка програми для моделювання віртуального виробництва з використанням мікросервісної технології 51
4.3 Аналіз результатів моделювання 61
4.4 Висновки по четвертому розділу роботи 65
Висновки 66
Перелік джерел посилань 68
Додаток А Демонстраційний матеріал 70

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

АСУ – автоматизована система управління;

ВВС– віртуальна виробнича система;

МАУ – мультнагентне управління;

МК – мікросервіси;

НВП – науково-виробниче підприємство;

ПЗ – програмні засоби;

ПУ – пульт управління;

РВС – розподілена виробнича система;

РТК – робото-технічний-комплекс;

ТНС – транспортно-накопичувальна система;

ТОВ – товариство з обмеженою відповідальністю;

ТП – технологічний процес;

УВС – узагальнена виробнича система;

ЧПУ – числове програмне управління;

AGV – автоматично керовані транспортні засоби;

DCS – Distributed Control System;

IOT – Internet of Things;

LAM – вантажозахватні пристосування.

ВСТУП

Розподілені системи автоматичного управління знаходять найширше застосування у сучасних інтелектуальних виробництвах. У тому числі це відноситься і до систем, до складу яких входять дистанційно керовані роботи і маніпулятори, що відносяться до класу маніпуляційних робототехнічних систем.

Багаторівнева структура системи управління забезпечує її надійність, оперативність, ремонтопридатність. При цьому легко вирішується оптимальний рівень централізації управління з мінімальною кількістю засобів технологічного контролю, управління і ліній зв'язку між ними.

Багато проектів автоматизованих систем контролю і управління для великого спектру областей застосування дозволяють виділити узагальнену схему їх реалізації. Як правило, це дворівневі системи, так як саме на цих рівнях реалізується безпосереднє управління технологічними процесами.

Розвинена високопродуктивна система зв'язку є невід'ємною частиною системи управління. У системах управління для управління, збору даних і обміну інформацією використовуються промислові шини. Промислова шина повинна відповідати вимогам різних прикладних сфер, володіти відповідними характеристиками, завдяки яким їх можна використовувати в умовах промислової експлуатації. це: детермінованість, підтримка великих відстаней між вузлами, захист від електромагнітних наведень, зміцнена механічна конструкція.

Таким чином, актуальністю досліджень є необхідність вдосконалення виробничих систем і підвищення ефективності функціонування промислових об'єктів з метою досягнення високих кінцевих результатів діяльності на основі раціонального використання виробничих ресурсів.

Об'єктом дослідження є – розподілені системи управління технологічними процесами.

Предмет дослідження – автоматизоване управління процесом виробництва на основі мікросервісів.

Метою атестаційної роботи є розробка методу побудови та керуванням автоматизованої системи управління технологічними процесами з використанням мікросервісів.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– дослідити особливості побудови розподілених систем управління технологічними процесами;

– проаналізувати можливості застосування мікросервісів в розподілених системах управління технологічними процесами;

– розробити методику побудови автоматизованої системи управління технологічними процесами з використанням мікросервісів;

– розробити тестову програму та виконати експериментальні дослідження;

– оформити магістерську атестаційну роботу згідно ДСТУ 3008:2015, а також з методичними вказівками з розробки й оформлення магістерської атестаційної роботи другого (магістерського) рівня вищої освіти галузі знань 17 Електроніка та телекомунікації за спеціальністю 172 Телекомунікації та радіотехніка.

1 АНАЛІЗ СИСТЕМ УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ

1.1 Аналіз особливостей побудови розподілених систем управління технологічними процесами

Розподілена система управління (у перекладі з англійського Distributed Control System, скорочено DCS) – це комплекс технічних і програмних рішень для побудови АСУ ТП, характерною рисою якої є децентралізована обробка даних і наявність розподілених систем введення і виведення інформації, підвищена відмовостійкість, стандартна і єдина структура бази даних.

Інтерес до розподілених систем управління виник в процесі збільшення кількості датчиків, площ виробництва, модернізації і ускладнення стандартних алгоритмів управління технологічними процесами. Контролери, модулі введення і виведення даних, датчики, виконавчі механізми рознесені в просторі.

Кожен ПЛК працює зі своїми датчиками і виконавчими механізмами, працює з конкретною частиною об'єкту управління, і не залежить від інших ПЛК, проте, взаємодіє з іншими контурами і пристроями для виконання загального завдання, досягнення заданих показників якості. Ділянки Distributed Control System можуть знаходитися один від одного на будь-якій відстані, а зв'язок між ними підтримуватиметься через інтернет.

Подібна тенденція отримала свій розвиток у зв'язку з успіхами об'єктно-орієнтованого програмування. Для досягнення оптимального режиму роботи DCS обмін даними між контроллерами має бути зведений до мінімуму. Синхронізація ж роботи ПЛК проводиться на більш високому рівні АСУ ТП.

Основними вимогами до DCS є:

– відмовостійкість. Зупинка технологічного процесу для Distributed Control System є неприпустимою. Досягається це резервуванням технічної і програмної бази.

– масштабованість. Так само украй важлива для розподіленої системи управління, оскільки нарощування технологічної бази, додавання нових контурів і частин об'єкту управління повинні відбуватися просто і без зупинки функціонування вже підключених рішень.

До складу Distributed Control System повинна входити безліч необхідних елементів. Серед них: системи програмування, бібліотеки стандартних елементів, налагоджені мережеві інтерфейси і протоколи.

Структура DCS в найбільш класичному виді складається з трьох рівнів (рисунок 1.1).

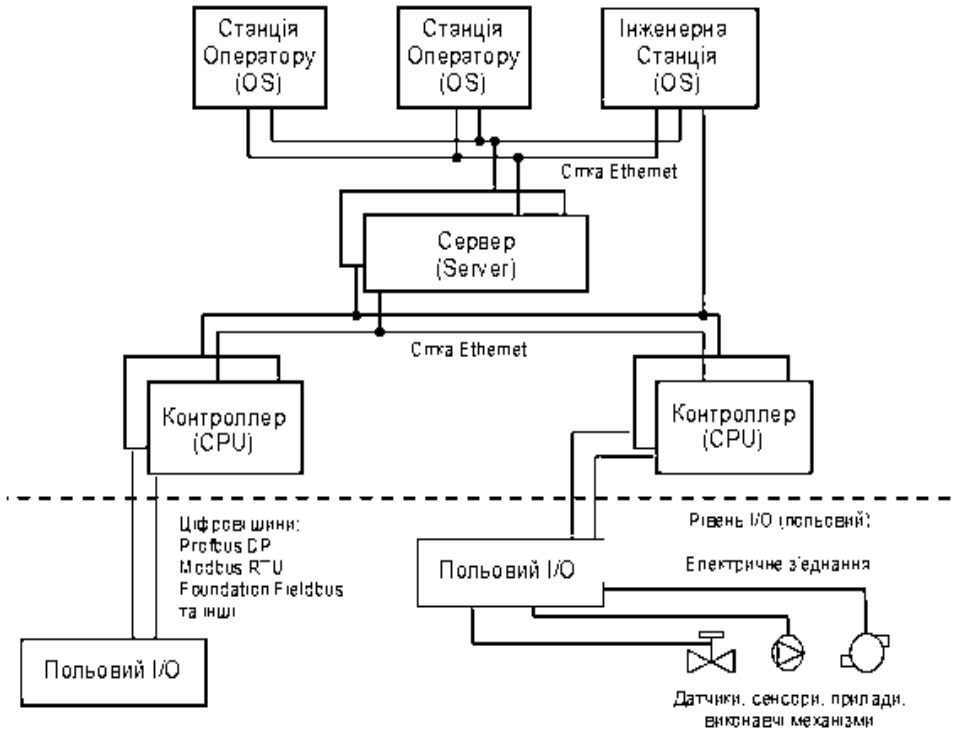


Рисунок 1.1 – Структура распределенной системы управления ТП

Нижній рівень введення/виводу містить в собі датчики, виконавчі механізми і інше [11]. Середній рівень розподіленої системи управління складається з контроллерів. Їх завдання обробити отримані дані, видати дію, що управляє, передати дані на верхній рівень. На верхньому рівні розташовані

сервери баз даних і операторські станції, завдання яких надати людино-машинний інтерфейс операторові і здійснювати обмін з сервером і ПЛК.

Часто в ієрархії Distributed Control System виділяють окремо інженерну робочу станцію, на якій встановлені комплекси розробки і конфігурації. Подібні станції мають пряме підключення до ПЛК у відмінності від операторських.

Людино-машинний інтерфейс [12] повинен надавати і візуалізувати усі процеси, що протікають в об'єкті і устаткуванні. Тут є невелика схожість DCS і SCADA програм. Відмінність Distributed Control System від SCADA полягає в різній функціональності цих рішень.

Розподілені системи управління технологічними процесами знайшли своє застосування в інтелектуальних виробництвах та досить актуальні в час переходу виробництв к Індустрії 4.0.

Більшість експертів в області світової Індустрії 4.0 єдині в розумінні трьох загальних характеристик (рис. 1.2). Цей фреймворк також показує, що 4.0 певною мірою є еволюцією (продовженням 3.0).

Інші важливі характеристики, які не наведені на рис. 1.3 і які в результаті роблять фабрики і заводи «розумними», це:

– інтероперабельність: кіберфізичні системи [17] дозволяють людям і розумному обладнанню (фабрикам) ефективніше поєднуватися один з одним;

– віртуалізація: в 4.0 можна створювати віртуальні копії розумних фізичних об'єктів (масштабованих від окремих пристроїв або машин до цілих заводів) і, відповідно, запускати різні механізми симуляцій, моделювання, а також оцінки реального стану;

– децентралізація: на відміну від високоцентралізованих підходів, у 3.0 і 4.0 кожна кіберфізична підсистема може робити власні рішення і взаємодіяти з іншими найбільш оптимальним способом;

– реальний час: всі дані та їх аналітику можливо отримувати в реальному часі;

Рисунок 1.2 – Головні характеристики і технології 4.0

– орієнтація на сервіси: кількість різних сервісів, як із взаємодії пристроїв і систем між собою, так і з взаємодії з людьми та учасниками екосистеми, зростає в рази;

– модульність: гнучка адаптація розумних фабрик до зовнішніх змін так само зростає, оскільки можна легко змінювати або розширювати окремі модулі систем керування.

Комунікація на виробництві в рамках технології 4.0 займає передове місце. В інтелектуальному виробництві обладнання, контролери і датчики взаємодіють як один з одним, так і безпосередньо з Ethernet або у хмарі.

Закрита система стає відкритою. Але змінюється не тільки кількість інформації, що обробляється безпосередньо на місці. Підвищується і якість до абсолютно нового рівня.

Інформація про стан виробничого обладнання та пов'язане з ним прогнозування про можливі простої виробництва за допомогою інноваційних систем зворотного зв'язку подає тільки один із прикладів. Це стало можливим завдяки швидкому збільшенню обчислювальної потужності, яку також можна вже використовувати децентралізовано в периферії, на краю мережі або в основах виробництва.

Подальший розвиток цифрового виробництва передбачає перенесення обчислень у хмарний сервіс, як показано на рис. 1.3.

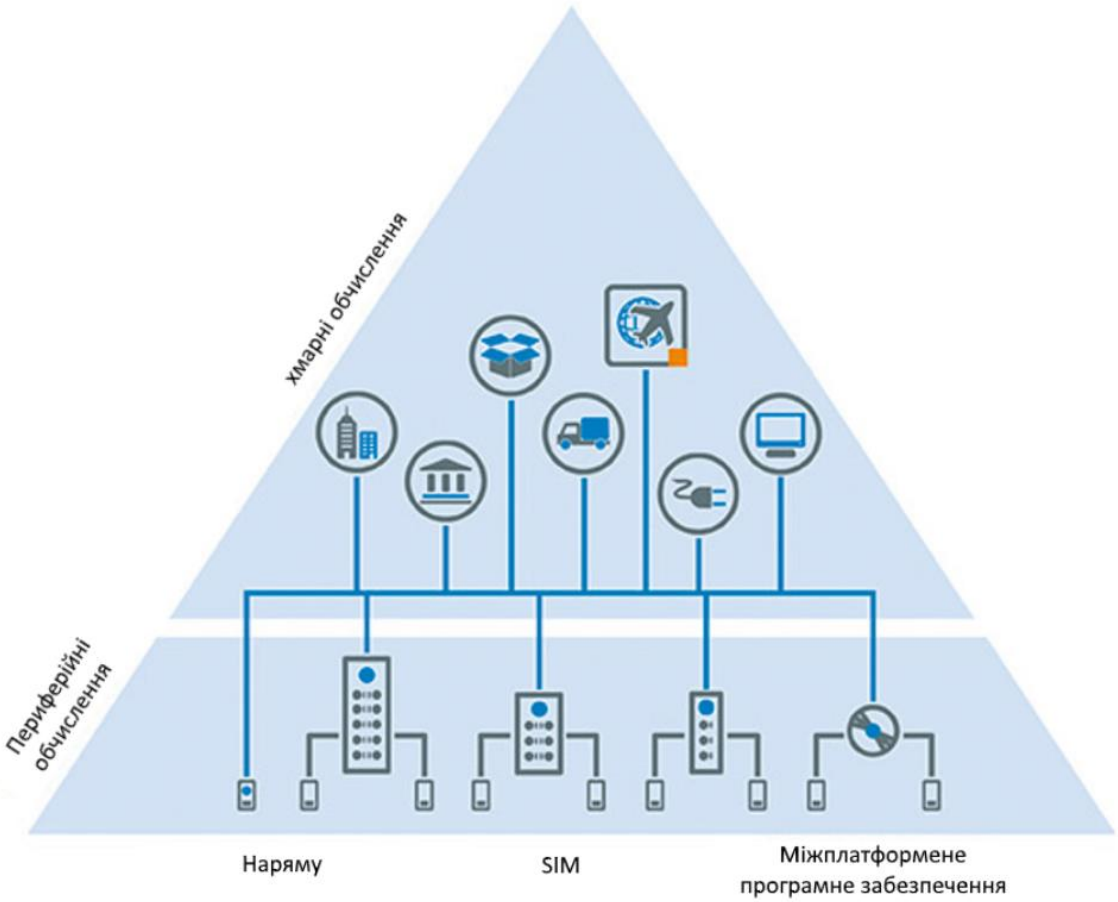


Рисунок 1.4 – Мережна інформація на цифровому виробництві

Таким чином, хмарний сервіс стає все більш важливим для управління загальними процесами. Але основна обчислювальна потужність все більше

переміщується на периферію. Датчики готують зібрані дані для передачі інформації, яка потім обробляється в Ethernet або у хмарному сервісі для подальшого процесу.

1.2 Порівняння централізованої і розподіленої архітектури АСУ ТП

Системи управління технологічними процесами застосовуються в різних галузях промисловості. При цьому кожному об'єкту автоматизації притаманні індивідуальні особливості технологічного процесу, які доводиться враховувати розробнику. Однак в цілому всю сукупність технологічних процесів розробники ділять на наступні типи:

– безперервний (Process Control);

– дискретно-безперервний (Batch Control);

– дискретний (Factory automation).

Управління описаними технологічними процесами підпорядковується загальної (класичної) моделі управління, наведеної на рисунку 1.2.

Архітектура сучасних систем АСУ ТП в даний час має два варіанти реалізації:

Централізовані – це системи які об'єднують керуючу підсистему, систему введення-виведення і відображення. приклади подібних систем: верстати з ЧПУ. пульти управління плавильними печами. автоматизовані лінії [11].

Розподілені – це системи які дозволяють:

– наблизити керуючий контролер до об'єкту:

– поділити підсистеми управління і візуалізації:

– управляти в масштабі підприємства.

Рисунок 1.2 – Модель управлення технологічними процесами

Централізовані АСУ ТП (рис. 1.3) зазвичай мають 2 рівня: верхній і нижній. На верхньому рівні розташовується обчислювальний пристрій. На нижньому – виконавчі приводи, датчики зворотних зв'язків.

Централізована АСУ ТП має ряд недоліків:

– необхідність застосування в керуючих контролерів потужних високопродуктивних процесорів. Оскільки всі завдання вирішуються тільки одним процесором, то при великій кількості периферійних пристроїв і модулів вводу-виводу він повинен мати достатньо велику продуктивність;

– великі труднощі, пов'язані з розширенням системи. При розширенні або модернізації потрібна заміна конструктиву контролера і модифікація або повна заміна програмного забезпечення;

– невисока надійність системи і підвищена схильність дії перешкод. Оскільки електронні блоки системи при такому підході зосереджені в одному місці, то до них доводиться прокладати велику кількість силових і сигнальних ланцюгів (від датчиків і виконавчих пристроїв), що при великих габаритах системи знижує надійність і збільшує вартість (велика витрата дорогої кабельної продукції) системи.

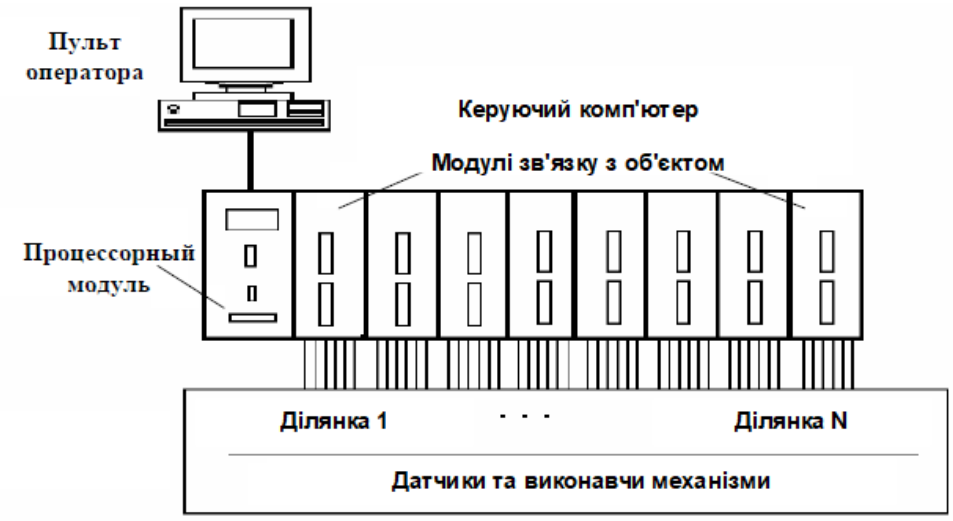


Рисунок 1.3 – Централізована АСУ ТП

Розподілені АСУ ТП (рис. 1.4), як правило, поділяються на 3 рівні: верхній, нижній і середній. На верхньому рівні розташовується автоматизоване робоче місце (АРМ) оператора.

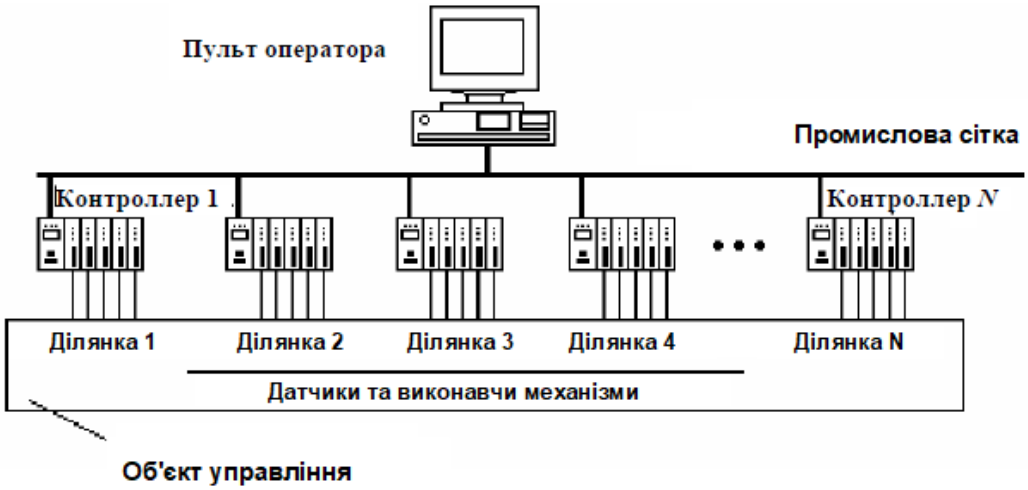


Рисунок 1.3 – Розподілена АСУ ТП

Переваги розподіленої АСУ ТП:

– висока надійність роботи системи. Чіткий розподіл обов'язків в розподіленої системі робить її працездатною навіть при виході з ладу або зависанні будь-якого вузла. При цьому працездатні вузли продовжують здійснювати збір даних і управління процесом або здійснюють послідовний останов технологічного обладнання;

– мала кількість дротових з'єднань. Контролери мають можливість працювати в важких промислових умовах, тому вони, як правило, встановлюються в безпосередній близькості від об'єкту управління. У зв'язку з цим істотно знижується витрата кабельної продукції, а для організації мережі, як правило, достатньо всього двох або чотирьох проводів;

– легка розширюваність системи. При появі додаткових точок контролю і управління досить додати в системи новий вузол (контролер, інтелектуальний модуль вводу-виводу);

– малі терміни проведення модернізації. Найбільший виграш досягається при модернізації великих систем, оскільки велика частина апаратних засобів і програмного забезпечення не вимагає модифікації.

– використання комп'ютерів і контролерів меншої потужності;

– легкість тестування і налагодження. Оскільки всі елементи системи активні, легко забезпечити самодіагностику і пошук несправності.

Останнім часом в області автоматизації технологічних процесів намітився перехід від централізованих систем, в яких один потужний процесорний модуль управляє великою кількістю пасивних периферійних пристроїв, до розподілених, де кожен елемент системи є активним пристроєм збору даних і управління [11].

У зв'язку з різким здешевленням мікропроцесорної техніки з одночасним підвищенням їх відмовостійкості, зменшенням їх розмірів і збільшенням їх функціональних можливостей з'явилася велика кількість малогабаритних контролерів і комп'ютерів, що володіють невисокою вартістю. Наявність розвинених мережевих засобів дозволяє пов'язувати ці контролери в єдину мережу, причому різні вузли (контролери, інтелектуальні модулі вводу-

виводу, комп'ютери) цієї мережі можуть бути рознесені один від одного на досить великі відстані, що робить найкращим варіант розподіленої АСУ ТП.

1.3 Збір інформації для автоматизації промислових процесів

Цифровізація та об'єднання обладнання в мережу – головна риса четвертої промислової революції. Нові технології роблять можливим злиття фізичного і віртуального світу в області виробництва і логістики в кіберфізичні системи (CPS). Машини можуть автономно обмінюватися одна з одною даними і, таким чином, оптимізувати перебіг процесів. При цьому Індустрія 4.0 чітко посилається на створення мереж у промисловому секторі.

Сенсорна техніка створює умови для прозорих процесів в Індустрії 4.0 (рис. 1.4). При цьому датчик утворює основу всіх наступних застосувань.

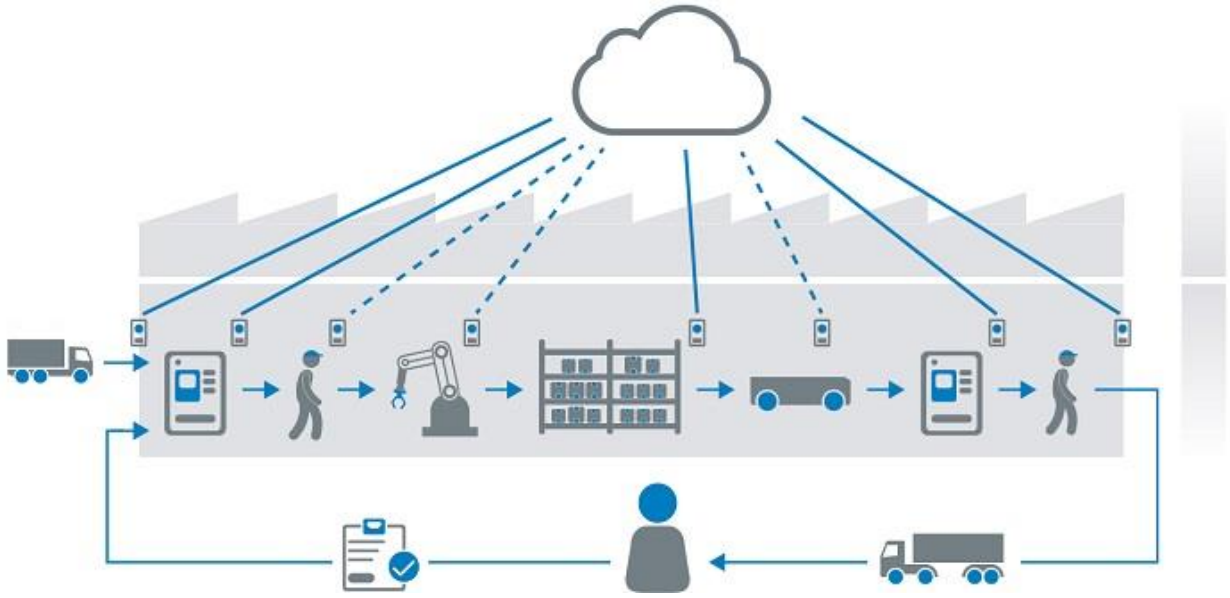


Рисунок 1.4 – Інтелектуальні датчики на мережевому заводі

На відміну від класичних, немережних датчиків, датчики Індустрії 4.0 надають більше, ніж просто дані вимірювань. Інтегрована, децентралізована обчислювальна потужність і гнучка програмованість є ключовими

характеристиками, які роблять виробництво більш гнучким, динамічним і ефективним.

Технологічний прогрес завжди є передумовою для змін у промисловості. Основою для динамічних, оптимізованих у реальному часі і самоорганізованих промислових процесів є отримання інформації та її подальша обробка. У них датчики виконують роль постачальників даних для інтелектуального виробництва. Сенсорна техніка є обов'язковою умовою для успішної реалізації Індустрії 4.0

Мережний завод є обов'язковою умовою для Індустрії 4.0. Кожен сенсор, кожна машина і всі залучені люди в будь-який час можуть спілкуватися між собою. Але цей обмін інформацією не закінчується у заводських воріт. Взаємодія між периферією і хмарним сервісом дозволяє управляти виробництвом і даними навіть ззовні.

Таким чином, це інтенсивне співробітництво між технологією і людиною робить процес більш прозорим, продуктивним і прибутковим.

1.4 Мультиагентне управління розподіленим технологічним устаткуванням

Спільне управління або управління розподіленими динамічними системами відноситься до ситуації, в якій кожен вузол може отримувати інформацію для управління тільки від самого себе і від своїх сусідів. Граф може задавати топологію мережі зв'язків, яка обмежує зв'язки між вузлами. Це також називають мультиагентним управлінням.

Розглянемо різні цілі управління, які вирішують системи мультнагентного управління (МАУ).

Синхронізація – збіг або зближення змінних стану двох або декількох систем, або погоджена зміна деяких кількісних характеристик систем.

Завдання синхронізації відрізняється від завдання управління з еталонною моделлю, оскільки в ній допускається збіг різних змінних, узятих в

різні моменти часу. Тимчасові зрушення можуть або бути постійними, або прагнути до постійних. Крім того, у багатьох завданнях синхронізації зв'язку між системами є двосторонніми (двонаправленими). Це означає, що граничний режим в системі (синхронне рішення) заздалегідь не відомий. Загальною особливістю завдань управління синхронізацією є те, що бажана поведінка однозначно не фіксована, а його характеристики задаються лише частково. У завданнях синхронізації часто основною вимогою є збіг або узгодженість коливань усіх підсистем, тоді як характеристики руху кожної підсистеми можуть варіюватися в широких межах [1]. У контексті МАУ під синхронізацією розуміють погоджену поведінку агентів, наприклад, повне або часткове зближення з часом станів агентів або їх спостережуваних виходів.

У МАУ така мета управління частіше формулюється як завдання досягнення консенсусу [2,3] чи узгодження характеристик [4], у яких кожен агент прагне зменшити відхилення своєї цільової змінної от відповідних змінних своїх сусідів. Якщо йдеться про відхилення від середнього арифметичного станів сусідів, то ставиться завдання досягнення усередненого консенсусу.

Завдання по перекладу групи агентів в деякий загальний стан називається завданням про зустріч агентів. У ній вимагається, щоб усі агенти опинилися в заданому місці в заданий час.

Ще одним класом завдань МАУ є розподілене управління формаціями (англ. formation control) – управління локально взаємодіючими агентами, що утворюють деякі геометричні конфігурації. Такі завдання зазвичай вирішуються стосовно мобільних роботів, безпілотних літальних апаратів, підводних автономних облаштувань та ін. Завдання групового управління (управління формаціями) [5] зводяться до завдань часткової координатної синхронізації за допомогою методу віртуального лідера.

При великому числі агентів (а у ряді завдань число агентів досягає тисяч і мільйонів), вимога заданої поведінки усіх без виключення агентів виявляється надмірно жорсткою і важкоздійснюваною. У таких випадках

виділяється характерна точка у безлічі станів агентів (центр, лідер, центр тяжіння), а бажаною поведінкою є задана поведінка центру за умови обмеженості відхилень від нього станів усіх агентів [6, 7].

Окремий клас завдань МАУ – розподіл ресурсів між різними можливими завданнями, завдання диспетчеризація [16] (англ. scheduling), яка зводиться до завдання досягнення консенсусу.

1.5 Принцип проектування автоматизованої системи управління технологічними процесами в розподілених виробничих системах.

Процес функціонування розподіленого виробництва характеризується, зокрема, ступенем завантаження технологічного обладнання її складових. У реальних умовах, при виконанні виробничих завдань, неминучий простий технологічного обладнання. Це пояснюється наступними факторами [5]:

– неможливість підбору обладнання за технологічними можливостями, що однозначно відповідає вимогам виконуваних ТП;

– неритмічністю проходження планових виробничих завдань;

– наявність запасу за кількістю і технологічним можливостями технологічного обладнання з метою гарантованого забезпечення необхідних умов функціонування виробничих систем.

Все це призводить до появи певного обсягу надмірності виробничих ресурсів. Розмір і склад ресурсів, а також характер їх зміни в часі є обумовленими параметрами виробничого процесу.

При виконанні виробничих завдань при розподіленому підході к організації підприємства виробничі процеси в ньому протікають незалежно один від одного. На рисунку 1.5 представлена схема проходження виробничого завдання через систему управління виробництвом. В і-й розподіленій виробничій системі (РВС) ($1 <= i <= n$) виконується m_i, видів готової продукції ($ГП_i$). Для виконання цього обсягу РВС, повинна володіти

технологічною гнучкістю, що дозволяє реалізувати відповідні технологічні процеси.

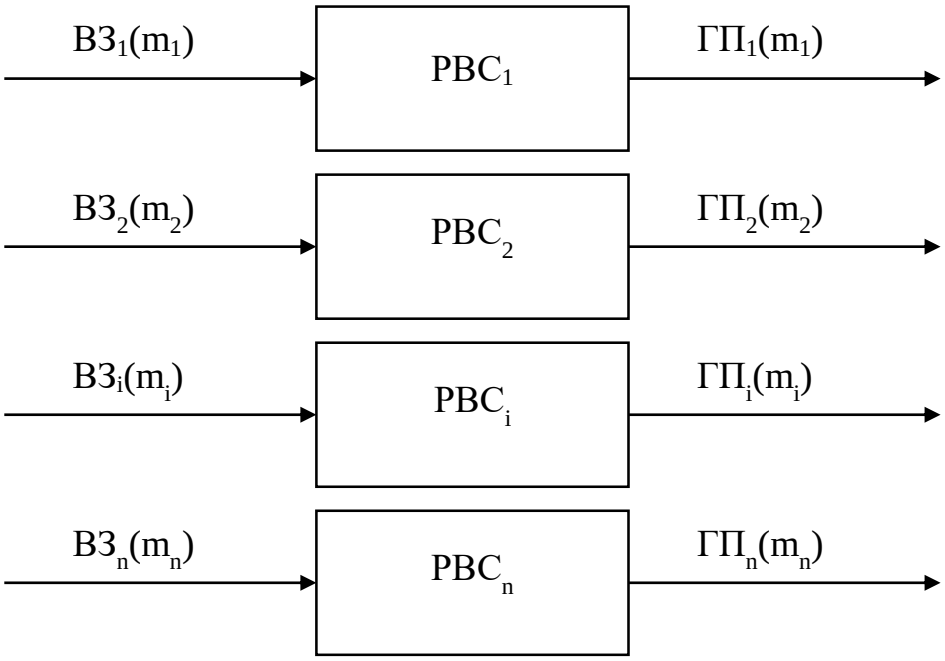


Рисунок 1.5 – Приклад проходження виробничого завдання через розподілений виробничий процес

Гнучкість системи VS_i характеризує технічно допустимі можливості переналагодження системи по окремим робочим параметрам (або їх сукупності) і являє собою наступну залежність:

$$VS_i = \bigcup_{j=1}^{n_{\text{т.oi}}} \left(\bigcup_{k=1}^{q_{i,j}} vs_{i,j,k} \right), \tag{1.1}$$

де $n_{m.oi}$ – кількість одиниць технологічного обладнання в і-й виробничій системі;

q_{ij} – множина значень параметрів *j-ї* одиниці технологічного обладнання, що є досяжними при її переналагодженні;

vs_{ijk} – унітарне значення параметра технологічних можливостей одиниці обладнання.

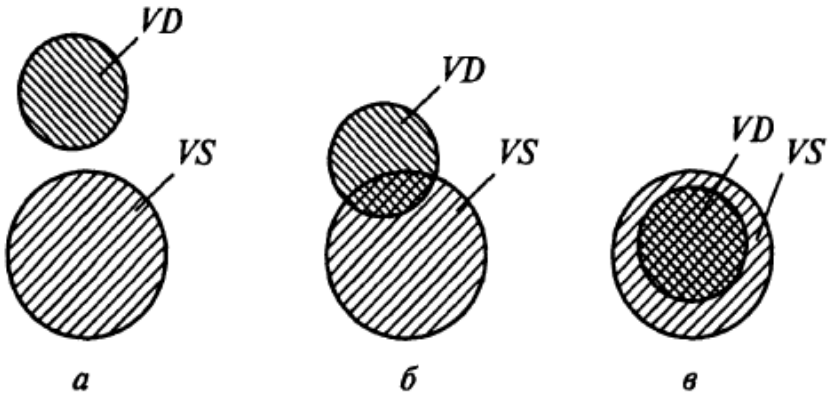
Технологічно необхідна гнучкість VD_i системи характеризується значеннями змінних параметрів, що відповідають значенням властивостей змінюваних деталей:

$$VD_i = \bigcup_{j=1}^{m_i}(\bigcup_{k=1}^{h_{i,j}} vs_{i,j,k}), \quad (1.2)$$

де vd_{ijk} – множина операцій, що необхідні для виконання m_i різних деталей;

h_{ij} – кількість операцій, необхідних для отримання j-ї готової деталі.

На рисунку 1.6 наведені різні варіанти розподілу технічно можливою гнучкості і технологічно необхідної гнучкості.



а – система не може бути використана;

б – система може використовуватися частково;

в – система використовується повністю

Рисунок 1.6 – Співвідношення технічно можливою гнучкості і технологічно необхідної гнучкості

На рисунку 1.6, а представлений випадок, коли РВС не можна використовувати для виконання заданого технологічного процесу навіть частково. Тут безліч параметрів технічно можливою гнучкості РВС, не має загальної області з безліччю параметрів технологічно необхідної гнучкості, тобто

$$VS_i \not\subset VD_i. \tag{1.3}$$

На рисунку 1.6,б представлений випадок, коли РВС, можна використовувати для виконання лише частини заданого технологічного процесу. Множина технічно можливої гнучкості має загальну область з множиною технологічно необхідної гнучкості

$$VS_i \cap VD_i. \tag{1.4}$$

На рисунку 1.6, в представлений випадок, коли РВС, може повністю забезпечити виконання заданого технологічного процесу. Множина технологічно необхідної гнучкості є підмножиною технічно можливою гнучкості

$$VS_i \supset VD_i. \tag{1.5}$$

У реальних виробничих системах має місце третій з наведених випадків. Таким чином, можна розглядати кожну виробничу систему як деякий масив параметрів, що характеризують її технологічні можливості, причому деяка частина цього масиву в кожен момент часу буде зайнята виконанням заданих технологічних процесів, а інша частина буде залишатися незавантаженою. Причому ці частини будуть змінюватися як за своїм складом, так і за співвідношенням розмірів в кожен момент часу у міру виконання одних технологічних процесів і початку інших. При цьому в кожен момент часу t

можна уявити сукупність вільних частин РВС як деяку віртуальну виробничу систему, характерною особливістю якої є те, що вона складається з частин РВС, які об'єднані лише організаційно, без проведення будь-яких матеріальних перетворень.

У кожний період часу технологічні можливості кожної РВС в основному пов'язані з виконанням деякого об'єму виробничих задач, на які вони орієнтовані. При цьому деяка їх частина виявляється вільною. Сукупність вільних частин РВС може розглядатися як узагальнена виробнича система (УВС), що володіє технологічними можливостями, які перевищують технологічні можливості окремих РВС. Таке об'єднання є умовним і інформація про УВС знаходиться тільки в пам'яті головного ПК автоматизованої системи. На рисунку 1.7 представлений процес формування УВС на основі множини РВС.

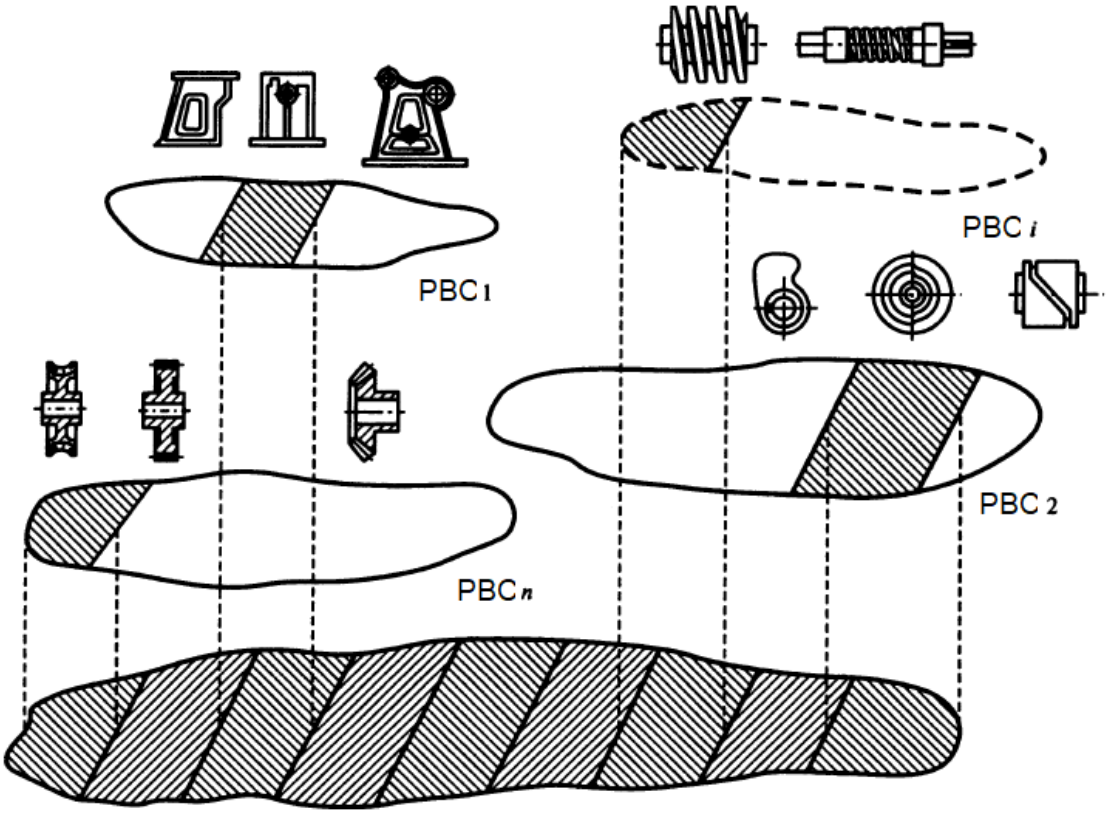


Рисунок 1.7 – Процес формування УВС на основі множини РВС

Технологічні можливості УВС можна представити в наступному вигляді:

$$WO = f(\bigcup_{i=1}^{n} VD_i;\ K_{vs1}, K_{vs2}, \dots, K_{vsi}, K_{vsn}), \qquad (1.6)$$

де K_{vsi} – кількість однакових технологічних операцій, які можуть бути суміщені ($1 <= i <= n$).

Потужність масиву технологічних можливостей УВС змінюється в часі. Це викликано тим, що виробничі завдання, що надходять в РВС, змінюються по номенклатурі і об'єму партій. При цьому співвідношення зайнятих і вільних частин РВС також змінюється. На рисунку 1.8 показано зміна потужності масиву технологічних можливостей у часі.

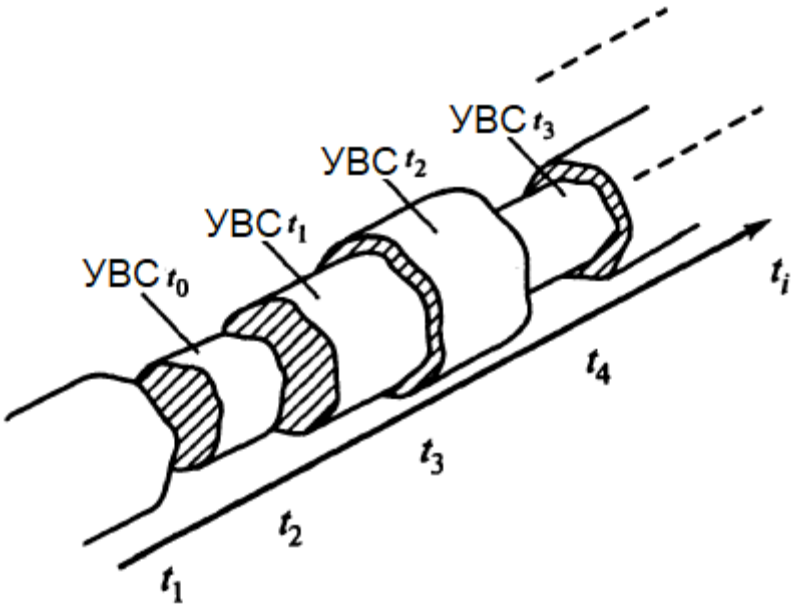


Рисунок 1.8 – Зміна потужності масиву технологічних можливостей у часі.

На відрізках часу t_i – t_{1+1} (i - довільна величина) потужність масиву технологічних можливостей є постійною величиною. У моменти часу t_i, відбувається закінчення чергової технологічної операції і (або) початок наступної, і потужність масиву змінюється стрибкоподібно.

У зв'язку з визначеністю термінів виконання планових завдань можливе визначення параметрів окремих РВС, складових УВС. Отже, з одного боку, відомо розподіл технологічних можливостей УВС в часі, з іншого – є певний обсяг виробничого завдання, який має бути виконаний. Виробниче завдання є завдання на виготовлення конкретних видів виробів заданої кількості.

Однотипні за складом технологічного обладнання РВС в параметре *WO* збільшують другу складову. Різнотипні РВС – збільшують першу складову. Параметр *WO* є комплексним параметром, що характеризує технологічні можливості УВС.

Технологічні можливості УВС можна представити у вигляді шарів (рисунок 1.9). Кожен шар характеризує фізичну прив'язку відповідної множини технологічних можливостей до конкретної РВС.

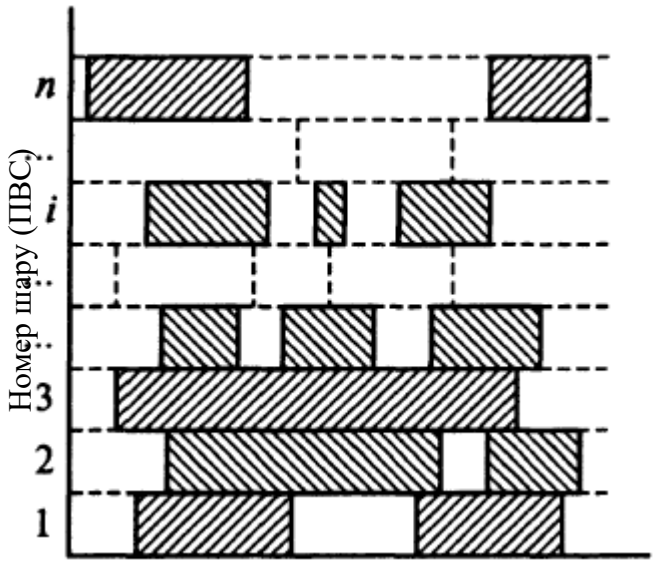


Рисунок 1.9 – Розподілення шарів в РВС

Якщо маршрут обробки передбачає перехід з одного шару в інший, з'являються додаткові витрати, пов'язані з транспортуванням виробу на етапі його виготовлення:

$$Q_1 = \sum q_{\text{внутр}} + \sum q_{\text{внеш}}, \tag{1.7}$$

де $q_{внутр}$ – витрати на транспортування виробу в процесі його виготовлення в межах однієї РВС;

$q_{зовн}$ – витрати на транспортування виробу в процесі його виготовлення при переміщенні з однієї РВС в іншу.

Метою формування виробничої системи є виконання виробничого завдання в задані терміни при мінімальній собівартості. З цією метою необхідно оптимізувати обсяг технологічних можливостей, задіяних для його виконання. При цьому сумарні витрати Q_{Σ} на виконання кожного виробничого задання повинні бути мінімальні:

$$K_1 Q_1 + K_2 Q_2 + K_3 Q_3 = \min Q_{\Sigma}, \tag{1.8}$$

де Q_1 – витрати на транспортування виробу в процесі його виготовлення;

Q_1 – витрати на обробку при виготовленні виробу;

Q_1 – витрати на штрафи, передбачені за порушення строків виконання виробничого завдання;

К1, К2, К3 – коефіцієнти значущості витрат.

Для виконання заданих виробничих завдання потрібне використання частини ресурсів УВС. Отже, на основі УВС необхідно сформувати виробничу систему, яка найкращим чином відповідає вимогам забезпечення виконання виробничого завдання. Оскільки інформація про організацію такої виробничої

системи міститься тільки в пам'яті головного ПК виробництва, вона є віртуальною виробничою системою (ВВС).

При формуванні ВВС необхідно виконання наступних умов:

– мінімізувати обсяг використовуваних ресурсів, необхідних для виконання виробничого завдання;

– максимізувати позитивний результат, який можна отримати з наявним запасом ресурсів;

– отримати найкращий баланс витрат (використовуваних ресурсів) і результатів (виконання виробничого завдання).

Час життя конкретної конфігурації ВВС визначається, з одного боку, вимогами поточного технологічного процесу, з іншого – ресурсами УВС, обумовленими можливостями кожної РВС з точки зору їх власного завантаження поточними технологічними процесами.

Процеси формування ВВС характеризується розподілом ресурсів УВС в часі і побудовою ТП для виконання виробничого завдання. Оскільки виробниче завдання складається з декількох видів виробів, процес проектування ТП виготовлення кожного з них буде визначатися результатами проектування ТП виготовлення інших видів. Таким чином, процеси проектування ТП всього виробничого завдання виявляються взаємопов'язаними і залежними один від одного, що визначає багатооб’єктність технологічного проектування в умовах ВВС.

Процес технологічного проектування в ВВС визначається наступними параметрами:

– розподілом виробничих ресурсів УВС;

– характеристиками кожного найменування виробів;

– кількістю виробів кожного найменування;

– термінами їх виконання.

Формування чергової конфігурації віртуального виробничого процесу визначається взаємодією двох інформаційних потоків:

– вироби, виготовляються;

– параметри УВС.

Управління процесами взаємодії окремих частин УВС при формуванні ВВС забезпечує виконання заданих виробничих завдань при мінімальних економічних затратах. Під управлінням будемо розуміти формування командної інформації на основі прийняття рішень з комплексної проблеми розподілу ресурсів і використання технологій.

Особливістю такого управління є в багатьох випадках попереднє прийняття рішень, так як досягнення бажаного стану ВВС залежить від набору взаємопов'язаних рішень (системи рішень). Інша особливість полягає в тому, що рішення, прийняті на ранніх етапах процесу управління, повинні прийматися з урахуванням їх впливу на наступні рішення.

Управління пов'язане, з одного боку, із запобіганням помилкових дій за рахунок виявлення таких на ранніх етапах, з іншого – зі зменшенням обсягу невикористаних можливостей існуючих виробничої системи. Велика кількість параметрів, які впливають на процес прийняття рішення, великий обсяг обчислень при моделюванні поведінки ВВС в часі і обмеженість часу вимагають побудови системи управління на основі інтелектуального управління . Інтелектуальне управління виробництвом дозволяє виконувати функції людини при підготовці вихідної інформації, моделюванні, аналізі отриманої інформації і формуванні командної інформації в умовах обмеження часу у багатьох випадках без участі самої людини.

1.6 Висновки по першому розділу роботи

В результаті виконання першого розділу магістерської атестаційної роботи виконали аналіз особливостей побудови розподілених систем управління технологічними процесами. Виконали порівняння централізованої і розподіленої архітектури АСУ ТП. Наведені переваги та недоліки кожної із зазначених архітектур.

Проведено аналіз застосування різних методик для управління розподіленими динамічними системами. Розглянуто принципи проектування автоматизованої системи управління технологічними процесами в розподілених виробничих системах.

2 АНАЛІЗ ТЕХНОЛОГІЇ КЕРУВАННЯ СКЛАДНИМИ ОБ’ЄКТАМИ ІЗ ЗАСТОСУВАННЯМ МІКРОСЕРВІСІВ

2.1 Аналіз можливостей застосування мікросервісів в розподілених системах управління технологічними процесами

Мікросервіс – це незалежний сервіс, що надається віддаленим ресурсом для використання рештою системи, що володіє однією і тільки однією надзвичайно вузько функціональної можливістю [7]. Наприклад, розглянемо систему управління сховищем інтелектуального виробництва. Якщо розбити її можливості на складові частини, вийде наступний список:

– приймаємо товар, що надходить до сховища;

– визначаємо, де повинен зберігатися новий товар;

– розраховуємо маршрути розміщення всередині складу, щоб помістити товар в потрібні сховища;

– розподіляємо маршрути розміщення товару між працівниками складу;

– отримуємо замовлення;

– розраховуємо маршрути вилучення товару зі складу для складання замовлень;

– розподіляємо маршрути вилучення товару між працівниками сховища.

Таким чином, кожна маленька функція в системі реалізується у вигляді окремого мікросервіса. Будь мікросервіс в системі:

– працює у власному окремому процесі;

– може бути розгорнуто окремо, незалежно від інших мікросервісов;

– має власне сховище даних;

– взаємодіє з іншими мікросервісамі для виконання своїх завдань.

Важливо також відзначити, що мікросервіси можуть взаємодіяти з іншими мікросервісамі, не обов'язково написаними на тій же мові програмування. Все, що їм потрібно знати, – як обмінюватися повідомленнями один з одним. Деякі можуть обмінюватися повідомленнями через службову

шину або двійковий протокол в залежності від системних вимог, але набагато частіше мікросервіси обмінюються повідомленнями через протокол HTTP.

2.2 Відмінні ознаки мікросервісів

Мікросервіс – це сервіс з однією вузьконаправленою функціональною можливістю. Існує шість основних відмінних ознак мікросервіса.

– мікросервіс відповідає за одну функціональну можливість;

– мікросервіс можна розгортати окремо;

– мікросервіс складається з одного або декількох процесів;

– у мікросервіса є власне сховище даних;

– невелика команда розробників може супроводжувати декілька мікросервісів;

– мікросервіс можна легко замінити.

Мікросервіси повинні допускати швидке розгортання окремо один від одного. При зміні мікросервіса повинна існувати можливість ввести цей змінений мікросервіс в експлуатацію, не розгортаючи (взагалі не зачіпаючи) будь-які інші частини системи. Решта мікросервісів в системі повинні виконуватися і працювати під час розгортання зміненого мікросервіса і продовжувати виконуватися після того, як нова версія буде розгорнута.

Можливість індивідуального розгортання кожного з мікросервісів важлива, оскільки в системі їх може бути дуже багато і кожен може взаємодіяти з декількома іншими. Одночасно з цим виконується розгортання деяких або всіх мікросервісів.

Щоб мати можливість розгортати окремий мікросервіс в той момент, коли інша система продовжує працювати, процес складання повинен бути налаштований з огляду на таке:

– всі мікросервіси повинні бути вбудовані в окремі контейнери або пакети;

– процес розгортання повинен бути налаштований таким чином, щоб забезпечити підтримку індивідуального розгортання при продовженні функціонування інших мікросервісів. Наприклад, можна використовувати процес плаваючого розгортання, при якому мікросервіс розгортається на одному сервері за один раз, щоб знизити час простою.

У мікросервіса має бути власне сховище, в якому знаходяться необхідні йому дані. Це ще один наслідок того, що сфера дії мікросервіса – завершена функціональна можливість. Більшій частині бізнес-можливостей потрібно якесь сховище даних.

Те, що у кожного мікросервіса є власне сховище даних, дозволяє використовувати різні технології баз даних для різних мікросервісів в залежності від потреб конкретного технологічного процесу. Такий підхід дає кожному мікросервісу можливість використовувати оптимальну для його завдань базу даних, що приносить виграш в сенсі часу розробки, продуктивності і масштабованості. Очевидний недолік – необхідність адміністрування та супроводу декількох баз даних.

Розподілена архітектура па основі мікросервісів дає можливість масштабувати кожен з сервісів окремо з урахуванням вузьких місць. Більш того, мікросервіси підтримують асинхронну взаємодію за допомогою подій і надають особливого значення відмовостійкості всюди, де потрібно синхронна взаємодія. Ці властивості при правильній реалізації забезпечують високу доступність і хорошу масштабованість інформаційній системи керування розподіленими процесами.

2.3 Опис методіки обміну повідомленнями між мікросервісами в технологічній системі

2.3.1 Механізм локализації межсервісних взаємодій

Механізм локалізації міжсервісних взаємодій складається з чотирьох фаз [15]: фази моніторингу, фази розподілу завдань, фази взаємодії і фази перерозподілу обов'язків.

Фаза моніторингу. Менеджер розподілу завдань оцінює інтенсивність міжвузлових і внутрішньовузлових комунікацій в системі [14] за допомогою системного монітора і менеджера повідомлень. Менеджер, також, використовує інформацію як про сервіс-посилача кожного повідомлення, і про сервису-отримувача. Цей програмний модуль періодично обчислює комунікаційні залежності $C_{ij}(t)$ у момент часу $t \in T=[t_1; t_2]$ між сервісом і-го та сервісом j-го вузла по формулі:

$$C_{ij}(t)=\alpha(M_{ij}(t)/\sum_k M_{ik}(t))+(1-\alpha)C_{ij}(t-1), \quad (2.1)$$

де $M_{ij}(t)$ – число повідомлень, відправлених сервісом і-го і сервісом j-го вузла за період часу $T=[t_1; t_2]$,

α – коефіцієнт, що характеризує відносну значущість нової інформації по відношенню до застарілої і використовується для ігнорування тимчасової інтенсивної взаємодії з сервісами на певній платформі.

Під $C_{ij}(t-1)$ розуміється значення тієї ж комунікаційної залежності на попередньому інтервалі часу.

Фаза розподілу мікросервісів. Після певного числа повторень стадії моніторингу менеджер обчислює коефіцієнт комунікаційної залежності між поточним вузлом сервісу n і усіма іншими вузлами системи. Коефіцієнт комунікаційної залежності R_{ij} між сервісом і та сервісом j-го вузла визначається по формулі:

$$R_{ij}= C_{ij}/C_{in}, j\neq n. \quad (2.2)$$

Коли максимальне значення коефіцієнта комунікаційної залежності мікросервісу більше зумовленого порогу θ, менеджер поточного вузла включає даний мікросервіс в групу сервісів, розташовану на видаленому вузлі системи:

$$\text{k} = arg_j \max(R_{ij}) \wedge (R_{jk} > \theta) \rightarrow \alpha_i \in G_k, \quad (2.3)$$

де α_i представляє сервіс i;

G_k вказиває на групу сервісів k,

arg_j – операція, що повертає значення j, при якому коефіцієнт комунікаційної залежності R_{ij} приймає максимальне значення.

Фаза взаємодії. Перед переміщенням певної групи завдань з вузла R_1 на приймаючий вузол R_2 менеджер вузла R_1 взаємодіє з менеджером вузла R_2. Менеджер вузла R_2 перевіряє поточний стан пам'яті, завантаженість центрального процесора і число сервісів, розміщених на цьому вузлі, за допомогою системного монітора. Якщо вузол R_2 має достатню кількість вільних системних ресурсів для нових завдань, менеджер вузла R_2 дозволяє здійснення переміщення групи завдань з вузла R_1 на вузол R_2.

4. Фаза міграції сервісів. Завершивши операції відбору сервісів для міграції і отримання позитивного відгуку від приймаючого вузла, менеджер відправляючого вузла спільно з менеджером міграції сервісів ініціює переміщення вибраної групи сервісів на приймаючий вузол.

Механізм локалізації міжсервісних взаємодій припускає аналіз динамічних змін в моделях взаємодії сервісів, але цей механізм може викликати перевантаження деяких з вузлів системи із-за великого числа мікросервісів, переміщуваних на ці вузли. Коли вузол переобтяжений,

системний монітор виявляє цей стан і активує процедуру перерозподілу завдань, грунтовану на переміщенні не окремих завдань між вузлами мережі, а цілих груп сервісів, що інтенсивно взаємодіють один з одним.

2.4 Висновки по другому розділу роботи

В результаті виконання другого розділу магістерської атестаційної роботи виконали аналіз можливостей застосування мікросервісів в розподілених системах управління технологічними процесами [11]. Було дано визначення мікросервісу та його властивостям і поведінці при взаємодії з іншими об’єктами виробничої системи [4].

Описана методика обміну повідомленнями між мікросервісами в технологічній системі [6]. Було визначено, що механізм локалізації міжсервісних взаємодій складається з чотирьох фаз: фази моніторингу, фази розподілу завдань, фази взаємодії та фази перерозподілу обов'язків.

3 РОЗРОБКА МЕТОДУ ВЗАЄМОДІЇ МІКРОСЕРВІСІВ

3.1 Розробка архітектури поєднання виробничих сервісів в моделі віртуального виробництва

На рисунку 3.3 приведена схема взаємодії між програмними модулями, що виконують роль мікросервісів.

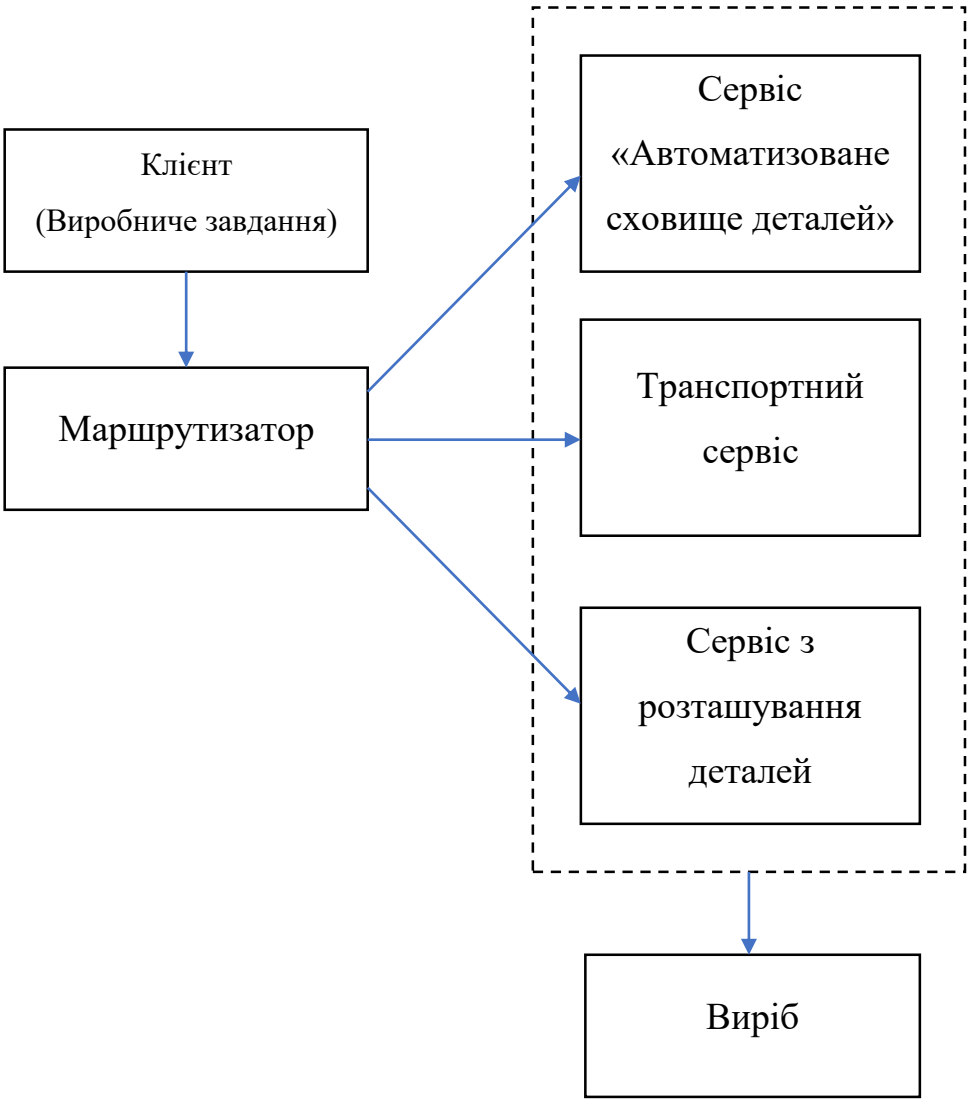


Рисунок 3.1 – Схема взаємодії між программными мікросервісами віртуального виробництва

Сервіс – це міні-додаток, що реалізує вузькоспеціалізовані функції, такі як управління замовленнями, управління клієнтами і т.д. Сервіси можуть масштабуватись в різних напрямках.

Узагальнене визначення мікросервісной архітектури [6] (або мікросервісов) звучить так: це стиль проектування, який розбиває додаток на окремі сервіси з різними функціями. Кожен сервіс повинен мати чіткий перелік пов'язаних між собою обов'язків.

Кожен сервіс є незалежним програмним модулем зі своє структурою даних. Виходячи з вимог до подібної технології, між собою сервіси пов’язані досить слабо. Синхронізація між сервісами відбувається через загальне середовище для обміну повідомленнями. Це можу бути загальна база даних або виділені програмні канали зв’язку.

Розглянемо формати повідомлень при взаємодії між програмними сервісами віртуального виробництва [13]. На початку роботи виконується пошук постачальників та замовників для організації віртуального виробного процесу.

При пошуку постачальників і замовників, транспортувальник сканує навколишній простір передаючи всім сервісам наступне повідомлення:

Усім: хто_розташовано_х?: {0}: ТРАНСПОРТ

У відповідь на запит про поточний стан сервісу він передає повідомлення наступного формату (рисунок 3.2).

В даному повідомленні:

– одержувач – адреса одержувача повідомлення (зазвичай найменування сервісу);

– статус – службове слово, яке визначає тип повідомлення;

– поточна_координата – координати розміщення сервісу на віртуальному поле;

– розмір – розмір графічного зображення сервісу;

– тип – тип сервісу (постачальник або замовник);

– стан – стан сервісу (активний або неактивний);

– відправник – адреса відправника повідомлення (зазвичай найменування сервісу).

Мітка одержувача
Статус
Поточна координата
Розмір
Тип
Стан
Мітка відправника

Рисунок 3.2 – Формат кадру відповіді від сервісу

Наступне повідомлення надсилається для отримання нового завдання:

одержувач: повідомлення? : відправник

В даному повідомленні:

– одержувач – адреса одержувача повідомлення (зазвичай найменування сервісу);

– повідомлення? – службове слово, яке визначає тип повідомлення;

– відправник – адреса відправника повідомлення (зазвичай найменування сервісу).

Наприклад, якщо повідомлення посилає транспортний робот, то воно буде виглядати так:

МАНІПУЛЯТОР : завдання? : ТРАНСПОРТ

Коли якийсь сервіс дає завдання іншому сервісу, то він посилає йому таке повідомлення (рисунок 3.3):

Мітка одержувача
Завдання
Текст завдання
Мітка відправника

Рисунок 3.3 – Відповідь на запит від сервісу

В даному повідомленні прийняті так поля даних:

– одержувач – адреса одержувача повідомлення (зазвичай найменування сервісу);

– завдання – службове слово, яке визначає тип повідомлення;

– текст завдання – текстове повідомлення, яке визначає те, що повинен зробити сервіс;

– відправник – адреса відправника повідомлення (зазвичай найменування сервісу).

Наприклад, так передає нове завдання промисловий маніпулятор на ділянці складання радіоелектронних модулів:

ТРАНСПОРТ: завдання: контролер STM32: МАНІПУЛЯТОР

Тут «контролер STM32» – це найменування компонента, який повинен бути доставлений промисловому маніпулятору [10] зі сховища.

Наступний тип повідомлень – це отримання замовлення. На рисунку 3.4 показано формат повідомлення для отримання замовлення.

Мітка одержувача
Замовлення
Текст замовлення
Мітка відправника

Рисунок 3.4 – Відповідь на запит від сервісу

В даному повідомленні:

– одержувач – адреса одержувача повідомлення (зазвичай найменування сервісу);

– замовлення – службове слово, яке визначає тип повідомлення;

– зміст замовлення – текстове повідомлення, яке визначає те, що повинен передати сервіс;

– відправник – адреса відправника повідомлення (зазвичай найменування сервісу).

Наприклад, так передає замовлення транспортувальник:

ПОСТАЧАЛЬНИК: замовлення: контролер STM32: ТРАНСПОРТ

При відвантаженні замовлення передається аналогічне повідомлення формату (рисунок 3.5):

Мітка одержувача
Замовлення
Зміст замовлення
Мітка відправника

Рисунок 3.5 – Відповідь на запит від сервісу при відвантаженні замовлення

В даному повідомленні:

– одержувач – адреса одержувача повідомлення (зазвичай найменування сервісу);

– замовлення – службове слово, яке визначає тип повідомлення;

– зміст_замовлення – текстове повідомлення, яке визначає те, що передає сервіс;

– відправник – адреса відправника повідомлення.

Наприклад, так передає замовлення транспортувальник:

ТРАНСПОРТ: замовлення: контролер STM32: ПОСТАЧАЛЬНИК

Як видно, в даному повідомленні (по відношенню до попереднього) транспорт і постачальник помінялися місцями.

Доставка замовлення підтверджується передачею такого типу повідомлень:

одержувач: замовлення: зміст_замовлення: відправник

В даному повідомленні:

– одержувач – адреса одержувача повідомлення;

– замовлення – службове слово, яке визначає тип повідомлення;

– зміст_замовлення – текстове повідомлення, яке визначає те, що передає сервіс;

– відправник – адреса відправника повідомлення.

Наприклад, так передає замовлення транспортувальник:

МАНІПУЛЯТОР : замовлення: контролер STM32: ТРАНСПОРТ

Завершує доставку замовлення повідомлення, в якому вказується команда на розвантаження транспортувалинка:

одержувач: розвантаження: відправник

В даному повідомленні:

– одержувач – адреса одержувача повідомлення;

– розвантаження – службове слово, яке визначає тип повідомлення;

– відправник – адреса відправника повідомлення.

Таким чином, у нас є вісім різних типів повідомлень, за допомогою яких сервіси можуть спілкуватися при виконанні їх обов'язків [6].

3.2 Разработка алгоритма работы віртуальної виробничої дільниці

Як було відзначено в попередніх розділах, в віртуальній виробничій дільниці застосовуються три незалежні програмні сервіси:

– промисловий маніпулятор;

– автоматизовано сховище;

– транспортувальник компонентів.

Розглянемо алгоритм роботи менеджеру повідомлень, який наведено на рисунку 3.6.

На даному рисунку видно, що менеджер має перелік всіх компонентів, необхідних для встановлення на друковану плату. Послідовно, рядок за рядком, зчитуються найменування необхідних в даний момент компонентів.

Отримані найменування передаються транспортувальнику, який отримує їх в автоматизованому сховищі [10].

Якщо необхідного компонента немає, то проводиться зчитування і доставка наступного за списком. Надалі промисловий маніпулятор чекатиме надходження всіх необхідних йому компонентів.

Коли все радіоелементи будуть доставлені, збірка друкованого модуля буде вважатися завершеною.

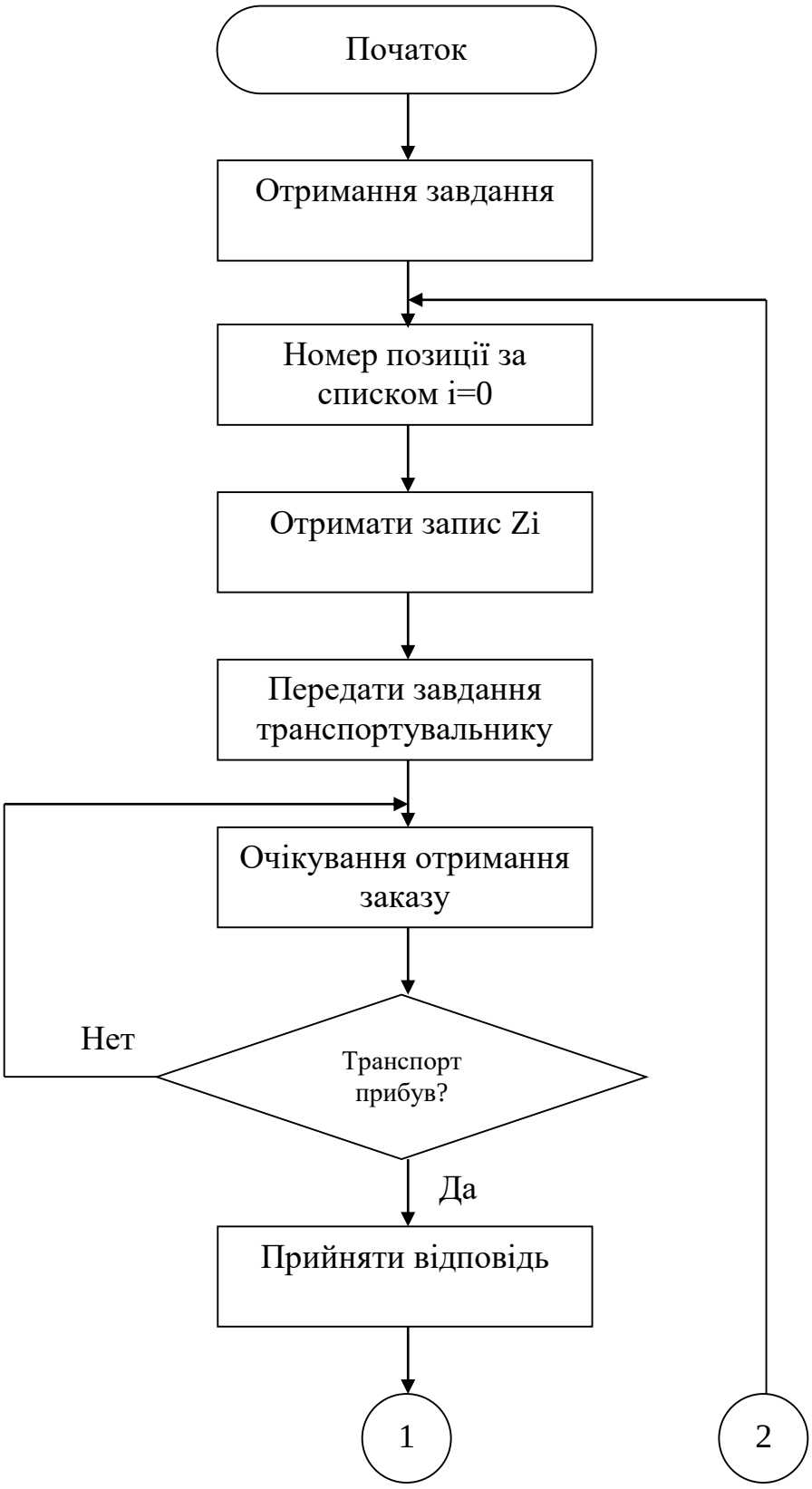


Рисунок 3.6 – Алгоритм роботи менеджеру віртуального виробництва

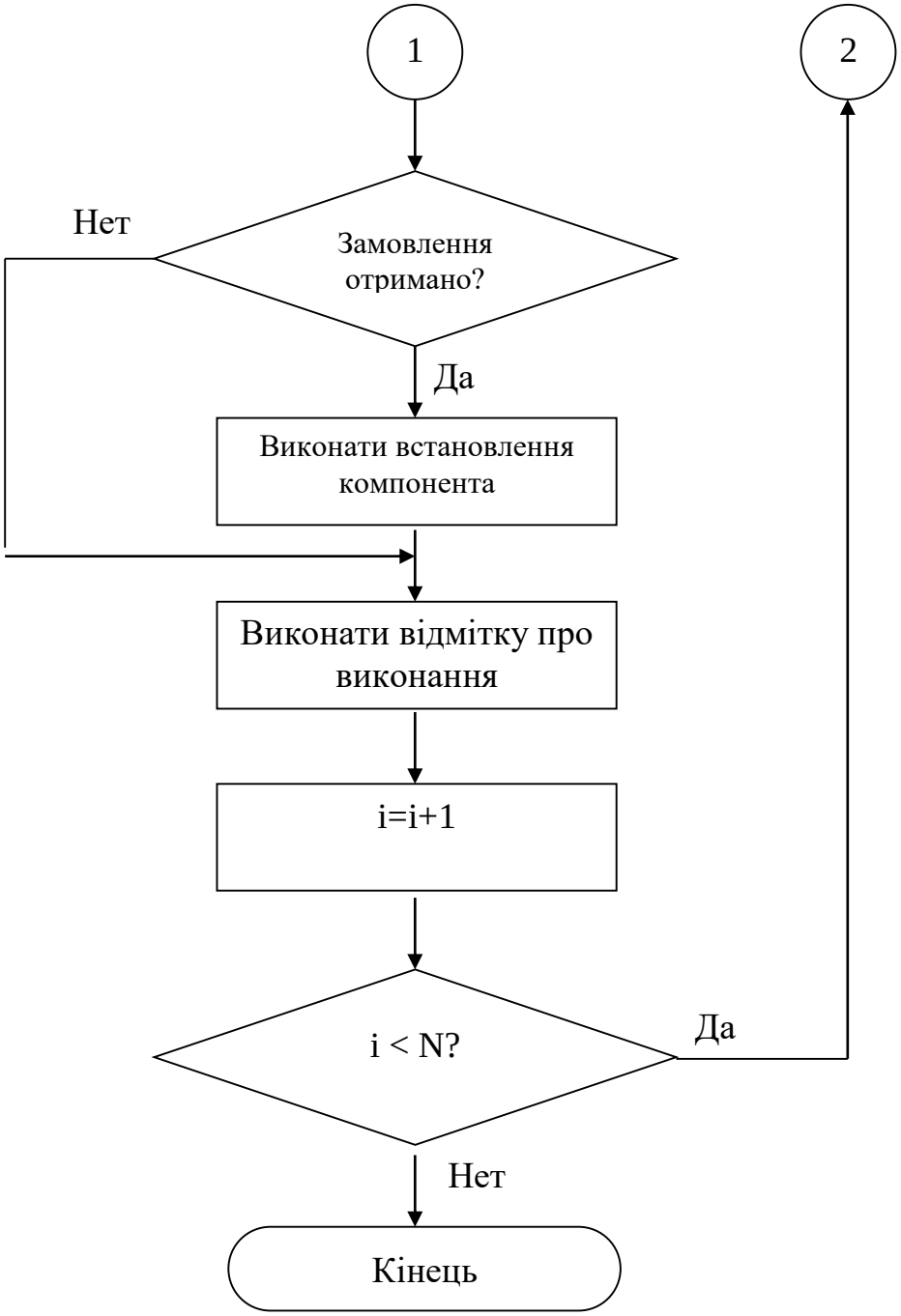


Рисунок 3.6, аркуш 2

3.3 Висновки по третьому розділу роботи

В результаті виконання третього розділу магістерської атестаційної роботи розроблена архітектура поєднання виробничих сервісів в моделі віртуального виробництва. Наведена схема взаємодії між програмними модулями, що виконують роль мікросервісів. Розглянуті формати повідомлень при взаємодії між програмними сервісами віртуального виробництва.

Розроблено алгоритм работы віртуальної виробничої дільниці. В віртуальній виробничій дільниці застосовуються три незалежні програмні сервіси: промисловий маніпулятор, автоматизовано сховище, транспортувальник компонентів.

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

4.1 Опис методу віртуалізації обчислень

На рисунку 4.1 показана структура програмного середовища. Керуюча програма (менеджер віртуального виробництва) взаємодіє з незалежними програмними сервісами. Кожен сервіс це окрема програма, яка пов’язана тільки з менеджером. Між собою програмні сервісі не мають зв’язків.

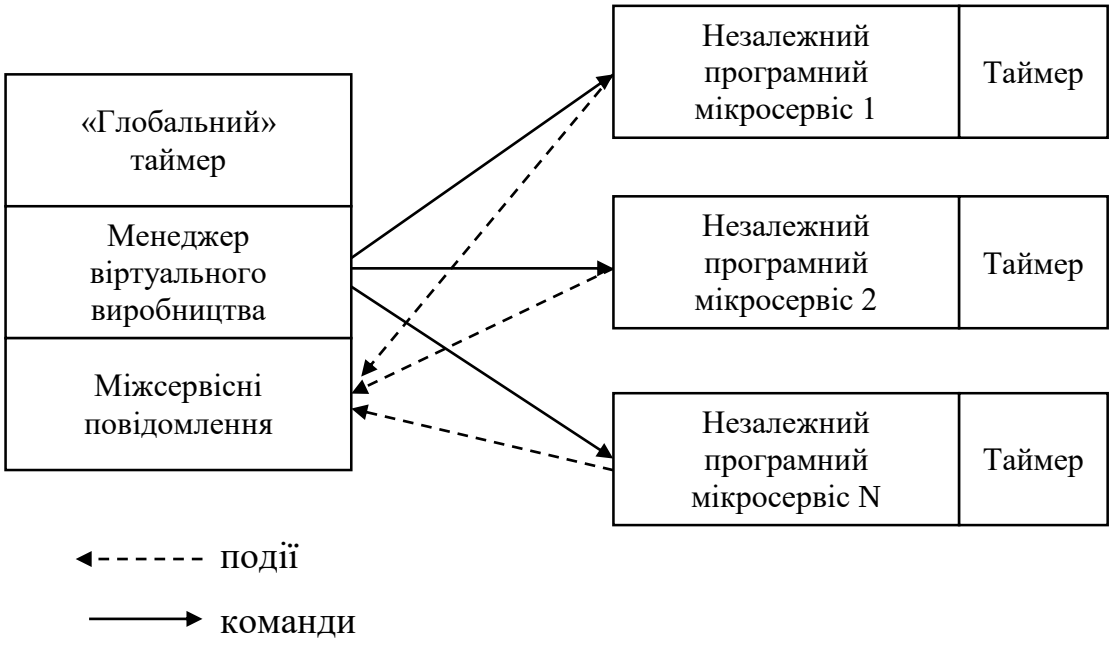


Рисунок 4.1 – Структура програмного середовища

Для реалізації виконання незалежних подій в віртуальному середовище використовується додаткова програма, яка виконує функцію менеджера для організації задачі отримання повідомлень від різних об'єктів в та розподілення запитів між сервісами. Така керуюча програма є необхідною частиною системи виконання будь-якої системи моделювання.

Керуюча програма взаємодіє з об'єктами відповідно до клієнт-серверних технологій обміну повідомленнями. Алгоритм роботи керуючої програми складається з наступних дій:

– активізація об'єктів для виконання подій, запланованих на поточне значення годин модельного часу. Видалення виконаних подій зі списку;

– включення в перелік подій нових подій, запланованих активними об'єктами. Подія включається в перелік подій разом зі значенням модельного часу, в яке ця подія має бути виконано в майбутньому;

– збільшення значення модельного часу, якщо на поточне значення модельного часу не залишилося невиконаних подій. Перехід в початок.

При такій організації виконання подій в віртуальній моделі виконуються послідовно, що й знайшло відображення в терміні «послідовне імітаційне моделювання». Послідовне імітаційне моделювання характерно для виконання моделі на автономному однопроцесорному комп'ютері. Разом з тим кілька подій в різних об'єктах можуть виконуватися в один і той же момент модельного часу, тому говорять, що об'єкти (процеси) виконуються квазіпаралельно.

4.2 Розробка програми для моделювання віртуального виробництва з використанням мікросервісної технології

Програма написана мовою програмування C# в середовище Monodevelop. Завдяки цьому, розроблений програмний засіб може бути завантажено для моделювання як на комп’ютерах під управлінням операційної системи Windows, так і Linux. Кожен мікросервіс розроблено у вигляді окремого програмного модуля, який завантажується окремо і працює незалежно один від одного.

Приклад сервісу «Автоматизоване сховище» показано на рисунку 4.2. Сервіс має логотип для ідентифікації на віртуальному полі, та поле з переліком деталей, що наразі значяться на зберіганні. Перелік деталей є інтерактивний та може доповнюватись в процесі роботи.

Рисунок 4.2 – Сервіс «Автоматизоване сховище»

Кількість деталей, що вказується після її назви змінюється в процесі роботи. Якщо кількість дійде до нуля, то така деталь буде вважатись відсутньою.

Кожен сервіс очікує повідомлення від менеджера віртуального виробництва. На початку роботи всі сервіси потрібно ініціювати. Для цього в них передбачено програмну функцію обробки broadcast-повідомлень. На рисунку 4.3 показано приклад такої функції.

```
if (attr[0]=="Усім")
{
    if (attr[1]=="хто_розташовано_х?")
    {
        int X = Convert.ToInt32(attr[2]);
        string Name = attr[3];
        int x, y;
        int w, h;
        this.GetPosition(out x, out y);
        this.GetSize(out w, out h);
        if ((X > x-5) && (X < (x+w+5))) MainWindow.message =
        String.Format("{0}:{1}:{2}:{3}:{4}:{5}:{6}",Name,
                "статус",x,w,"Постачальник",status,"СХОВИЩЕ");
    }

}
```

Рисунок 4.3 – Функція обробки broadcast-повідомлень

Якщо буде отримане повідомлення, в якому буде зазначено посилання на даний тип мікросервісу (в даному випадку це мітка «Постачальник»), то програмний сервіс повинен виконати свою головну функцію – обробити замовлення [7].

Обробка замовлення полягає в аналізі тіла повідомлення та виділені з нього переліку необхідних деталей, що треба видати, наприклад, транспортувальнику. Обраний тип деталі розшукується в загальному переліку деталей в пам’яті сервісу «Автоматизоване сховище». Якщо деталь знайдена, то формується відповідь де вказується найменування деталі та записується адреса відправника повідомлення. В даному випадку в повідомленні вказується, шо воно відправлено від «Постачальника».

Якщо перелік деталей порожній, або потрібної деталі немає, то в відповіді записується відмова в запиті [11].

Приклад функції обробки повідомлень, що надходять від замовника послуг наведено на рисунку 4.4.

```
if (attr[0]=="ПОСТАЧАЛЬНИК")
{
    if (attr[1]=="замовлення")
    {
        string Name = attr[3];
        string [] list = textview1.Buffer.Text.Split('\n');
        int i = 0;
        foreach (string str in list)
        {
            string [] zakaz = str.Split('=');
            int num = Convert.ToInt32(zakaz[1]);
            if (zakaz[0] == attr[2])
            {
                if (num > 0)
                {
                    num--;
                    zakaz[1] = num.ToString();
                    MainWindow.message = String.Format("{0}:{1}:{2}:{3}",Name,"замовлення",zakaz[0],"ПОСТАЧАЛЬНИК");
                }
                else
                {
                    MainWindow.message = String.Format("{0}:{1}:{2}:{3}",Name,"замовлення","ні","ПОСТАЧАЛЬНИК");
                }
                list[i] = zakaz[0]+"="+zakaz[1];
                break;
            }
            i++;
        }
        label2.Text= String.Format("");
        textview1.Buffer.Clear();
        string buf = "";
        for (int k=0;k<list.Length;k++)
        {
            buf = buf + list[k] + '\n';
        }
        textview1.Buffer.Text = buf;
    }

}
```

Рисунок 4.4 – Приклад функції обробки повідомлень, що надходять від замовника послуг

На рисунку 4.5 Показано зовнішній вигляд програмного сервісу «промисловий маніпулятор». Аналогічно попередньому сервісу, програма має функцію обробки broadcast-повідомлень та безпосередньо адресований даному сервісу.

Промисловий маніпулятор є представником замовника послуг. Для виконання задачі складання радіоелектронного модуля йому потрібні комплектуючі. В програмному сервісі є перелік необхідних деталей для виконання завдання та їх кількість. Перебираючи послідовно деталі із переліку він формує заявки, що передаються транспортному роботу, який в свою чергу, передає мікросервісу «Автоматизоване сховище».

Отримавши потрібні деталі транспортний робот доставляє їх до промислового маніпулятору, який виконує операції по встановленню компоненті на друковану плату.

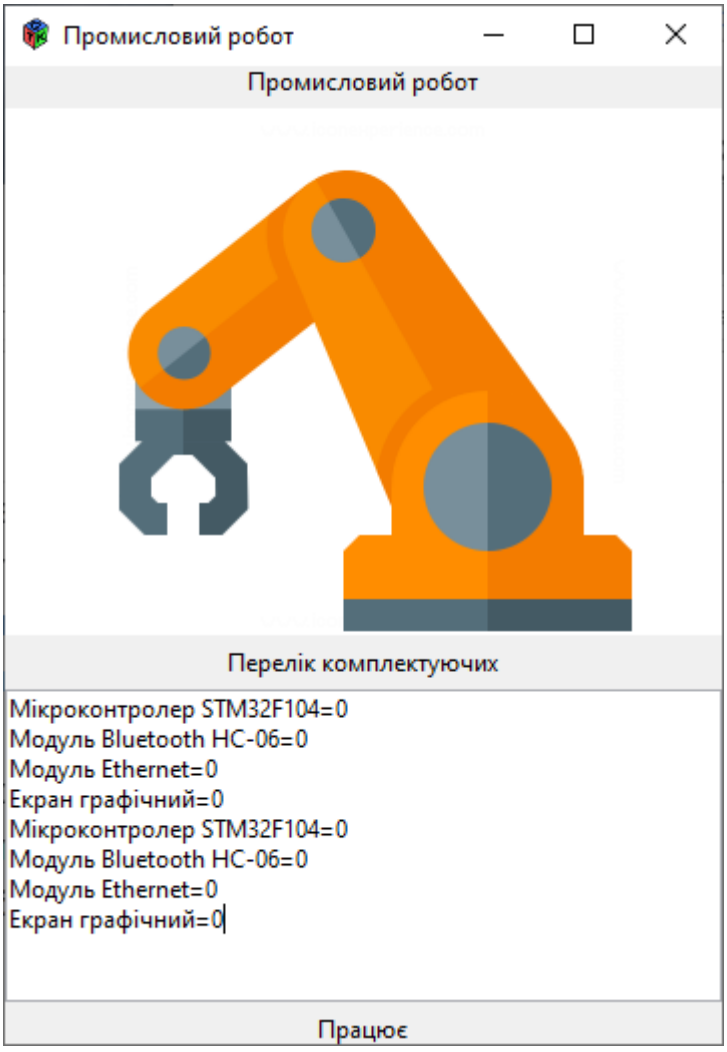


Рисунок 4.5 – Зовнішній вигляд програмного сервісу «промисловий маніпулятор»

При роботі програми навпроти кожної назви комплектуючих буде з'являтися ознака вдалою доставки компонента зі сховища – «ОК» або ознака помилки «Помилка». На початку роботи навпроти кожного компоненту знаходиться символ «0».

Основну роботу по переміщенню деталей виконує сервіс «Мобільний робот». Алгоритм його є найскладнішим серед інших сервісів. Розглянемо

кілька підпрограм [8], які викликаються при обробці прийнятих транспортом повідомлень. Підпрограма обробки відповіді про мікросервісів вигляде наступним чином:

```
if (attr[1]=="статус")
{
        int X = Convert.ToInt32(attr[2]);
        int W = Convert.ToInt32(attr[3]);
        string Type = attr[4];
        string Status = attr[5];
        string Name = attr[6];
        MainWindow.SaveAgent(X,W,Type,Status,Name);
}
```

В коді, наведеному вище, проводиться обробка повідомлень і розбір його полів. Виділяються тип, статус, найменування сервісу та його координати на полі віртуального виробництва.

Далі порівнюється тип сервісу («Постачальник» або «Замовник») і робляться відповідні записи в системних змінних:

```
        if (Type == "Постачальник")
        {
                Postavshik = true;
                Postavshik_X = X;
                Postavshik_W = W;
                Postavshik_S = Status;
                _MW.Postavshik_Location = Convert.ToString(X+W+13);
                _MW.Report = "Знайдено постачальник!";
        }
        if (Type == "Замовник")
        {
```

```
            Zakazchik = true;
            Zakazchik_X = X;
            Zakazchik_W = W;
            Zakazchik_S = Status;
            _MW.Zakazchik_Location = Convert.ToString(X-3);
            _MW.Report = " Знайдено замовник!";
        }
```

Після того, як будуть знайдені «Постачальник» і «Замовник», транспортувальник буде переведений з режиму пошуку в режим «робота»:

```
        if ((Postavshik) && (Zakazchik))
        {
            mode = "транспорт_замовник";
            _MW.Report = "Транспорт перейшов в режим <Робота>";
            if (Postavshik_X < Zakazchik_X) Direct = true;
            else Direct = false;
            MainWindow.message = String.Format("");
        }
```

Наступний код виконується, коли в команді передається службове слово "замовлення":

```
        if (attr[1]=="замовлення")
        {
            textview1.Buffer.Text = attr[2];
            mode = "транспорт_постачальник";
            _MW.Report = "Транспорт отримав завдання "+attr[2];
            Direct = false;
            MainWindow.message = String.Format(" ");
        }
```

При отриманні повідомлення від сховища, транспортувальник виконує його обробку за такою функцією [8]:

```
if (attr[1]=="замовлення")
{
	textview1.Buffer.Text = attr[2];
	mode = "транспорт_заказ";
	_MW.Report = "Транспорт отримав замовлення "+attr[2];
	Direct = true;
	MainWindow.message = String.Format(" ");
}
```

Після того, як транспортувальник доставить вантаж замовнику його необхідно розвантажити. Для цього був написаний наступний код:

```
if (attr[1]=="розвантаження")
{
	textview1.Buffer.Text = "";
	mode = "транспорт_замовник";
	Direct = true;
	MainWindow.message = String.Format(" ");
}
```

Режим пошуку сервісів «Постачальник» і «Замовник» описаний за допомогою наступного програмного коду:

```
if (mode == "пошук сервісів")
{
	if (Direct)    Xt++;
	else Xt--;
	if (Xt + w + 10 > (MainWindow.MaxX))
	{
		Direct = false;
	}
	if (Xt < MainWindow.MinX)
	{
		Direct = true;
```

```
}
this.Move(Xt,200);
if (Direct)    MainWindow.message =
                    String.Format("УСІМ:хто_розташовано x?:{0}:
                    ТРАНСПОРТ",Xt + w + 10);
          else MainWindow.message =
                    String.Format("УСІМ:кто_розташовано _x?:{0}:
                    ТРАНСПОРТ",Xt - 10);
}
```

В даному випадку видно, що агент «Транспортувальник» постійно виконує переміщення по полю з однієї крайньої координати в іншу, поки на ньому не будуть виявлені шукані агенти [8].

На рисунку 4.6 показано зовнішній вигляд пульту керування.

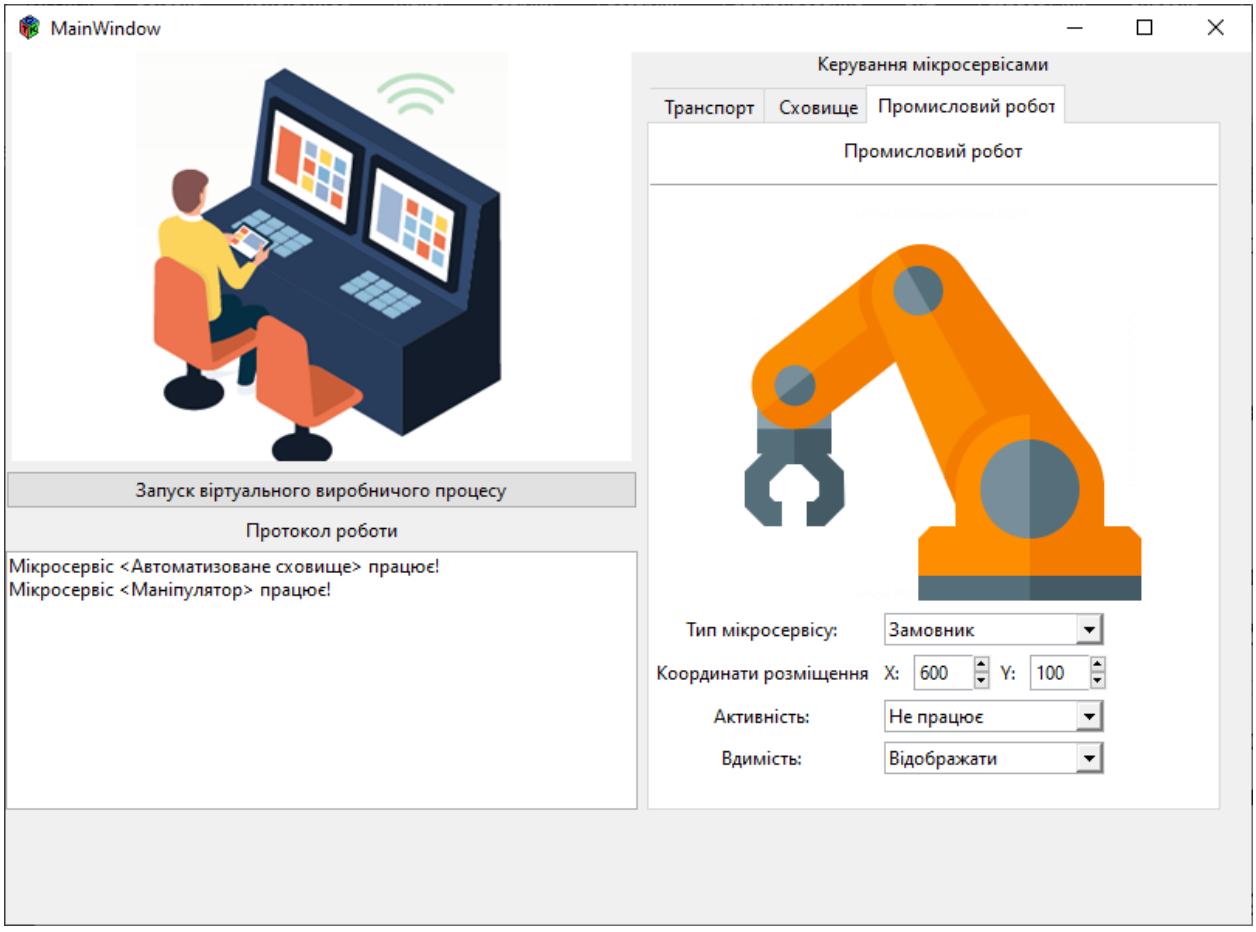


Рисунок 4.6 – Зовнішній вигляд диспетчерського пульту

Керування роботою віртуального виробництва відбувається з диспетчерського пульту, який виконує функцію менеджера повідомлень між програмними мікросервісами [9].

Дана програма має інтерфейс запуску програмних сервісів, обирання їх типу та задає координати їх розміщення на віртуальному полі виробничої ділянки.

В даному вікні є три закладки:

– транспорт;

– сховище;

– промисловий робот.

У кожній закладці розташовані поля введення для управління характеристиками сервісу:

– вибір типу сервісу;

– початкові координати розміщення сервісу на поле програмної моделі;

– знайдені координати постачальника і замовника;

– компоненти керуючої активністю і видимістю сервісу.

Початок роботи програмної моделі починається після натискання на кнопку «Запуск віртуального виробничого процесу».

Агент з'явиться в зазначених координатах після зміни значення властивості активність на «Включено».

За допомогою перемикачу «Активність» можна деактивувати мікросервіс. В такому стані він не буде відповідати на запити від інших програмних сервісів та буде знаходитись в режимі простою.

За допомогою перемикачу «Видимість» відбувається запуск нового екземпляру сервісу та включення його в роботу. На рисунку 4.7 показано робочій режим виконання завдань завантаженими та активованими мікросервісами [9].

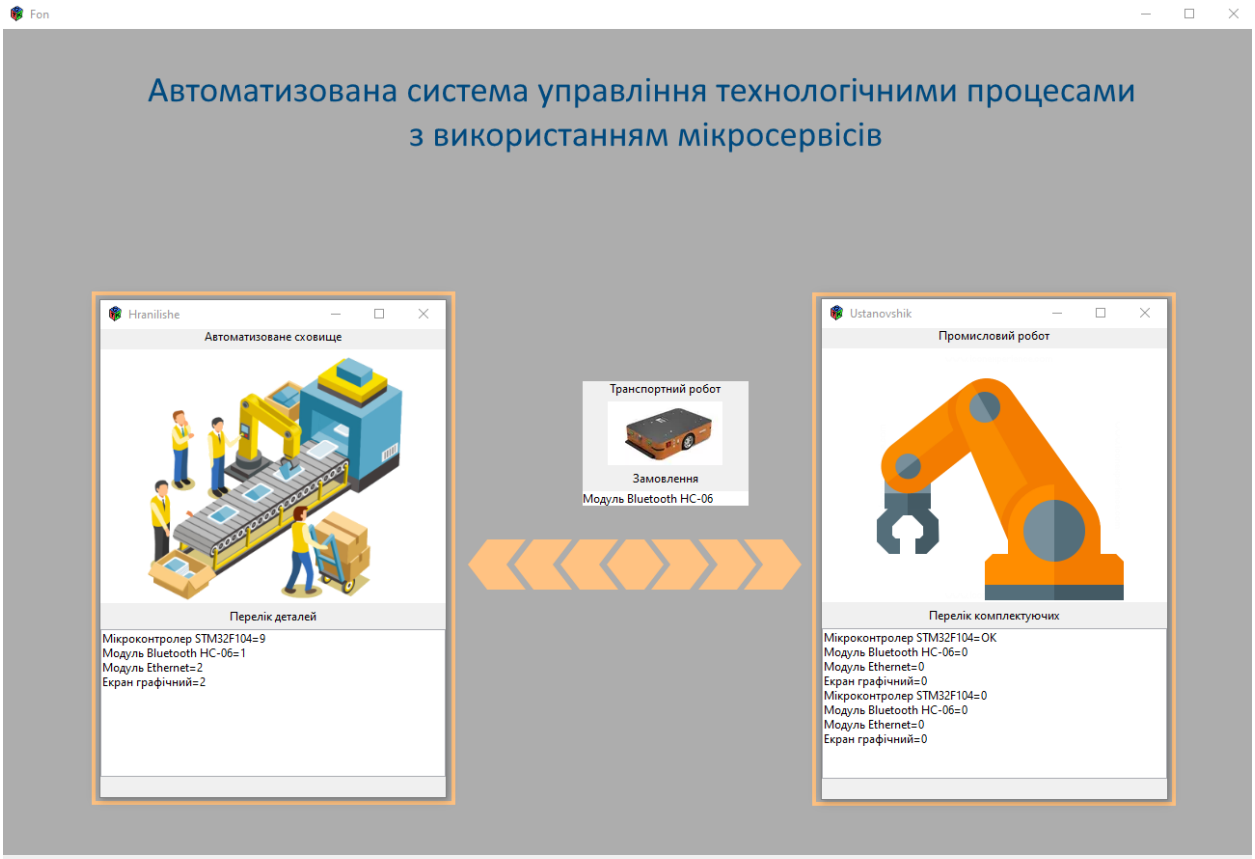


Рисунок 4.7 – Зовнішній вигляд завантаженої програми

4.3 Аналіз результатів моделювання

Диспетчерський пульт має поле для виводу протоколу випробувань. Кожне системне повідомлення дублюється в даному полі. На рисунку 4.8 показано приклад виведення повідомлень

В роботі проводилось моделювання автоматичної роботи програмних мікросервісів для виконання завдання послідовного встановлення наступного переліку деталей на друковану плату промисловим роботом-маніпулятором [12]:

– Мікроконтролер STM32F104;

– Модуль Bluetooth HC-064

– Модуль Ethernet;

– Екран графічний.

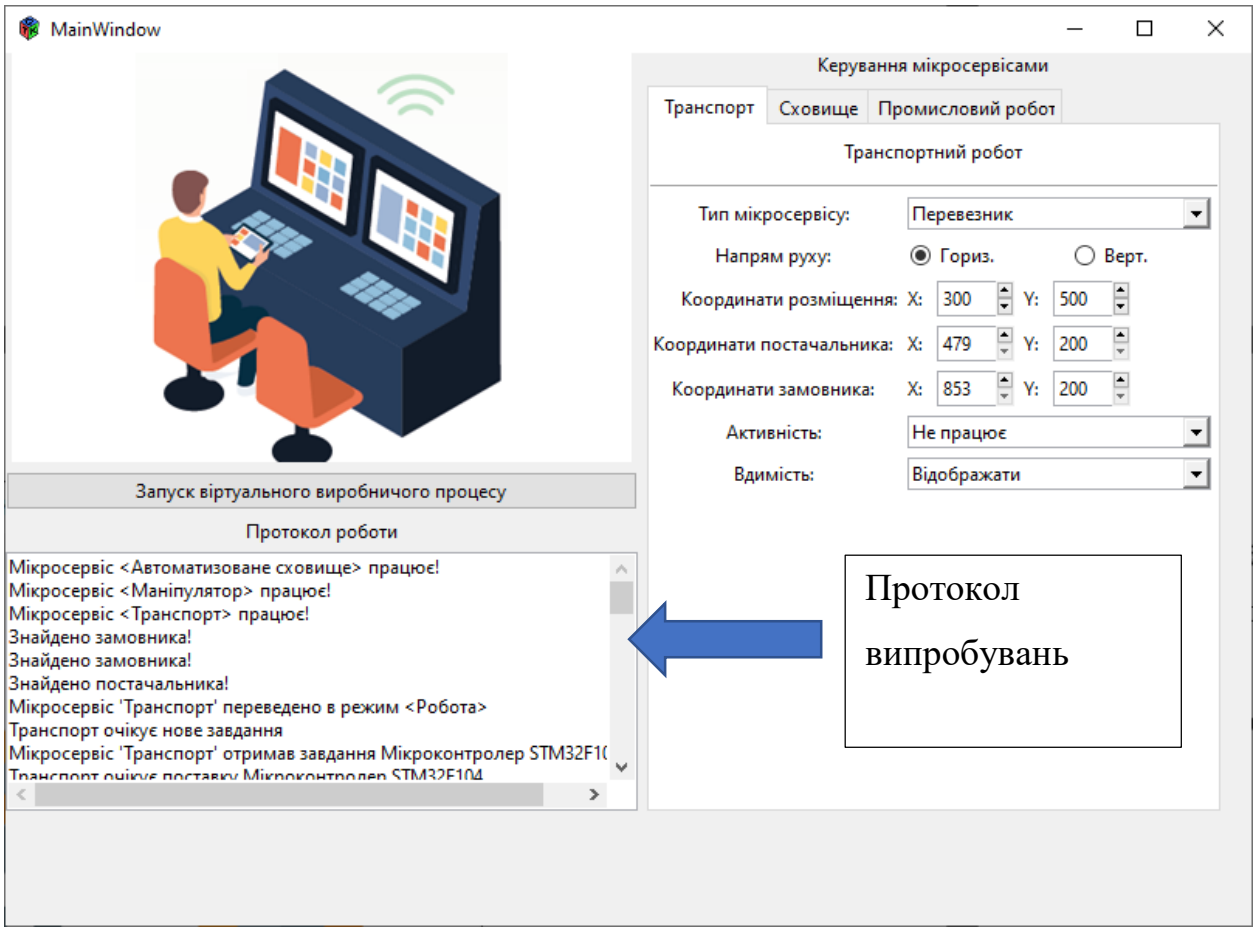


Рисунок 4.8 – Приклад виведення повідомлень

Перелік було продубльовано два рази та запущено процес автоматичної роботи. Нижче представлено повний звіт роботи всіх сервісів віртуального виробництва:

– Мікросервіс <Автоматизоване сховище> працює!;

– Мікросервіс <Маніпулятор> працює!;

– Мікросервіс <Транспорт> працює!;

– Знайдено замовника!;

– Знайдено замовника!;

– Знайдено постачальника!;

– Мікросервіс 'Транспорт' переведено в режим <Робота>;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Мікроконтролер STM32F;

– Транспорт очікує поставку Мікроконтролер STM32F104;

– Мікросервіс 'Транспорт' отримав замовлення Мікроконтролер STM32F104;

– Транспорт доставив замовлення Мікроконтролер STM32F104;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Модуль Bluetooth HC-06;

– Транспорт очікує поставку Модуль Bluetooth HC-06;

– Мікросервіс 'Транспорт' отримав замовлення Модуль Bluetooth HC-06;

– Транспорт доставив замовлення Модуль Bluetooth HC-06;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Модуль Ethernet;

– Транспорт очікує поставку Модуль Ethernet;

– Мікросервіс 'Транспорт' отримав замовлення Модуль Ethernet;

– Транспорт доставив замовлення Модуль Ethernet;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Екран графічний;

– Транспорт очікує поставку Екран графічний;

– Мікросервіс 'Транспорт' отримав замовлення Екран графічний;

– Транспорт доставив замовлення Екран графічний;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Мікроконтролер STM32F104;

– Транспорт очікує поставку Мікроконтролер STM32F104;

– Мікросервіс 'Транспорт' отримав замовлення Мікроконтролер STM32F104;

– Транспорт доставив замовлення Мікроконтролер STM32F104;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Модуль Bluetooth HC-06;

– Транспорт очікує поставку Модуль Bluetooth HC-06;

– Мікросервіс 'Транспорт' отримав замовлення Модуль Bluetooth HC-06;

– Транспорт доставив замовлення Модуль Bluetooth HC-06;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Модуль Ethernet;

– Транспорт очікує поставку Модуль Ethernet;

– Мікросервіс 'Транспорт' отримав замовлення Модуль Ethernet;

– Транспорт доставив замовлення Модуль Ethernet;

– Транспорт очікує нове завдання;

– Мікросервіс 'Транспорт' отримав завдання Екран графічний;

– Транспорт очікує поставку Екран графічний;

– Мікросервіс 'Транспорт' отримав замовлення Екран графічний;

– Транспорт доставив замовлення Екран графічний;

– Транспорт очікує нове завдання.

В результаті роботи були отримані та встановлені всі компоненти, що підтверджено звітом від програмних сервісів «Промисловий маніпулятор» та «Автоматизоване сховище» на рисунку 4.9.

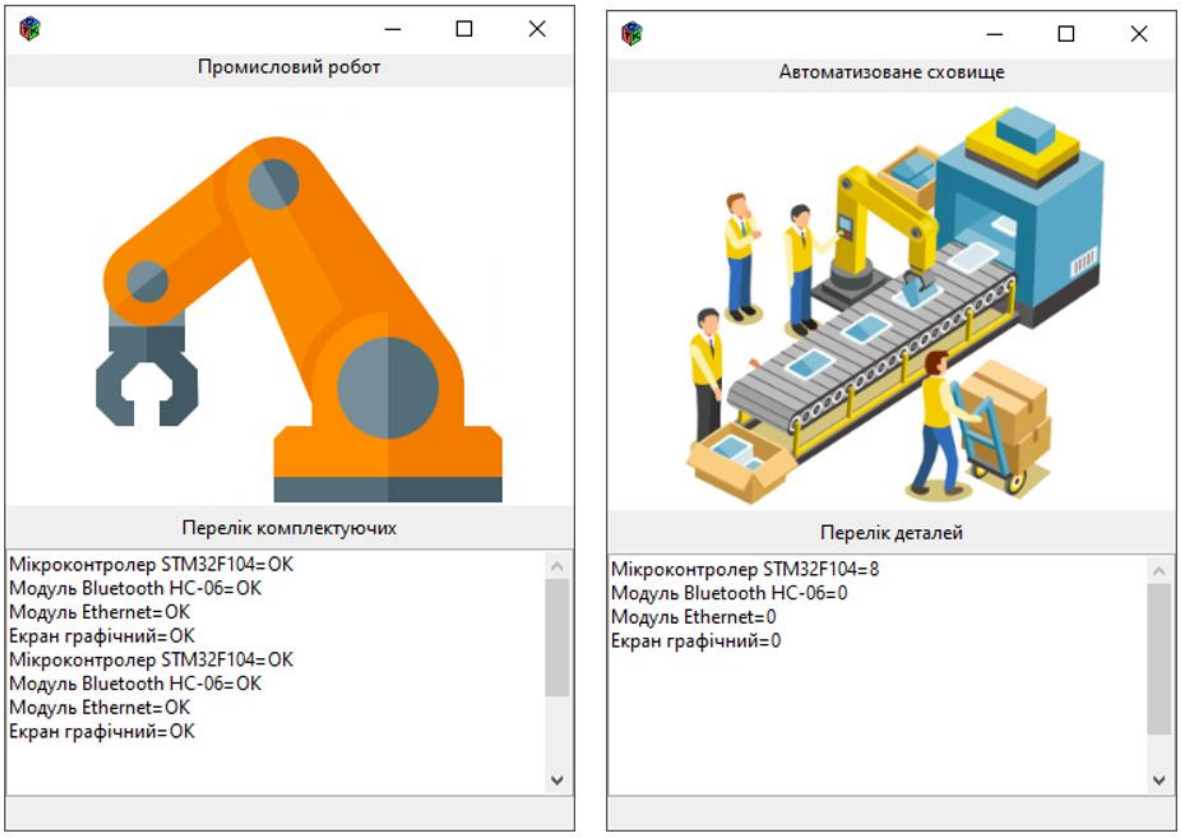


Рисунок 4.9 – Результат роботи програмних мікросервісів

4.4 Висновки по четвертому розділу роботи

В результаті виконання четвертого розділу магістерської атестаційної роботи виконана практична реалізація запропонованого методу керування технологічними процесами на виробництві з використанням мікросервісів. Описано метод віртуалізації обчислень за допомогою обчислювальних засобів.

Програма написана мовою програмування C# в середовище Monodevelop. Завдяки цьому, розроблений програмний засіб може бути завантажено для моделювання як на комп’ютерах під управлінням операційної системи Windows, так і Linux. Кожен мікросервіс розроблено у вигляді окремого програмного модуля, який завантажується окремо і працює незалежно один від одного.

Керування роботою віртуального виробництва відбувається з диспетчерського пульту, який виконує функцію менеджера повідомлень між програмними мікросервісами.

В результаті роботи отримані та встановлені всі компоненти, що підтверджено звітом від програмних сервісів. Таким чином ми підтвердили правильність теоретичних досліджень[13].

ВИСНОВКИ

В результаті виконання першого розділу магістерської атестаційної роботи виконали аналіз особливостей побудови розподілених систем управління технологічними процесами. Виконали порівняння централізованої і розподіленої архітектури АСУ ТП. Наведені переваги та недоліки кожної із зазначених архітектур.

Проведено аналіз застосування різних методик для управління розподіленими динамічними системами. Розглянуто принципи проектування автоматизованої системи управління технологічними процесами в розподілених виробничих системах.

В результаті виконання другого розділу магістерської атестаційної роботи виконали аналіз можливостей застосування мікросервісів в розподілених системах управління технологічними процесами. Дано визначення мікросервісу та його властивостям і поведінці при взаємодії з іншими об’єктами виробничої системи.

Описана методика обміну повідомленнями між мікросервісами в технологічній системі. Визначено, що механізм локалізації міжсервісних взаємодій складається з чотирьох фаз: фази моніторингу, фази розподілу завдань, фази взаємодії та фази перерозподілу обов'язків.

В результаті виконання третього розділу магістерської атестаційної роботи розроблена архітектура поєднання виробничих сервісів в моделі віртуального виробництва. Наведена схема взаємодії між програмними модулями, що виконують роль мікросервісів. Розглянуті формати повідомлень при взаємодії між програмними сервісами віртуального виробництва.

Розроблено алгоритм работы віртуальної виробничої дільниці. В віртуальній виробничій дільниці застосовуються три незалежні програмні сервіси: промисловий маніпулятор, автоматизовано сховище, транспортувальник компонентів.

В результаті виконання четвертого розділу магістерської атестаційної роботи виконана практична реалізація запропонованого методу керування технологічними процесами на виробництві з використанням мікросервісів. Описано метод віртуалізації обчислень за допомогою обчислювальних засобів.

Програма написана мовою програмування C# в середовище Monodevelop. Завдяки цьому, розроблений програмний засіб може бути завантажено для моделювання як на комп’ютерах під управлінням операційної системи Windows, так і Linux. Кожен мікросервіс розроблено у вигляді окремого програмного модуля, який завантажується окремо і працює незалежно один від одного.

В результаті роботи отримані та встановлені всі компоненти, що підтверджено звітом від програмних сервісів. Таким чином ми підтвердили правильність теоретичних досліджень.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. ДСТУ 3008–2015. Звіти у сфері науки і техніки. Структура і правила оформлення. Документація. – Введ. 2015-06-22. - К.: Держстандарт України, 2015. - 31 с.

2. Методичні вказівки з дипломного проектування для студентів усіх форм навчання спеціальностей 7.05090202 «Автоматизовані комплекси радіоелектронних виробництв», 7.05090203 "Інтелектуальні технології мікросистемної радіоелектронної техніки" [Текст] : методичні вказівки / В. А. Палагін, С. П. Новоселов, Є. А. Разумов-Фризюк, І. В. Жарікова. МОНМС України, ХНУРЕ. - Харків: ХНУРЕ, 2012. – 68 с.

3. CramonJ. Microservices: It’s not (only) the size that matters, it’s (also) how you use them. [Електронний ресурс]. — Режим доступу: http://mng.bz/zQ2a. – 23.10.2020 р.

4. Fowler M. MicroservicePrerequisites [Електронний ресурс]. — Режим доступу: http://martinfowler.com/bliki/MicroservicePrerequisites.html. – 20.10.2020 р.

5. Monolith First [Електронний ресурс]. — Режим доступу: http:// martinfovvler.com/bliki/MonolithFirst.html. – 14.10.2020 р.

6. Lewis J., Fowler M. Microservices [Електронний ресурс]. — Режим доступу: http://martinfowler.com/articles/microservices.html. – 21.10.2020 р.

7. Newman S. Building Microservices: Designing Fine-Grained Systems. — O’Reilly Media, 2015.

8. Pattern: Backends for Frontends [Електронний ресурс]. — Режим доступу: http://samnewman.io/patterns/architectural/bff. – 21.10.2020 р.

9. Tilkov S. Don’t start with a monolith Електронний ресурс]. — Режим доступу: http://martinfowler.com/articles/dontstart-monolith.html. – 21.10.2020 р.

10. Novoselov, Sergii & Sychova, Oksana. Using Wireless Technology for Managing Distributed Industrial Automation Objects within the Concept of Industry

// IEEE International Scientific-Practical Conference: Problems of Infocommunications Science and Technology, PIC S and T – 2019 – P. 580-584.

11. Технічні засоби автоматизації: Підручник / І.Ш. Невлюдов, А.О. Андрусевич, О.І. Филипенко, Н.П. Демська, С.П. Новоселов. – Кривий Ріг : Криворізький коледж НАУ, 2019. – 366 с.

12. Невлюдов І. Ш. Людино-машинний інтерфейс в технічних засобах автоматизації: Навчальний посібник / І. Ш. Невлюдов, О. І. Филипенко, Б. О. Шостак. – Харків : «ХТМТ», 2019. – 244 с.

13. Основи наукових досліджень: Навч. посібник / І.Ш. Невлюдов, Ю.М. Олександров, А.О. Андрусевич, О.О. Чала. – Кривий Ріг: Криворізький коледж НАУ, 2019. – 396 с.

14. Коммуникации в системе управления [Електронний ресурс]. — Режим доступу: https://psyera.ru/kommunikacii-v-sisteme-upravleniya_7383.htm. – 24.10.2020 р.

15. Стратегії розвитку сервіс-орієнтованих систем у Хмарному середовищі [Електронний ресурс]. — Режим доступу:https://ela.kpi.ua/bitstream/123456789/24435/1/Petrenko_diss.pdf – 24.10.2020 р.

16. Диспетчеризація виробництва [Електронний ресурс]. — Режим доступу:https://studopedia.com.ua/1_209526_dispetcherizatsiya-virobnitstva.html. – 24.10.2020 р.

17. Кіберфізичні системи та їх програмне забезпечення [Електронний ресурс]. — Режим доступу: http://science.lpnu.ua/sites/default/files/journal-paper/2018/sep/14522/0064.pdf. – 24.10.2020 р.