НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Інженерно-хімічний факультет
(повна назва інституту/факультету)

Кафедра автоматизації хімічних виробництв
(повна назва кафедри)

«На правах рукопису»
УДК 004.658/2

«До захисту допущено»
Завідувач кафедри
_____ А.І.Жученко
(підпис) (ініціали, прізвище)
“___”_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 151 – Автоматизація та комп’ютерно-інтегровані технології
(код і назва)

на тему: *Багаторівнева система управління виробництвом безперервного хлорування бензолу*

Виконав: студент VI курсу, групи ЛА-61м:
Усманов Дмитро Олегович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник: *к.т.н., доцент Ковалюк Д.О.* _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інженерно-хімічний факультет
(повна назва)

Кафедра автоматизації хімічних виробництв
(повна назва)

Рівень вищої освіти – другий (магістерський)

Спеціальність 151- Автоматизація та комп'ютерно-інтегровані технології
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Жученко
(підпис) (ініціали, прізвище)
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Усманову Дмитру Олеговичу
(прізвище, ім'я, по батькові)

1. Тема дисертації *Багаторівнева система управління виробництвом безперервного хлорування бензолу* _____,
науковий керівник дисертації к.т.н. доцент Ковалюк Д. О._____ ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «21» березня 2018 р. №979-с

2. Термін подання студентом дисертації 18 травня 2018 р.

3. Об'єкт дослідження *технологічний процес виробництва хлорбензолу* _____

4. Предмет дослідження Багаторівнева система управління виробництвом безперервного хлорування бензолу _____

5. Перелік завдань, які потрібно розробити розробка математичної моделі хлоратора, синтез ПІ, ПІД регуляторів. Синтез ЛКГ регулятора. Розробка інтеграції з прикладним програмним інтерфейсом. Розробка сховища даних. Розробка рішення візуалізації даних. Розробка стартап проекту. _____

6. Орієнтовний перелік ілюстративного матеріалу Рисунки-слайди в програмі PowerPoint для презентації матеріалів з дисертації _____

7. Орієнтовний перелік публікацій Усманов Д.О., Ковалюк Д.О. Багаторівнева система керування виробництвом безперервного хлорування бензолу – Автоматизація та комп'ютерно-інтегровані технології [Текст]: Матеріали V

Міжнародної науково-практичної конференції молодих учених, аспірантів і студентів (АКІТ-2018). - Київ, 11-12 квітня 2018 р. – К.: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2018. – С. 131-132.

8. Дата видачі завдання 29 березня 2018 року_____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз виробництва безперервого хлорування бензолу	15.03.2018	
2	Архітектура багаторівневої системи управління	6.04.2018	
3	Математичне моделювання хлоратора	14.04.2018	
4	Синтез та дослідження системи керування	19.04.2018	
5	Розробка MES системи керування	01.05.2018	
6	Розробка стартап проекту	05.05.2018	
7	Оформлення матеріалів до магістерської дисертації	11.05.2018	

Студент _____ Усманов Д.О.
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Ковалюк Д.О.
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Основною задачею магістерської дисертації є створення та дослідження багаторівневої системи управління виробництвом безперервного хлорування бензолу: система з ПІД та ПІ регулятором, система з ЛКГ регулятором, MES системи згідно з стандартом ISA-95.

Відповідно до теми магістерської дисертації надруковано 2 тез конференцій.

Пояснювальна записка до магістерської дисертації має обсяг 122 сторінки та містить 54 рисунока, 30 таблиць, 26 літературних джерела.

Результати роботи можуть бути використані та впроваджені в системах керування виробництвом безперервного хлорування бензолу. Розроблені в програмному пакеті Matlab моделі та схеми систем керування можуть бути використані в курсах «Спеціальні розділи теорії автоматичного керування». Розроблені методи отримання та інтеграції даних можуть бути використані в курсах «Бази даних»

Ключові слова: багаторівнева система, ПІ регулятор, ПІД регулятор, ЛКГ регулятор, контур керування, об’єкт керування, математична модель, статична характеристика, канал збурення, канал керування, база даних, прикладний програмний інтерфейс, динамічна характеристика, стартап.

ABSTRACT

The major goal of the master's dissertation is the creation and research of a multilevel system for producting continuous chlorination of benzene, which is the system with PID and PI regulator, a system with LQG regulator, and also MES systems in accordance with the ISA-95 standard.

According to the topic of the master's dissertation, two thesis of the conferences were published.

The results of the work can be used and implemented in the systems controlling the production of continuous chlorination of benzene. Developed with the help of Matlab's software package, models and drafts of the systems can be used in the course named «Special sections of the theory of automatic control». Methods created for obtaining and integrating data can be exploited in the courses of databases.

Keywords: multilevel system, PI regulator, PID regulator, LCH regulator, control loop, control object, mathematical model, static characteristic, channel rotation, channel management, database, application program interface, dynamic characteristics, start.

Зміст

Перелік скорочень, умовних позначень

АСУТП− автоматизована система управління технологічним процесом.

ПІД – пропорційно-інтегрально-диференціальний регулятор.

ПІ – пропорційно-інтегральний регулятор.

ТП – технологічний процес.

ТОК – технологічний об’єкт керування.

АФХ – амплітудно-фазова характеристика

ОК – об’єкт керування

LQG – лінійно квадратичний Гауссовий регулятор

Ц-Н – метод Циглера-Нікольса

API – прикладний програмний інтерфейс

БД – база даних

MES – багаторівнева система

JSON – текстовий формат обміну даними між комп'ютерами.

SQL – мова програмування

Вступ

Дана магістерська дисертація присвячена вивченню процесу одержання хлорбензолу безперервним хлоруванням бензолу.

Одним із найпоширеніших хімічних розчинників є бензол. Значну частину отримуваного бензолу використовують для синтезу інших продуктів, таких як: етилбензол, кумол, циклогексан, тощо. Крім того бензол входить до складу бензину. Хлорбензол – є важливим органічним розчинником, окрім того, він застосовується в органічному синтезі, наприклад синтезі пестицидів. Також використовується в виробництві фенолу та фармакології.

Відповідно до сучасних даних виробництво хлорбензолу майже шість млн. тонн і цьогорічні масштаби виробництва її зростають. Тому перед сучасною хімічною промисловістю поставлено завдання розробити методи полегшення й шляхи інтенсифікації отримання хлорбензолу, поліпшення якості продукту, зменшення витрат на його виробництво.

Новизною даного дипломного проекту є створення багаторівневої системи керування автоматизації процесу одержання хлорбензолу безперервним хлоруванням бензолу. Також для якісної роботи системи керування обрані такі параметри регуляторів, які забезпечують задані показники якості системи.

1. Аналіз процесу виробництва хлорбензолу

1.1. Фізичні і хімічні властивості хлорбензолу

Одним із найпоширеніших хімічних розчинників є бензол. Значну частину отримуваного бензолу використовують для синтезу інших продуктів, таких як: етилбензол, кумол, циклогексан, тощо. Крім того бензол входить до складу бензину.

Виробництво кам'яновугільних барвників, штучних дубителей, пластмас, запашних речовин тощо, вимагають розчинників смол. В якості таких розчинників виступають хлоровані вуглеводні ароматичного ряду, зокрема хлорбензол. Хлорбензол – важливий напівпродукт в отриманні анілінових фарбників. Його отримують безперервним способом каталітичного хлорування бензолу*. За якість хлорбензолу відповідає процес хлорування, який відбувається у хлораторі. Хлоратор – апарат для дозування хлору і приготування його водного розчину

Хлорбензол – ароматичне з'єднання, котре має формулу C_6H_5Cl безбарвна, летюча рідина з миндальним запахом, не розчинна в воді і реагує з більшістю органічних розчинників. Хлорбензол був відкритий в 1851 році як продукт реакції фенола з хлоридом фосфору(V) і так він зазвичай і отримується в лабораторіях, в промисловості ж хлорбензол отримується хлоруванням бензолу. Властивості хлорбензолу: температура кипіння 132 градуси Целсія. Не дивлячись на малу подвижність галоїда, хлорбензол має широке використання в якості вихідної речовини для синтезу різноманітних органічних сполук. В промисловості частіше всього для отримання хлорбензолу використовують безперервне хлорування бензолу. Він має властивості ароматичних сполук, добре розчиняється в органічних розчинниках, використовується у виробництві лікарських препаратів, фарб, пестицидів, барвників, тощо. Хлорбензол викликає головний біль, сонливість, запаморочення, розлад травлення, при попаданні на шкіру – екзему, найімовірніший шлях враження – інгаляційний. Хлорбензол є депресантом ЦНС. Молекулярна маса – 112.55 г/моль.

1.2. Методи одержання хлорбензолу

У промисловості хлорбензол отримують безперервним хлоруванням бензолу за способом, розробленим Б. Є. Беркманом і складається з чотирьох стадій: безперервного хлорування бензолу, промивання продуктів хлорування, ректифікації продуктів хлорування, нейтралізації газів, що відходять кислотного характеру.

Безперервне хлорування бензолу проводять в хлораторі, що представляє собою трубу, заповнену перемішаними сталевими і керамічними кільцями розміром 25 х 25 мм. У верхній частині хлоратора є сепараційний обсяг, не заповнений насадкою. Вступний на хлорування бензол піддається попередній азеотропній осушці, що здійснюється безперервним методом в сталевих ректифікаційних колонах, заповнених насадкою з керамічних кілець. Бензол і хлор подають прямотоком в нижню частину апарата. Подачу реагентів регулюють таким чином, щоб повністю використовувати хлор, і щоб температура в апараті підтримувалася на рівні 76 - 83°С. Пари бензолу і, в невеликому ступені, хлорбензолу разом з виділеним HCl потрапляють в теплообмінник, конденсуються, відокремлюються від HCl і після осушення повертаються в процес.

Безперервне хлорування бензолу здійснюється в реакторах колонного типу, заповнених залізними кільцями.

Реактори виконані з вуглецевої сталі, футерованной діабазовий плиткою. Вступний на хлорування бензол піддається попередній азеотропному осушуванню, здійснюваної безперервним методом у сталевих ректифікаційних колонах, заповнених насадкою з керамічних кілець.

При безперервному хлоруванні бензолу великі труднощі викликає необхідність відводу тепла, що виділяється при цій екзотермічної реакції.

За схемою безперервного хлорування бензолу при кипінні реакційної маси (розроблена за пропозицією радянського інженера Б. Є. Беркмана) відвід тепла реакції здійснюється за рахунок випаровування надлишкового бензолу. При цьому відпадає необхідність в охолоджувальних пристроях і значно зменшується обсяг хлораторів.

Цей спосіб, широко застосовуваний в даний час, є значним удосконаленням виробництва хлорбензолу.

При цьому відпадає необхідність в охолоджувальних пристроях і значно зменшується обсяг хлораторів. Цей спосіб, широко застосовуваний в даний час, є значним удосконаленням виробництва хлорбензолу. Також реакцію закінчують, коли в суміші залишається ще близько 50% непрореагованого бензолу.

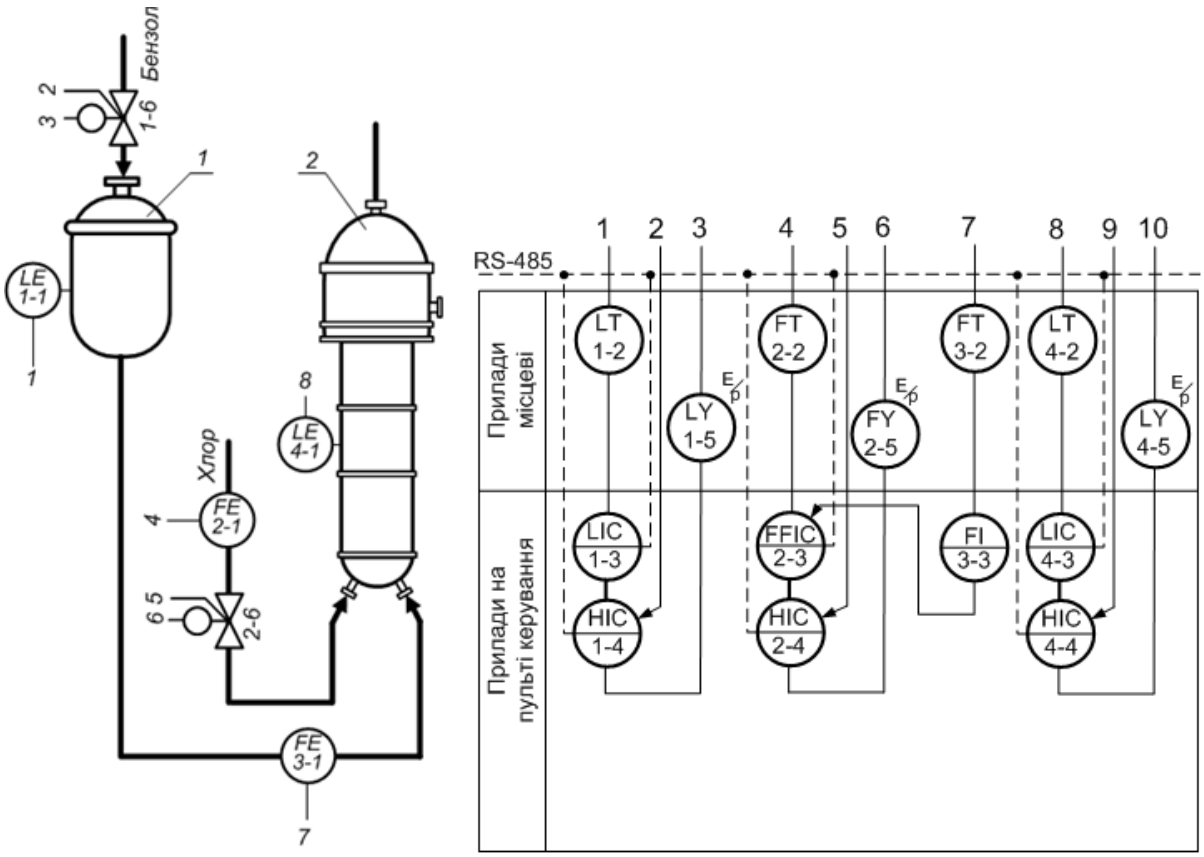


Рис. 1.1. – Схема автоматизації хлоратора: 1 – бачок для підтримання постійного рівня; 2 – хлоратор.

- контур 1 забезпечує контроль рівня бензолу, який надходить в бачок для підтримання рівня;

Розроблена схема автоматизації забезпечує дозування газоподібного хлору на заданому технологічному рівні і приготування хлорованого бензолу високої якості у головному апараті процесу – хлораторі.

В процесі хлорування бензолу відбувається виділенні тепла, при температурі 80-85 С, щоб утворювалось менше поліхлориду. Відведення тепла відбувається за рахунок випаровування надлишкового бензолу у верхній частині апарату.

Реакція відбувається за рівнянням:

$$C_6H_6 + Cl_2 \rightarrow C_6H_5Cl + HCl + 234{,}46\ \frac{\text{кДж}}{\text{кг} \cdot \text{моль}}$$

Схема автоматизації ректифікації процесу безперервного хлорування бензолу наведена на кресленні. Зменшена копія функціональної схеми автоматизації безперервного хлорування бензолу представлена на рис.1.2

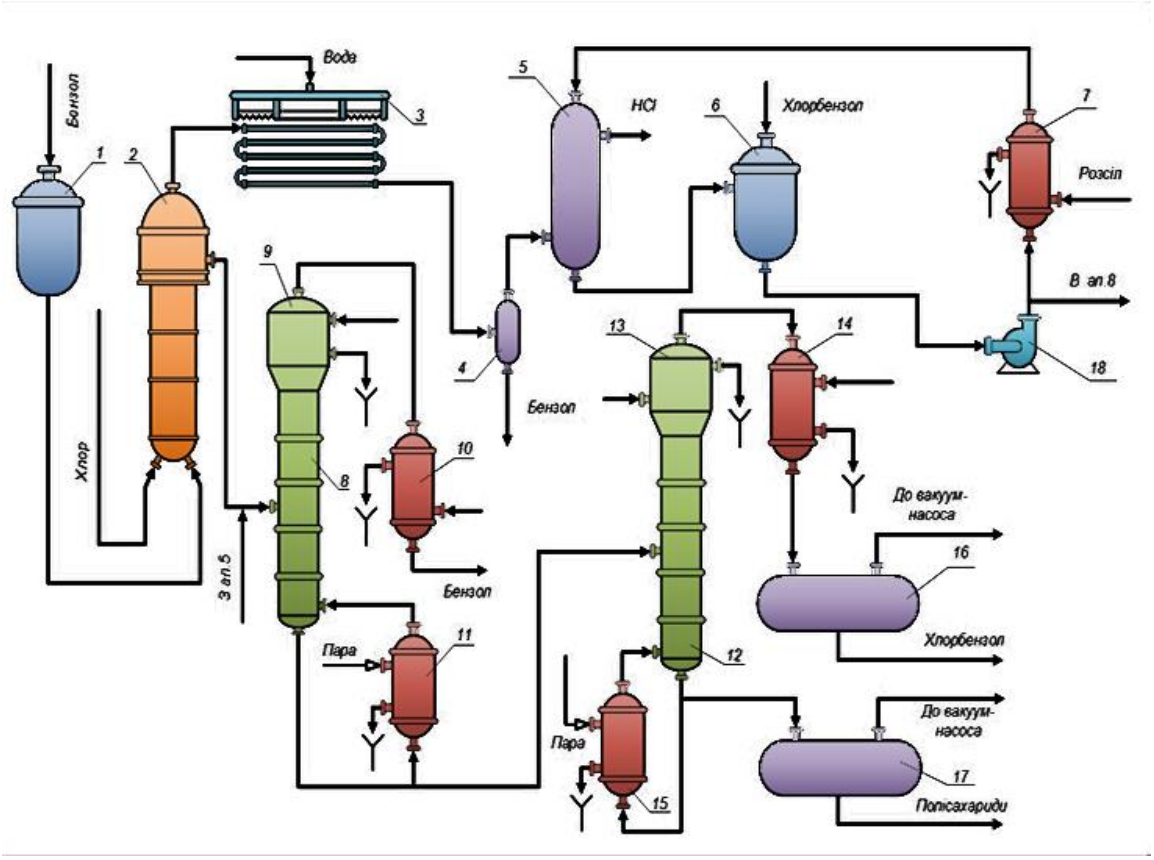


Рис. 1.2. – Функціональна схема автоматизації безперервного хлорування бензолу

Процес безперервного хлорування бензолу потребує:

– автоматизації процесу;

– необхідно забезпечити безперервне контролювання і реєстрацію усіх параметрів у ході виконання процесу безперервного хлорування бензолу;

– забезпечити наявність контурів сигналізації і технологічних блокувань;

– максимально зменшити імовірність;

Пристрої автоматизації у даному процесі відіграють важливу роль, адже від них значною мірою залежить те, наскільки якісно і ефективно буде працювати підприємство.

Тому можна виділити основні типи пристроїв автоматизації:

FE - вимірювачі витрати, вимірюють витрату води і повітря.

TE– вимірювачі температури.

PT – вимірювач тиску.

FT, LT, TT – пристрої автоматизації для передачі сигналу на відстань;

FIC, LIC, TIC – пристрої для автоматичного регулювання технологічного параметру (витрати, рівня та температури);

HIC – пристрої для ручного контролю технологічного параметру;

QIAS, PIAS – пристрої на пульті керування для показу і реєстрації параметрів, а також для сигналізації високого або низького рівня контрольованого параметру;

HL – сигнальні лампочки;

KM – електромагнітне реле (контактор магнітний);

SB – кнопки вмикання/вимикання електродвигунів;

SA – кнопки відключення електромотора від фази електричної мережі

1.3. Постановка задач автоматизації

Технологічний процес безперервного хлорування бензолу є складним об’єктом, для ефективного функціонування якого необхідно підтримувати заданий режим. Для попередження відхилень від встановлених норм потрібен постійний контроль параметрів за допомогою контрольно-вимірювальних і регулюючих приладів, об’єднаних в систему керування. Саме тому ми маємо розробити якісну математичну модель невеликого порядку та синтезувати для неї: ПІ регулятор та ПІД, після чого зробити їх порівняння, адже для вибору найкращого регулятора слід провести детальний аналіз факторів, які впливають на вибір регулятора. Також, для того, щоб

аналіз був повним нам слід побудувати декілька оптимальних регуляторів – лінійно квадратичний гауссовий регулятор, агрессивний та більш робастний (балансний).

Виходячи з сучасних реалій ринку та тенденцій в автоматизації за останні 2 роки нам слід побудувати багаторівневу систему керування безперервним хлоруванням бензолу. А для того, щоб рішення було достатньо універсальним ми маємо узгодити його з міжнародними стандартами, такими як ISA-88 та ISA-95. Оскільки розробка повністю багаторівневої системи досить часозатратна та потребує повного доступу до виробництва ми зробимо частину багаторівневої системи для якої нам не потрібно аппелювати до інших джерел окрім публічних, проте розроблене рішення має маштабуватись та мати бути модульним – що значить, що рішення можна буде легко застосувати до більшої частини процесів верхнього рівня.

Виходячи з економічної ситуації на ринку – розробити стартап проект, котрий буде відповідати до нашого виробництва, але в той же час, щоб його можна було перенести на виробництва іншого типу – для простого масштабування та пітчінгу проекту. Саме через сучасні реалії ринку та тенденції в сучасних виробничих процесах виникає необхідність розробки багаторівневої системи.

2.Архітектура багаторіневої системи керування

Для побудови багаторівневої системи ми маємо розробити її архітектуру. Для того, щоб рішення було модульним та універсальним практично для будь-якого підтриємства будемо використувовувати стандарт для багаторівневих систем ISA-88 та ISA-95. Типова архітектура багаторівневої системи:

Рис 2.1. – Типова архітектура багаторівневої системи

2.1. Стандарт ISA-88

Зрозуміло, що на підприємствах використовується найрізноманітніше обладнання, системи управління будуються на основі різних обчислювальних засобів. Для забезпечення їх спільного функціонування необхідно, щоб всі ці системи відповідали єдиним стандартам, використовували уніфікований спосіб запису рецептур, відображення поточного стану обладнання і т.д. Тенденція розвитку корпоративних систем привела до розуміння, що для ефективного управління такими складними виробництвами потрібен єдиний стандарт, і в кінці 80 х рр. XX століття було розпочато роботу зі створення стандарту ANSI / ISA 88 (часто також використовується скорочення S88). Розробник стандарту - комітет SP88 міжнародної

організації ISA, що об'єднала кінцевих користувачів і постачальників обладнання та рішень для періодичного виробництва.

У 1995 р стандарт був схвалений спільнотою ISA і отримав офіційний статус. За минулі роки ISA 88 набув широкого поширення і став визнаним стандартом реалізації систем управління періодичним виробництвом. Незважаючи на гадану очевидність затребуваності, для визнання стандарту S88 потрібні були роки. Але в результаті стандарт став широко застосовуватися на практиці і отримав подальший розвиток у вигляді стандарту ISA 95.

В даний час стандарт розвивається і вдосконалюється, великий обсяг інформаційних матеріалів можна знайти на офіційному сайті організації стандарту. В роботі були проаналізовані наявні матеріали щодо практичного використання стандарту ISA88 при вирішенні різного роду завдань як самостійно, так і в зв'язці з іншими нормативними документами. При підготовці роботи використовувалися матеріали асоціації ISA, World Batch Forum, асоціації MESA International, а також матеріали спеціалізованих Internet сайтів. Стандарт ISA-88 розроблявся на базі існуючого на той момент стандарту NAMUR N33 і був покликаний допомогти у вирішенні кількох фундаментальних проблем, таких як відсутність єдиної моделі рецептурного виробництва, складність узгодження вимог, труднощі інтеграції рішень різних постачальників, складне управління рецептурним виробництвом. Для вирішення цих проблем необхідно було визначити єдині моделі, термінологію, структуру даних і мова опису процесу. Структура стандарту відповідає поставленим завданням і включає чотири частини:

ISA88.01 1995 року, Batch Control Part 1: Models and Terminology - визначає стандартні моделі і термінологію для формалізації вимог до систем управління періодичним виробництвом, його еквівалент - IEC 61512 1;

ANSI / ISA 88.00.02 2001, Batch Control Part 2: Data Structures and Guidelines for Languages - визначає моделі даних для управління виробництвом, структури даних для обміну інформацією, а також форму записи рецептури;

ANSI / ISA 88.00.03 2003 Batch Control Part 3: General and Site Recipe Models and Representation - визначає моделі для подання узагальнених рецептур і обміну такими

рецептурами між підрозділами підприємства, а також між підприємством і його партнерами;

ANSI / ISA 88.00.04 2006, Batch Control Part 4: Batch Production Records - визначає моделі даних і орієнтовну модель системи для запису, зберігання, вилучення та аналізу даних про хід періодичного виробництва.

Крім того, готується до виходу п'ята частина стандарту Implementation Models & Terminology for Modular Equipment Control, мета якої полягає у визначенні методів формалізації інтеграції виробничого і пакувального обладнання, а також корпоративних інформаційних систем. Також на основі концепцій стандарту ISA 88 розроблені конструкції XML для обміну даними в системах управління періодичним виробництвом - BatchML.

Перш за все, стандарт ISA 88 визначає поняття періодичного процесу (batch process) - це процес, результатом якого є виробництво кінцевого кількості продукту шляхом виконання над деякою кількістю вихідних матеріалів (сировини) впорядкованої послідовності дій за обмежений період часу з використанням однієї або більше одиниць обладнання. Це визначення однозначно вказує на наявність обмеженого інтервалу часу (періоду) виготовлення кінцевого кількості продукту (тобто партії) і тим самим відділяє періодичний виробничий процес від безперервного або дискретного процесу. Термін "партія" ("batch") має значення: по-перше, "матеріал, який виробляється в результаті однієї стадії batch process", а по-друге - якась сутність, яка визначає виробництво матеріалу на будь-якій стадії процесу. Під терміном "рецептура" ("recipe") визначається мінімально необхідний набір інформації, яка унікальним чином визначає вимоги до виробництва конкретного продукту.

Для управління періодичним виробництвом необхідні три сутності (і саме їх охоплює стандарт):

- формальне визначення процесу виготовлення партії продукту - рецептури (recipe);
- інформація про обладнання, яким потрібно керувати (модель обладнання);

- формальне визначення керуючих впливів.

Трохи докладніше зупинимося на модельних уявленнях, описаних у стандарті. Стандарт ISA 88.01 визначає фізичну і процедурну моделі виробництва. Фізична модель в цілому визначає виробничу осередок обладнання, необхідного для виробництва партії продукції. Основним поняттям тут є модуль - основна одиниця устаткування, що виконує головний крок процесу. Фізична модель (модель обладнання) в загальному випадку включає сім рівнів:

- блок керування (Control Module);
- агрегат (Equipment Module);
- установка (Unit);
- осередок про процесу (Process Cell);
- виробнича дільниця (Area);
- виробництво (Site);
- підприємство (Enterprise).

Осередок процесу (Process cell) - єдиний рівень моделі. Вона включає всі установ ки, агрегати і блоки управління, необхідні для вироб ництва однієї або декількох партій. Осередок процесу може включати лінії (train), що складаються з обладнання, необхідного для виготовлення певної партії. При виготовленні партії необов'язково використовується все обладнання, що входить в лінію, в той же час одна лінія може бути задіяна одночасно у виготовленні кількох партій і / або продуктів. Осередок процесу може включати більше однієї лінії, при цьому лінія не може вклю ча обладнання, що не входить в осередок процесу. Осередки процесу є основою для планування і управління виробництвом продукції. Процедурна модель (модель ТП) в загальному випадку включає чотири рівні:

- фаза (Phase);
- виробництв жавна операція (Operation);
- процес (Unit Procedure);
- технологія (Procedure).

Технологія (Procedure) на стратегічному рівні визначає порядок заходів, які необхідно виконати для виробництва партії. Вона складається з необхідного числа процесів (Unit Procedure). А про процес, в свою чергу, складається з зв'язковою послідовно ності виробничих операцій, що виконуються на одній установці, і моделює один з ос новних кроків (стадій) створення партії. , передбачається, що в кожен момент часу на установці виконується тільки одна операція, при цьому кожна операція повністю завершується в межах установки. Проте безліч процесів однієї технології може виконуватися на кількох установках одночасно, а кожен процес на своїй установці.

Застосування моделей стандарту ISA-88 як до великих виробництв зі складними ТП і структурою обладнання, так і до простих за структурою підприємствам неодноразово перевірена на практиці. Обумовлені стандартом моделі дозволяють вбудовувати прості процеси в процеси вищого рівня, наявність яких не передбачалося спочатку. Даний факт дає можливість невеликим підприємствам завдяки застосуванню стандарту ISA-88 отримати більше можливостей у частині кооперації з великими підприємствами - замовниками їхньої продукції. Підходи, аналогічні моделі обладнання стандарту ISA- 88, і її розвиток в стандарті ISA-95 можуть використовуватися і при описі не тільки виробничих технологічних об'єктів і структур.

При практичній реалізації ієрархія моделей стандарту використовується як етапність для впровадження системи управління. Впровадження відбувається знизу вгору: спочатку блоки управління, потім моделі обладнання, потім фази обладнання і далі до моделей процедур. Цей процес впровадження може бути припинений на будь-якій стадії, як тільки будуть виконані вимоги до системи управління. При цьому на кожній стадії впровадження (на кожному з рівнів моделі) досягаються цілком певні результати, що дає можливість оцінити економічну ефективність кожного з етапів впровадження в окремо.

На сучасному виробничому підприємстві АСУ ТП повинні взаємодіяти не тільки з аналогічними системами, але і з системами управління верхнього рівня, в тому числі MES і ERP. Якщо обмін даними з MES ще може бути описаний в рамках

ISA 88, то передача даних від MES до ERP вимагає вже інших стандартів. Один з найбільш відомих стандартів обміну інформацією між MES і корпоративними інформаційними системами - стандарт ISA-95. На перший погляд, ISA-95 і ISA-88 чудово доповнюють один одного, так що складається враження, що разом вони охоплюють всі виробничі системи - від MES до локальних систем управління. Але в той же час між двома стандартами існують серйозні відмінності, що стосуються моделей обладнання, функціональних та інформаційних моделей. Без урахування таких відмінностей важко розраховувати на позитивний результат при спільному використанні стандартів.

Візуально стандарт ISA-88 можна зобразити наступним чином:

Рис 2.2. – Стандарт ISA-88

2.2. Стандарт ISA-95

ISA-95, або ANSI / ISA-95, - міжнародний стандарт для розробки інтерфейсу між підприємствами і керуючими системами. Цей стандарт був розроблений для застосування в усіх видах виробництва, для всіх видів процесів - наприклад - безперервних або повторюваних. ISA-95 - стандарт для північноамериканського

ринку, йому відповідає європейський стандарт IEC 62264. На сьогодні відомі 6 частин стандарту ISA-95.

ANSI / ISA-95.00.01-2000. Частина 1: Моделі і термінологія складається зі стандартної термінології та об'єктних моделей, які можуть бути використані, щоб вирішити, яка інформація підлягає обміну.

Моделі допомагають визначити інтерфейси між системами підприємства і системами управління. Вони допомагають вирішити, які завдання можуть бути виконані якими функціями і якою інформацією повинні обмінюватися між собою додатки.

ANSI / ISA-95.00.02-2001. Частина 2: Атрибути об'єктної моделі містить атрибути для кожного об'єкта, визначеного в частині 1. Об'єкти і атрибути частини 2 можуть бути використані для обміну інформацією між різними системами, але ці об'єкти і атрибути також можуть бути використані в якості основи для проектування реляційних баз даних.

ANSI / ISA-95.00.03-2005. Частина 3: Операційні моделі управління виробництвом спеціалізується на функціях і діяльності на рівні 3 (Виробництво / MES шар). Це забезпечує керівні принципи для опису і порівняння рівнів виробництва різних сайтів в стандартизованої формі.

ISA-95.04. Частина 4: Об'єктні моделі і атрибути для операцій управління виробництвом. (Ще не опублікована). Ця технічна специфікація визначає обмін інформацією між функціями MES, які визначені в частині 3 ISA-95. Моделі і атрибути з частини 4 є основою для розробки і впровадження стандартів інтерфейсу і забезпечення гнучкого взаємодії та обміну інформацією між різними MES-функціями.

ISA-95,05 Частина 5: Трансакції між бізнесом і виробництвом (B2M трансакції). Визначає взаємодію між системами, що автоматизують роботу офісу і виробництва.

ISA-95,05 Частина 6: Трансакції між виробничими операціями (Ще не опублікована).

Стандарт передбачає еталонні моделі, які описують всі види діяльності - виробництва, контролю якості, технічного обслуговування і інвентаризації.

Стандарт ISA-95 можна зобразити наступним чином:

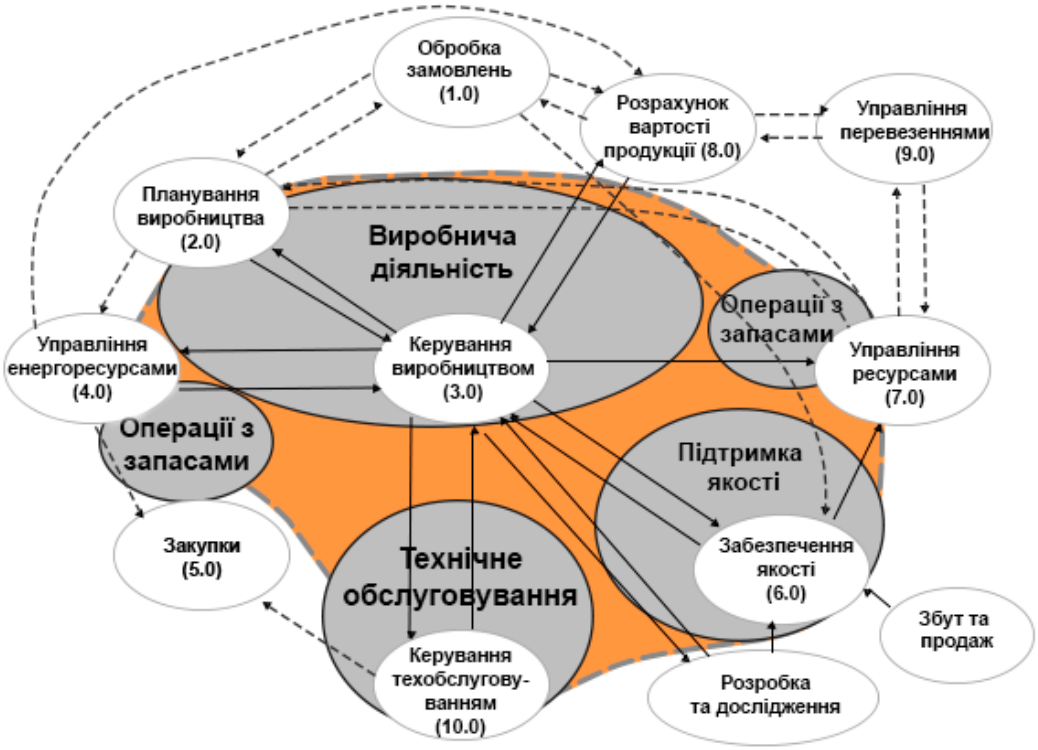


Рис. 2.3 – Стандарт ISA-95

2.3. Застосування стандартів ISA-88 та ISA-95 для виробництва безперервного хлорування бензолу

Згідно стандарту ISA-88 було розроблено найнижчий рівень, а саме моделювання та інтеграцію рішення на рівень ISA-95.

Рис 2.4 – Схема багаторівнеової системи згідно стандартів ISA-88, ISA-95

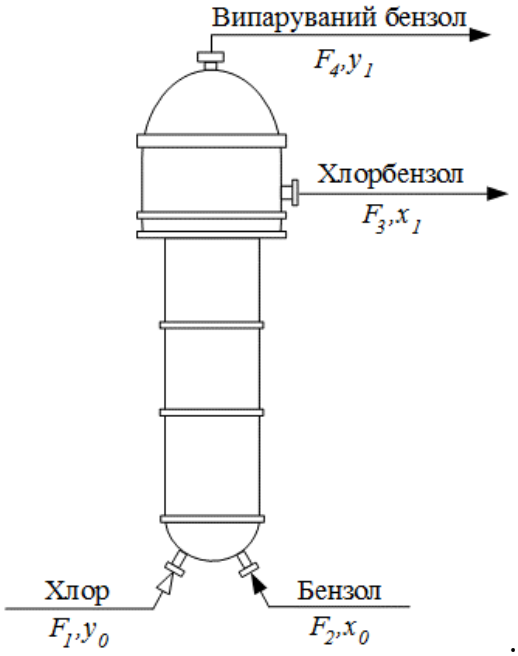
Для розробки системи згідно стандарту ISA-95 виникає необхідність розробки MES системи. Для стандарту ISA-88 необхідно розробити математичну модель та налаштування регуляторів в одному з програмних пакетів, котрі використовують для автоматизації. Для цього було обрано програмний пакет MATLAB. Для стандарту ISA-95 фронт можливих робіт дуже обширний, тому ми будемо розглядати лише малу частину стандарту ISA-95 та в свою чергу MES систем. А саме: скрізна аналітика, отримання даних, інтеграція даних в одну базу даних, обробка та збереження даних, та, звичайно візуальну інтерпретацію та трансляцію даних через веб сервіс. Система в фінальному матиме вигляд зображений на рисунку 2.4

3. Математичне моделювання хлоратора

3.1. Моделювання статичного режиму

Оскільки процес хлорування (абсорбції) бензолу є основною технологічною стадією одержання хлорбензолу, постає задача створення системи керування хлоратором(абсорбером) для забезпечення ефективності виробництва. Основним апаратом є хлоратор. Керування об’єктом полягає в підтриманні заданої концентрації хлорбензолу у вихідному розчині. Це досягається за рахунок зміни витрати бензолу.

Розрахункову схема хлоратора наведено на рисунку 3.1.



:

F_1 – витрата хлору; F_2 – витрата бензолу; F_3 – витрата хлорбензолу; F_4 – витрата випарюваного бензолу з хлорбензолом; x_0 – концентрація бензолу; x_1 – концентрація хлорбензолу; y_0 – концентрація хлору; y_1 – концентрація випарюваного бензолу з хлорбензолом

Рисунок 3.1. – Розрахункова схема абсорбера

Для створення системи керування необхідно отримати математичну модель каналу керування. Спочатку розглянемо статичний режим.

Статичний режим – це режим роботи системи автоматичного керування, в якому керована величина та всі проміжні величини не змінюються у часі. Графічно цей режим зображується за допомогою статичної характеристики, яка являє собою

залежність керованого параметра (вихідної величини) від керуючого впливу (вхідної величини).

Каналом керування називається внутрішній зв'язок між входом та виходом об’єкта. Окрім основної керуючої величини на об’єкт можуть діяти різного виду збурення, що являють собою будь-яку дію, яка намагається порушити необхідний функціональний зв'язок у системах автоматичного керування. Зв'язок вихідної величини зі збурюючим впливом називається каналом збурення.

В нашому випадку маємо такі канали об’єкта:

- Канал керування $F_2 \rightarrow x_1$
- Канал збурення $x_0 \rightarrow x_1$.

Рівняння матеріального балансу мають вигляд:

- для газу:

$$-F_4 y_1 - SK_{\text{г}} \frac{(y_0 - y_r(x_1)) + (y_1 - y_r(x_0))}{2} = 0, \quad (3.1)$$

- для рідини:

$$F_2 x_0 - F_3 x_1 + SK_{\text{р}} \frac{(x_0 - x_r(y_1)) + (x_r(y_0) - x_1)}{2} = 0, \quad (3.2)$$

де $y_r(x)$ – рівноважна концентрація C_6H_6 у випарованому бензолі з хлорбензолом; $K_{\text{г}}$ – коефіцієнт масообміну, поданий через мольні частки компонента в газовій фазі; S– поверхня масообміну у хлораторі; $V_{\text{г}}$ – об’єм хлоратора, зайнятий газом; $\rho_{\text{г}}$ – густина суміші. де $x_r(y)$ – рівноважна концентрація C_6H_6 у хлорбензолі; $K_{\text{р}}$ – коефіцієнт масообміну, поданий через мольні частки компонента у рідкій фазі; S– поверхня масообміну у хлораторі; $V_{\text{р}}$ – об’єм хлоратора, зайнятий рідиною; $\rho_{\text{р}}$ – густина розчину.

Так як ми використовуємо канал керування $F_2 – X_1$, у подальших розрахунках будемо використовувати рівняння матеріального балансу для рідини. Тому виразимо з формули (3.2) вихідну величину:

$$x_1 = \frac{2F_2 x_0 + SK_p(x_0 - x_r(\frac{SK_{\text{г}}(y_r(x_0) + y_r(x_1) - y_0)}{2F_4 + SK_{\text{г}}} + x_r(y_0))}{2F_3 + SK_p} \quad (3.3)$$

Таблиця 3.1 – Параметри статичного режиму

№ п/п	Назва параметру	познач.	одиниці вимір.	числове значення
1	Витрата хлору	$F_г$	кг /с	0.38
2	Витрата бензолу	$F_р$	кг /с	0.76
3	Початкова концентрація бензолу в хлорі	y_0	кг бензолу/кг суміші	0
4	Кінцева концентрація бензолу що випаровуваному бензолі	y_1	кг / кг суміші	0,71
5	Початкова концентрація бензолу	x_0	кг / кг розчину	1
6	Кінцева концентрація бензолу в хлорбензолі	x_1	кг / кг розчину	0,55
7	Об’єм, який займають гази у хлораторі	$V_г$	$м^3$	38
8	Об’єм, який займає рідина у хлораторі	$V_р$	$м^3$	57
9	Густина газової фази	$\rho_г$	кг/ $м^3$	139
10	Густина рідкої фази	$\rho_р$	кг/ $м^3$	1066
11	Кількість бензолу, яка є випаровуваному бензолі на одиницю рушійної сили в газовій фазі	$SK_г$	кг суміші/год	356
12	Кількість бензолу, яка прийшла на одиницю рушійної сили в рідині	$SK_р$	кг розчину/год	1718

13	Постійна Генрі	M	кг розчину/кг суміші	8,73
14	Рівноважна концентрація бензолу у газі на вході в хлоратор	$x_r(y_0)$	кг / кг розчину	0
15	Рівноважна концентрація бензолу у рідині на вході в хлоратор	$y_r(x_0)$	кг / кг суміші	8,73
16	Рівноважна концентрація бензолу в рідині на виході із хлоратора	$x_r(y_1)$	кг / кг розчину	0,081
17	Рівноважна концентрація бензолу у газі на виході із хлоратора	$y_r(x_1)$	кг / кг суміші	4,802

Дослідимо залежність концентрації хлорбензолу X_1 на виході з хлоратора (абсорбера) від витрати бензолу:

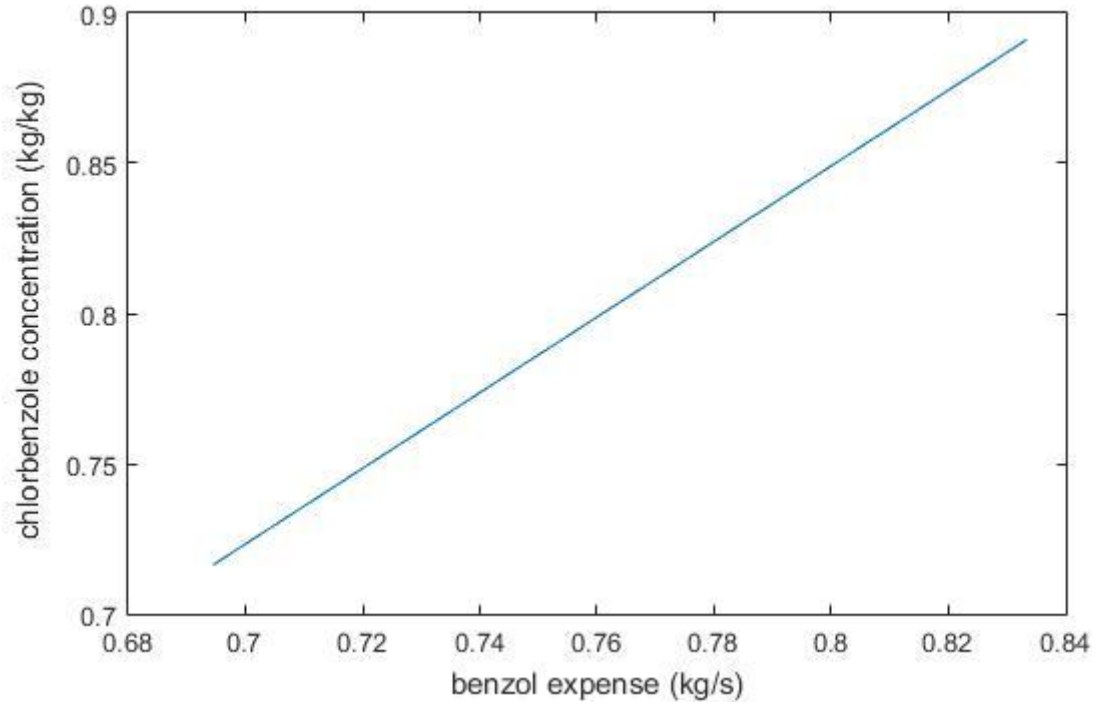


Рисунок 3.2. – Статична характеристика каналу керування

Із отриманого графіка видно, що збільшення витрати бензолу призводить до збільшення концентрації хлорбензолу.

3.2. Моделювання динамічного режиму

Динамічний режим - режим роботи об’єкта (пристрою, механізму, машини тощо), за якого хоч би один з параметрів режиму змінюється у часі.

Характерною особливістю об’єктів регулювання є розподіл їх параметрів у просторі. Інколи він незначний, тоді можна припустити, що параметри об’єкта зосереджені в одній точці – це об’єкти із зосередженими параметрами. У протилежному випадку виділяють об’єкти із розподіленими параметрами. Для зосереджених об’єктів змінні, що описують поведінку об’єкта, змінюються тільки в часі. Для об’єкта з розподіленими параметрами – і у часі, і у просторі. Ми розглядаємо абсорбер як об’єкт із зосередженими параметрами, тобто припускаємо, що параметри по довжині об’єкта не змінюються.

Рівняння динаміки мають вигляд:

Розроблено матеріальні баланси для C_6H_6 у рідкому та газовому станах. У рідкій фазі матеріальний баланс виглядатиме:

$$F_2x_0 - F_3x_1 + SK_{\text{р}}\frac{(x_0 - x_r(y_1)) + (x_r(y_0) - x_1)}{2} = V_{\text{р}}\rho_{\text{р}}\frac{d}{dt}\left(\frac{x_0 + x_1}{2}\right), \quad (3.4)$$

де $x_r(y)$ – рівноважна концентрація C_6H_6 у хлорбензолі; $K_{\text{р}}$ – коефіцієнт масообміну, поданий через мольні частки компонента у рідкій фазі; S– поверхня масообміну у хлораторі; $V_{\text{р}}$ – об’єм хлоратора, зайнятий рідиною; $\rho_{\text{р}}$ – густина розчину.

У газовій фазі матеріальний баланс виглядає:

$$-F_4y_1 - SK_{\text{г}}\frac{(y_0 - y_r(x_1)) + (y_1 - y_r(x_0))}{2} = V_{\text{г}}\rho_{\text{г}}\frac{dy_1}{dt}, \quad (3.5)$$

де $y_r(x)$ – рівноважна концентрація C_6H_6 у випарованому бензолі з хлорбензолом; $K_{\text{г}}$ – коефіцієнт масообміну, поданий через мольні частки компонента в газовій фазі; S–

поверхня масообміну у хлораторі; $V_г$ – об'єм хлоратора, зайнятий газом; $\rho_г$ – густина суміші.

Для рівноважних концентрацій прийнято наступне: $y_r(x) = mx$, $x_r(y) = y/m$.

Лінеаризуємо рівняння

Змінними параметрами є:

- Регульований параметр: $x_{1;}$
- Керуюча дія : $F_{р.}$
- Збурення: $x_{0.}$
- Проміжна змінна: y_1.

Для отримання рівнянь у відхиленнях (лінеаризованих) приймаємо:

$$x_1(t) = x_{1_0} + \Delta x_1(t);$$

$$F_р(t) = F_{р0} + \Delta F_р(t);$$

$$x_0(t) = x_{0_0} + \Delta x_0(t);$$

$$y_1(t) = y_{1_0} + \Delta y_1(t).$$

Лінеаризуємо рівняння в точці основного статичного режиму. Візьмемо частинні похідні від першого та другого рівнянь по змінним параметрам та отримаємо систему рівнянь у відхиленнях:

$$\begin{cases} -\frac{V_г\rho_г}{2}\frac{d\Delta y_1}{dt} - \left(F_г + \frac{SK_г}{2}\right)\Delta y_1 = -(x_0 - x_1)\Delta F_p - \frac{SK_г m}{2}\Delta x_1 - \frac{SK_p m}{2}\Delta x_0, \\ -\frac{V_p\rho_p}{2}\frac{d\Delta x_1}{dt} + \left(-F_р - \frac{SK_р}{2}\right)\Delta x_1 = \frac{V_p\rho_p}{2}\frac{d\Delta x_0}{dt} + \left(-F_р - \frac{SK_р}{2}\right)\Delta x_0 + \left(\frac{SK_р}{2m}\right)\Delta y_1 \end{cases} \quad (3.6)$$

Рівняння в безрозмірному вигляді змінних

Приведемо систему рівнянь (3.6) до безрозмірного виду визначальних величин і поділимо перше рівняння на y_{10}, а друге - x_{10}:

$$\Delta x_1 = \Delta x_1^{\,б} \times x_{1_0}, \Delta x_0 = \Delta x_0^{\,б} \times x_{0_0}, \Delta F_p = \Delta F_p^{\,б} \times F_{p_0}, \Delta y_1 = \Delta y_1^{\,б} \times y_{1_0}.$$

$$\begin{cases} T_{y_1} \frac{d\Delta y_1^{\,б}}{dt} + \Delta y_1^{\,б} = \frac{K_{F_p y_1}}{y_{1_0}} \Delta F_p^{\,б} + \frac{K_{x_1 y_1}}{y_{1_0}} \Delta x_1^{\,б} + \frac{K_{x_0 y_1}}{y_{1_0}} \Delta x_0^{\,б}, \\ T_{x_1} \frac{d\Delta x_1^{\,б}}{dt} + \Delta x_1^{\,б} = T_{x_0} \frac{d\Delta x_0^{\,б}}{dt} \Delta x_0^{\,б} + \frac{K_{x_0 x_1}}{x_{1_0}} \Delta x_0^{\,б} - \frac{K_{y_1 x_1 y_{1_0}}}{x_{1_0}} \Delta y_1^{\,б} \end{cases}.$$

де позначення T_{x1}, T_{y0} і всіх коефіцієнтів наведено далі.

Рівняння в канонічній формі і в формі Коші

Запишемо систему рівнянь (3.6) у канонічній формі. Для цього потрібно перше та друге рівняння для газової та рідкої фаз поділити на коефіцієнти перед Δy_1, та на коефіцієнт перед Δx_1. Отримаємо таку систему рівнянь:

$$\begin{cases} T_{y_1} \frac{d\Delta y_1}{dt} + \Delta y_1 = K_{F_p y_1} \Delta F_p + K_{x_1 y_1} \Delta x_1 + K_{x_0 y_1} \Delta x_0, \\ T_{x_1} \frac{d\Delta x_1}{dt} + \Delta x_1 = T_{x_0} \frac{d\Delta x_0}{dt} \Delta x_0 + K_{x_0 x_1} \Delta x_0 + K_{y_1 x_1} \Delta y_1, \end{cases} \quad (3.7)$$

$$\text{де} \quad T_{y1} = \frac{V_{\text{г}} \rho_{\text{г}}}{2F_{\text{г}} + SK_{\text{г}}}; \; K_{x_1 y_1} = \frac{m \cdot SK_{\text{г}}}{2F_{\text{г}} + SK_{\text{г}}}; \; K_{x_0 y_1} = \frac{mSK_{\text{р}}}{2F_{\text{г}} + SK_{\text{г}}};$$

$$T_{x0} = \frac{V_{\text{р}} \rho_{\text{р}}}{-2F_p - SK_{\text{р}}}; \; K_{F_p y_1} = \frac{2(x_0 - x_1)}{2F_{\text{г}} + SK_{\text{г}}};$$

$$T_{x1} = \frac{V_{\text{р}} \rho_{\text{р}}}{2F_p + SK_{\text{р}}}; \; K_{\text{y}_1 x_1} = \frac{SK_{\text{р}}}{m(2F_p + SK_{\text{р}})}; \; K_{\text{x}_0 \text{x}_1} = 1.$$

Запишемо систему рівнянь (3.7) у формі Коші. Для цього залишимо похідні від зліва, а все інше перенесемо вправо і поділимо на відповідні коефіцієнти при похідних, тобто на $V_{\text{г}} \rho_{\text{г}}$ та $V_{\text{р}} \rho_{\text{р}}$:

$$\begin{cases} \frac{d\Delta y_1}{dt} = -\frac{2F_{\text{г}} + SK_{\text{г}}}{(V_{\text{г}} \rho_{\text{г}})} \Delta \text{y}_1 + \frac{2(x_0 - x_1}{(V_{\text{г}} \rho_{\text{г}})} \Delta F_p + \frac{m \cdot SK_{\text{г}}}{(V_{\text{г}} \rho_{\text{г}})} \Delta \text{x}_1 + \frac{m \cdot SK_{\text{р}}}{(V_{\text{г}} \rho_{\text{г}})} \Delta \text{x}_0, \\ \frac{d\Delta x_1}{dt} = -\frac{d\Delta \text{x}_0}{dt} + \frac{(2F_p + SK_{\text{р}})}{V_{\text{р}} \rho_{\text{р}}} \Delta x_0 - \frac{(2F_p + SK_{\text{р}})}{V_{\text{р}} \rho_{\text{р}}} \Delta x_1 - \frac{SK_{\text{р}}}{m(V_{\text{р}} \rho_{\text{р}})} \Delta y_1 \end{cases}.$$

Перетворення за Лапласом змінної часу

Виконаємо перетворення за Лапласом змінної $t \to p$ системи (3) при нульових початкових умовах, де:

$$\Delta x_1 \ \to \ x_1(p);$$

$$\Delta x_0 \ \to \ x_0(p);$$

$$\Delta F_p \ \to \ F_p(p);$$

$$\Delta y_1 \ \to \ y_1(p).$$

Отримаємо наступну систему лінійних рівнянь відносно регульованої величини $x_1(p)$ та другої вихідної змінної $y_1(p)$:

$$\begin{cases} (T_{y_1}p + 1)\, y_1(p) = \ K_{F_p y_1} F_p(p) + K_{\mathrm{x_1 y_1}} x_1(p) + K_{\mathrm{x_0 y_1}} x_0(p), \\ (T_{x_1}p + 1)\, x_1(p) = (T_{\mathrm{x_0}}p + K_{\mathrm{x_0 x_1}}) x_0(p) - K_{y_1 x_1} y_1(p), \end{cases}$$

Визначення коефіцієнтів рівняння

За формулами знайдемо відповідні значення коефіцієнтів:

$T_{y_1} = 1{,}17e + 04;\ T_{x_1} = 3e + 04;\ T_{x_0} = -3{,}0525e + 04;\ K_{F_p y_1} = -33.97;$

$K_{\mathrm{x_1 y_1}} = 1{,}92;\ K_{x_0 \mathrm{y_1}} = 9{,}26;\ K_{\mathrm{y_1 x_1}} = 0{,}025;\ K_{\mathrm{x_0} x_1} = 1.$

Передатні функції за каналами керування і збурення

Передатні функції за каналами керування $F_p \to x_1$ та збурення $x_0 \to x_1$:

$$\begin{cases} W_{\mathrm{x_0}}(p) = \dfrac{\left(T_{y_1}p + 1\right)\left(T_{x_0}p + 1\right) - K_{\mathrm{y_1 x_1}} K_{\mathrm{x_0 y_1}}}{\left(T_{y_1}p + 1\right)\left(T_{x_1}p + 1\right) - K_{x_1 y_1} K_{y_1 x_1}}, \\ W_{F_p}(p) = \dfrac{-K_{\mathrm{y_1 x_1}}\, K_{F_p y_1}}{\left(T_{y_1}p + 1\right)\left(T_{x_1}p + 1\right) - K_{x_1 y_1} K_{y_1 x_1}}. \end{cases}$$

Вигляд передавальних функцій $W_{Fp}(p)$ та $W_{x0}(p)$ у чисельній формі зображено системою рівнянь:

```
W_control =

                0.9329
  ---------------------------------
  3.587e08 s^2 + 4.228e04 s + 1.053

W_zb =

  -3.587e08 s^2 - 1.877e04 s + 0.7454
  ---------------------------------
   3.587e08 s^2 + 4.228e04 s + 1.053
```

Тепер побуємо перехідні характеристики по цим каналам:

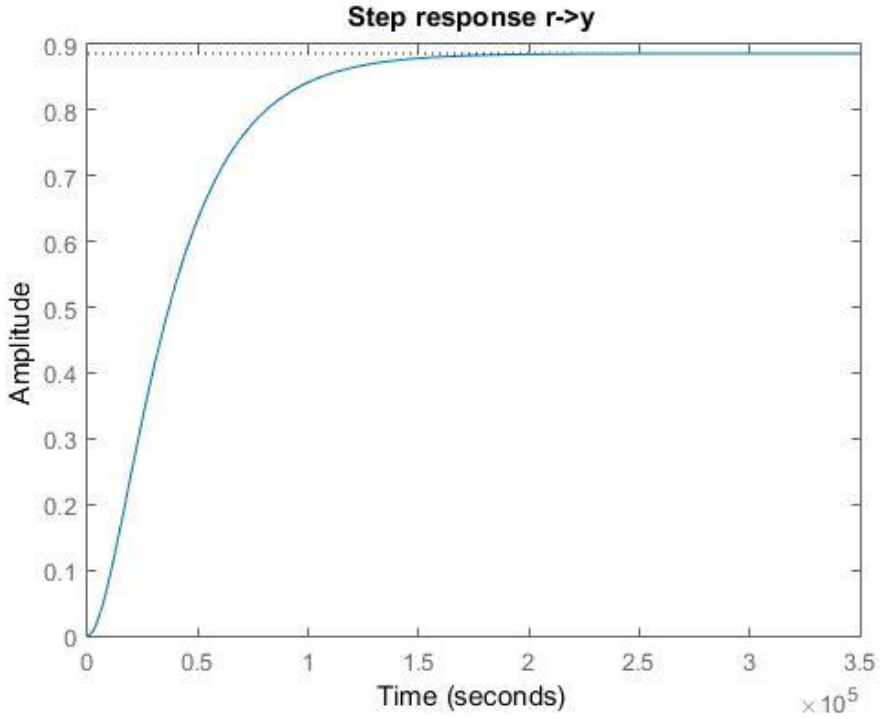


Рисунок 3.4. – Перехідна характеристика каналу керування

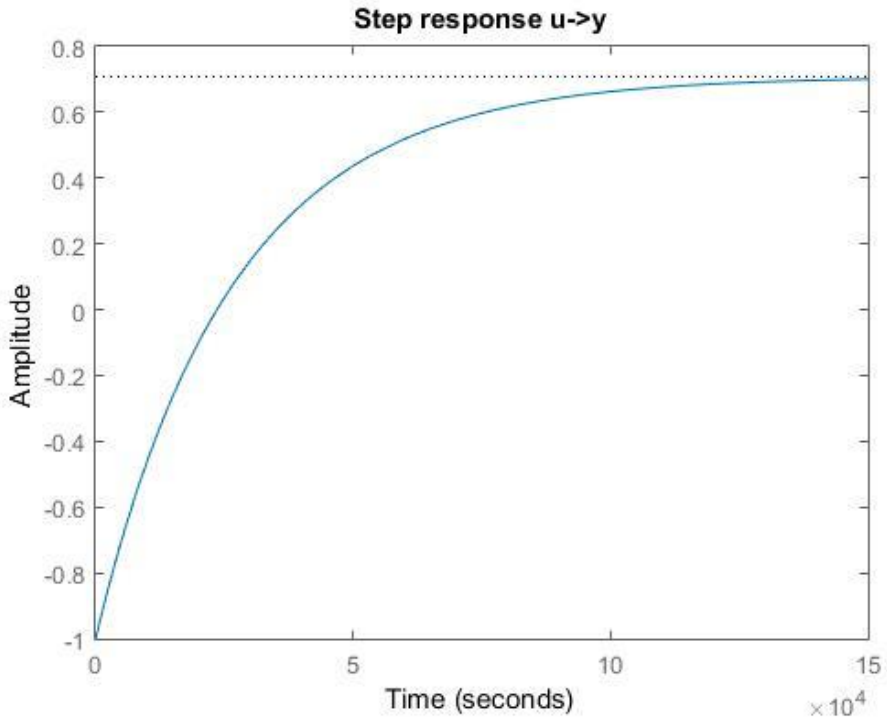


Рисунок 3.5. – Перехідна характеристика каналу збурення

Лістинг програми побудови динаміки та перехідних характеристик в Matlab:

```
clear all;
F_g = 631;
V_g = 38;
m = 8.73;
F_r = 2724;
V_r = 57;
y_0 = 0;
p_g = 139;
y_1 = 0.71;
p_r = 1066;
x_0 = 1;
S_Kg = 356;
x_1 = 0.55;
S_Kr = 1718;
xr_y0 = y_0/m;
xr_y1 = y_1/m;
yr_x0 = m*x_0;
yr_x1 = m*x_1;
T_y1 = V_g*p_g/(2*F_g+S_Kg);
K_Fr_y1 = 2*(x_0-x_1)/(2*F_g+S_Kg);
```

```
K_x1_y1 = m*S_Kg/(2*F_g+S_Kg);
K_x0_y1 = m*S_Kr/(2*F_g+S_Kg);
T_x0 = V_r*p_r/(-2*F_r-S_Kr);
T_x1 = V_r*p_r/(2*F_r+S_Kr);
K_y1_x1 = S_Kr/(m*(2*F_r+S_Kr));
K_x0_x1 = 1;
W_control = tf(K_y1_x1*K_Fr_y1, [T_y1*T_x1 T_y1+T_x1 K_y1_x1*K_x1_y1+1]);
W_zb = tf([T_y1*T_x0 T_y1+T_x0 1-K_y1_x1*K_x0_y1],[T_y1*T_x1 T_y1+T_x1
1+K_x1_y1*K_y1_x1]);
figure;
step(W_control);
figure;
step(W_zb);
figure;
impulse(W_control);
figure;
impulse(W_zb);
```

Ми отримали передатні функції каналів керування та збурення. Тепер можна перейти до підбору і розрахунку налаштувань регулятора.

4. Синтез та дослідження системи керування абсорбером

Маючи математичну модель об’єкту у вигляді передатної функції каналу керування

$$W(s) = \frac{0.9329}{3.587e08s^2+4.2e04s+1}, \quad (4.1)$$

можна на її основі підібрати параметри регулятора для забезпечення заданих показників якості системи керування. ЇЇ можна налаштувати за різними показниками. Розглянемо ці показники. Проаналізувавши передавальну функцію ми можемо зробити висновок, що для усталенню системі треба досить багато часу(оскільки коефіцієнти при поліномі знаменника досить великі), звідси можемо зробити висновок, що синтез регулятора для системи – необхідний.

4.1. Аналіз показників якості системи керування

До систем автоматичного керування (САК) пред'являються вимоги не тільки стійкості процесів регулювання. Для працездатності системи важливо, щоб процес автоматичного регулювання здійснювався при забезпеченні певних показників якості процесу управління.

Якщо досліджувана САК є стійкою, виникає питання про те, наскільки якісно відбувається регулювання в цій системі і чи задовольняє воно технологічним вимогам об'єкта управління.

Класифікація показників якості складається з декількох груп:

- прямі - визначаються безпосередньо по перехідній характеристиці процесу;
- кореневі - визначаються по кореню характеристичного полінома;
- частотні - по частотних характеристиках;
- інтегральні – одержуються шляхом інтегрування функцій.

Прямими показниками якості процесу управління, які визначаються безпосередньо по перехідній характеристиці є:

1) Усталене значення вихідної величини Yуст;

2) Ступінь затухання Ψ;

3) Час досягнення першого максимуму tmax;

4) Час регулювання tp;

5) Помилка регулювання Ест (статистична або середньоквадратична складова);

6) Перерегулювання у;

7) Динамічний коефіцієнт регулювання Rd;

8) Показник коливності М.

Наприклад, перехідна характеристика, знята на об'єкті керування при відпрацюванні ступінчатого впливу, має коливний вигляд і представлена на рис. 5.1.

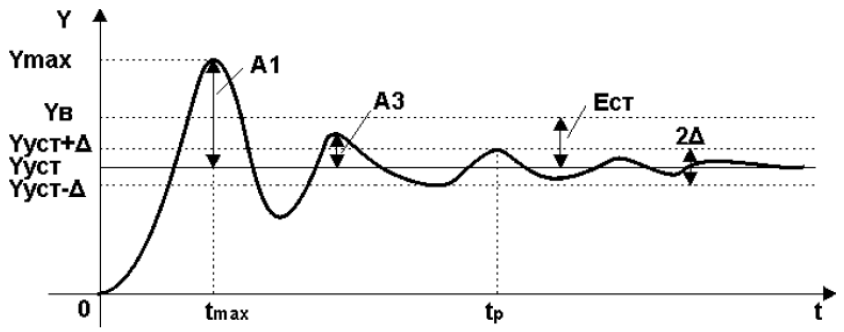


Рисунок 4.1. – Визначення показників якості по перехідній характеристиці

1) Усталене значення вихідної величини Yуст;

Усталене значення вихідної величини Yуст визначається по перехідній характеристиці, представленої на рис. 4.1.

2) Ступінь затухання Ψ;

Ступінь затухання Ψ визначається за формулою:

$$\Psi = \frac{A1 - A3}{A3}$$

де A1 та A3 - відповідно 1-а і 3-а амплітуди перехідної характеристики рис. 4.1.

3) Час досягнення першого максимуму tmax;

Час досягнення першого максимуму tmax визначається по перехідній характеристиці, представленої на рис. 4.1.

4) Час регулювання tp;

Час регулювання tp визначається згідно рис. 4.1 таким чином: Знаходиться допустиме відхилення Д, наприклад, задане Д = 5%Yуст і будується «зона» завтовшки 2Д (див. рис. 4.1). Час tp відповідає останній точці перетину Y(t) з даною кордоном. Тобто час, коли коливання регульованої величини перестають перевищувати 5% від сталого значення.

Налаштування регулятора необхідно вибирати так, щоб забезпечити мінімально можливе значення загального часу регулювання, або мінімальне значення першої напівхвилі перехідного процесу. У безперервних системах з типовими регуляторами цей час буває мінімальним при так званих оптимальних аперіодичних перехідних процесах. Подальшого зменшення часу регулювання до абсолютного мінімуму можна досягти при використанні спеціальних оптимальних по швидкодії систем регулювання.

5) Помилка регулювання Ест (статистична або середньоквадратична складова);

Статична помилка регулювання Ест=Ув-Ууст, де Ув - вхідна величина (див. рис. 4.1). У деяких САК спостерігається помилка, яка не зникає навіть після закінчення тривалого інтервалу часу – це статична помилка регулювання Ест. Дана помилка не повинна перевищувати деякої наперед заданої величини.

У регуляторів з інтегральної складової помилки в установленому стані теоретично дорівнюють нулю, але практично незначні помилки можуть існувати із-за наявності зон нечутливості в елементах системи.

6) Перерегулювання у;

Величина перерегулювання «у» залежить від виду відпрацьовуваного сигналу. При відпрацюванні ступеневого впливу (по сигналу завдання) - див рис. 4.1 величина перерегулювання «у» визначається за формулою:

$$y = \frac{Y\max - Y\ уст}{Y\ уст} * 100\%$$

де значення величин Ymax і Yуст визначаються згідно рис. 4.1.

При відпрацюванні збурюючого впливу, величина перерегулювання «у» визначається співвідношенням:

$$y = \frac{X1}{Xm} *100\%$$

де значення величин Xm і X1 визначаються згідно рис. 4.2.

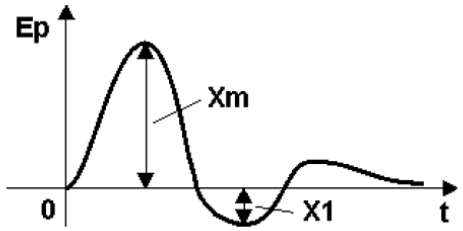


Рис. 4.2. – Графік перехідного процесу при при відпрацьовуванні збурення

7) Динамічний коефіцієнт регулювання Rd;

Динамічний коефіцієнт регулювання Rd визначається із формули:

$$Rd = \frac{Y1}{Y0} *100\%$$

де значення величин Y1 і Y0 визначаються згідно рис. 4.3.

Рис. 4.3. – Зображення впливу динамічного коефіцієнта Rd на процес

Величина динамічного коефіцієнта Rd характеризує ступінь впливу регулятора на процес, тобто ступінь зниження динамічного відхилення в системі з регулятором і без нього.

8) Показник коливності М.

Показник коливності М характеризує величину максимуму модуля частотної передавальної функції замкнутої системи (на частоті резонансу) і, тим самим, характеризує коливальні властивості системи. Показник коливності наочно ілюструється на рис. 5.4.

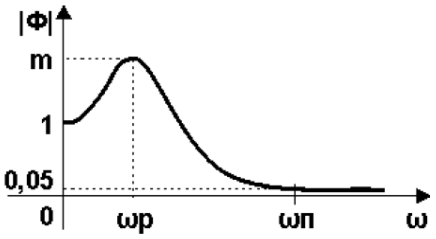


Рис. 4.4. – Графік модуля частотної передавальної функції замкнутої системи

Умовно вважається, що значення M = 1,2-1,4 є оптимальним для промислових САК, тому що в цьому випадку у забезпечується в районі від 20% до 40%. При збільшенні значення M коливність в системі зростає. У деяких випадках нормується смуга пропускання системи щп, яка відповідає рівню посилення в замкнутій системі 0,05. Чим більше смуга пропускання, тим більше швидкодія замкнутої системи. Однак при цьому підвищується чутливість системи до шумів в каналі вимірювання і зростає дисперсія помилки регулювання.

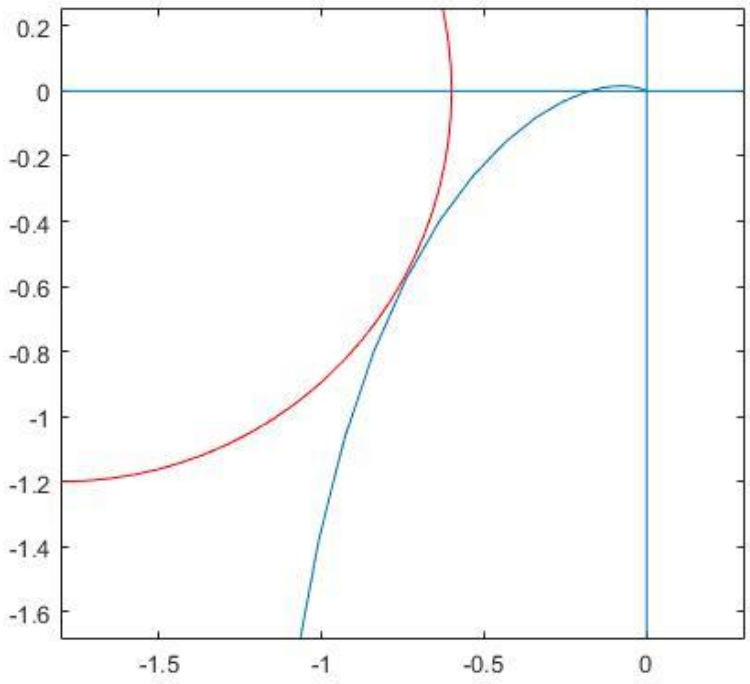


Рис. 4.5. – Дотик АФХ розімкненої системи до М-кола

Параметри регулятора, що забезпечують заданий показник коливності:

$$K = 5, T = 7415, M = 1.4$$

Побудуємо перехідні характеристики замкнутої системи

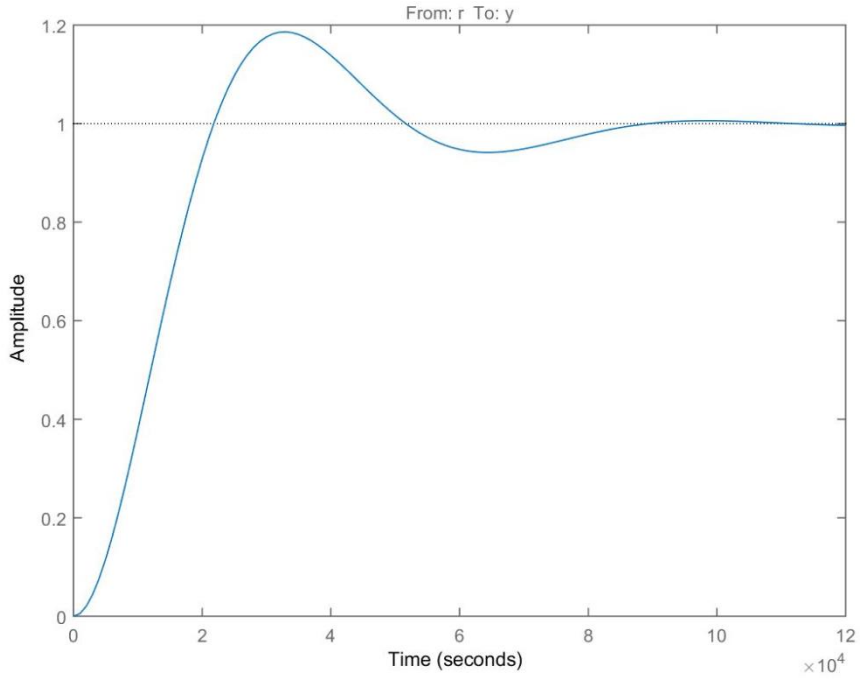


Рис. 4.6. – Перехідна характеристика замкнутої системи з ПІ-регулятором

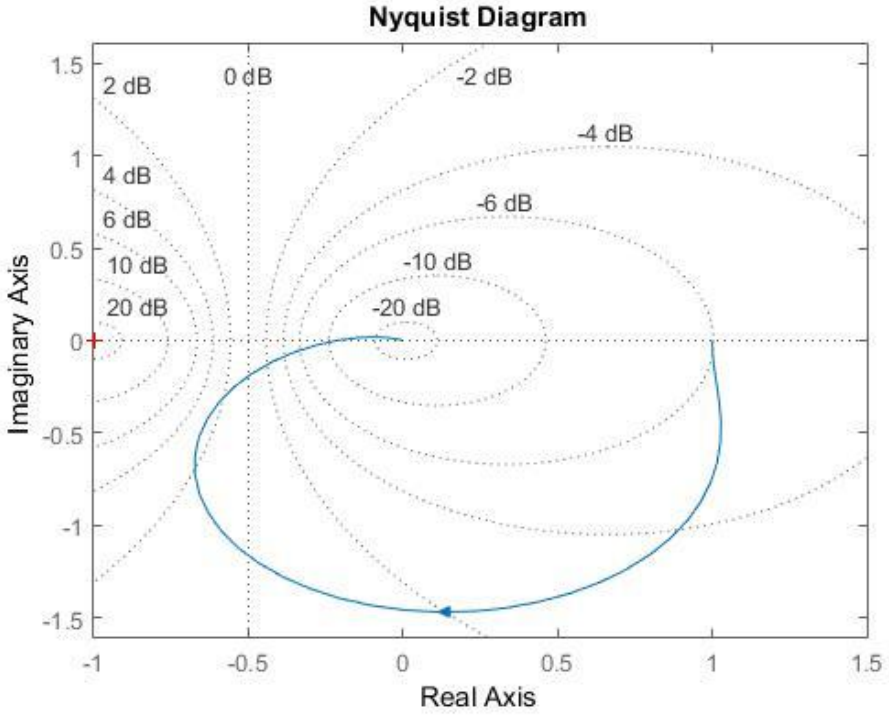


Рис. 4.7. – Годограф Найквіста системи з ПІ регулятором

В результаті використаного методу були знайдені налаштування ПІ-регулятора, які забезпечують заданий показник коливності системи. За рахунок даного регулятора

можна реалізувати роботу хлоратора в процесі безперервного хлорування бензолу. Також з отриманих розрахунків можна зробити висновок, що наша система є стійкою.

4.2Розрахунок параметрів регулятора методом Чина-Ресквіка

У практичній діяльності для налагодження систем регулювання, як правило, використовують наближені методи розрахунку параметрів регулятора. Такий підхід дозволяє досить швидко, без проведення складних попередніх досліджень розв’язати задачу вибору властивостей регулятора згідно до прийнятого критерію керування та в залежності від властивостей ОК. Для наближеної оцінки динамічних властивостей використовують спрощені математичні моделі ОК у вигляді передаточних функції. Отримано та використовується багато еврістичних правил щодо настройки параметрів регулятора: один із них - метод перехідного режиму. Данний метод був розроблений трьома вченими: Чином, Хронсом і Ресквіком. Цей метод дозволяє отримати більший запас стійкості аніж метод Циглера-Нікольса, але пропорційна складова в цих методах зазвичай менша, це слід враховувати в налаштуванні, оскільки надмірне зменшення пропорційної складової призведе до вигляду перехідного процесу з меншою точністю регулювання. Недоліком усіх експерементальних методик є неповнота інформації про запас стійкості системи та робастності. Запас стійкості та робастність визначає надійність роботи регулятора. В даному методі використовуються наступні параметри налаштування регулятора(для 0% перерегулювання):

Таблиця 4.1 – Параметри налаштування регуляторів

Регулятор	K	Ti	Td
П	0.3T/τ	-	-
ПІ	0.6T/τ	4τ	-
ПІД	0.95T/τ	2.4τ	0.42τ

Де τ – стала часу запізнення, T – стала часу.

Побудуємо перехідні характеристики для ПІ та ПІД регуляторів згідно даного методу:

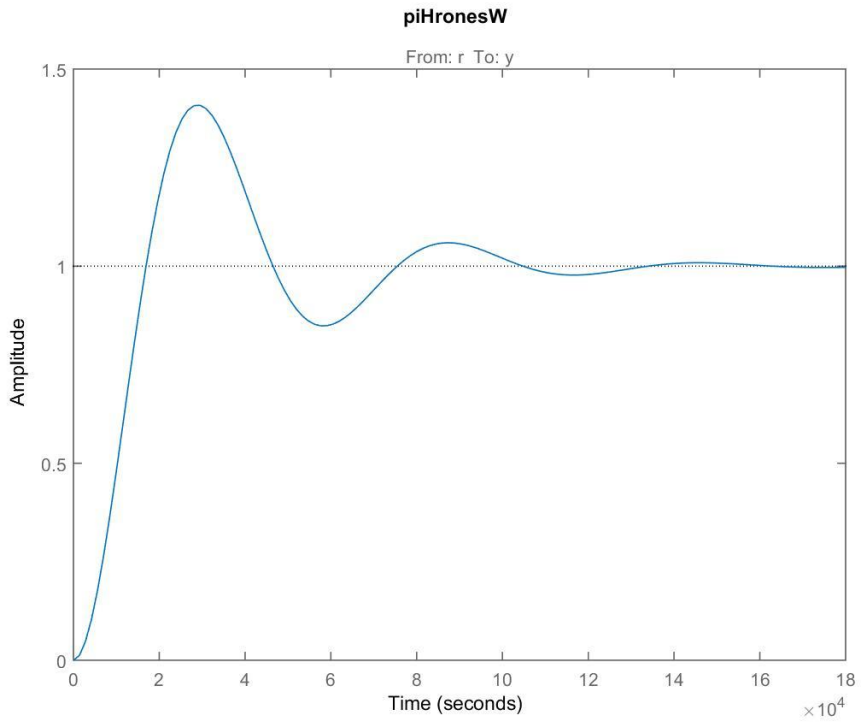


Рис. 4.8. – Перехідна характеристика системи з ПІ регулятором

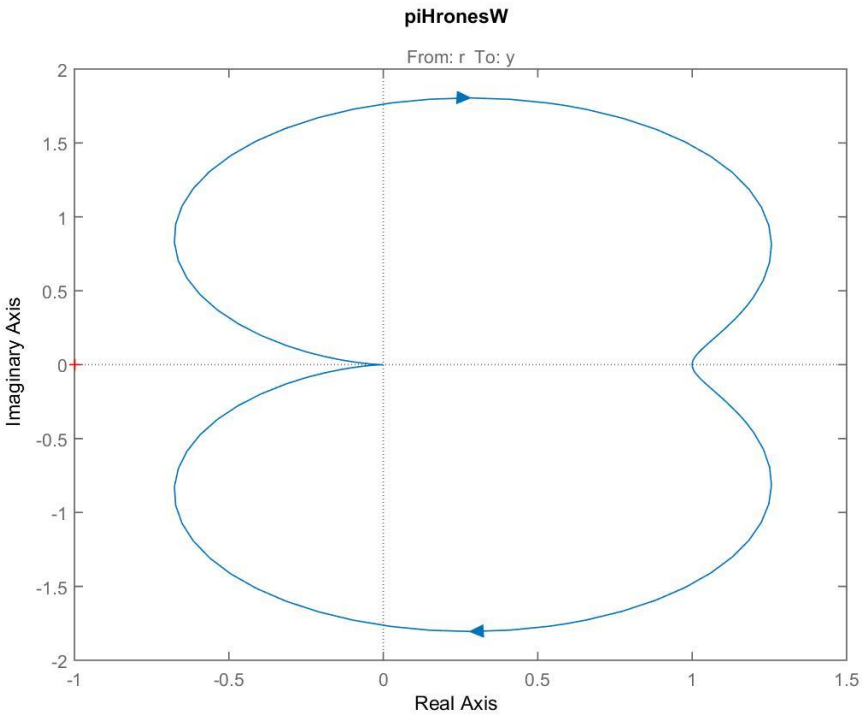


Рис. 4.9. – Годограф Найквіста системи з ПІ регулятором

Побудуємо перехідні характеристики для системи з налаштованим ПІД регулятором:

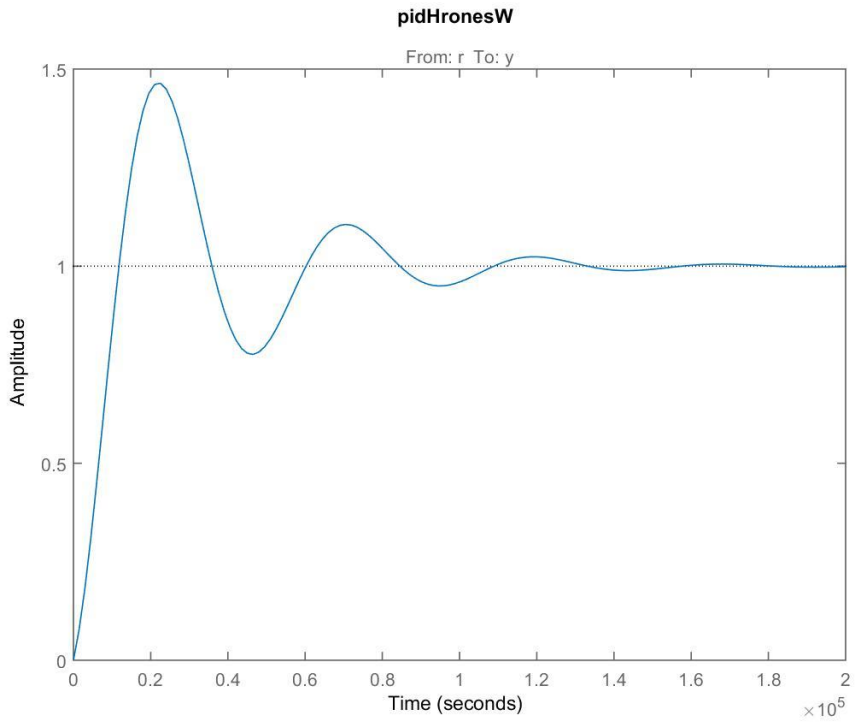


Рис. 4.10. – Перехідна характеристика системи з ПІД регулятором

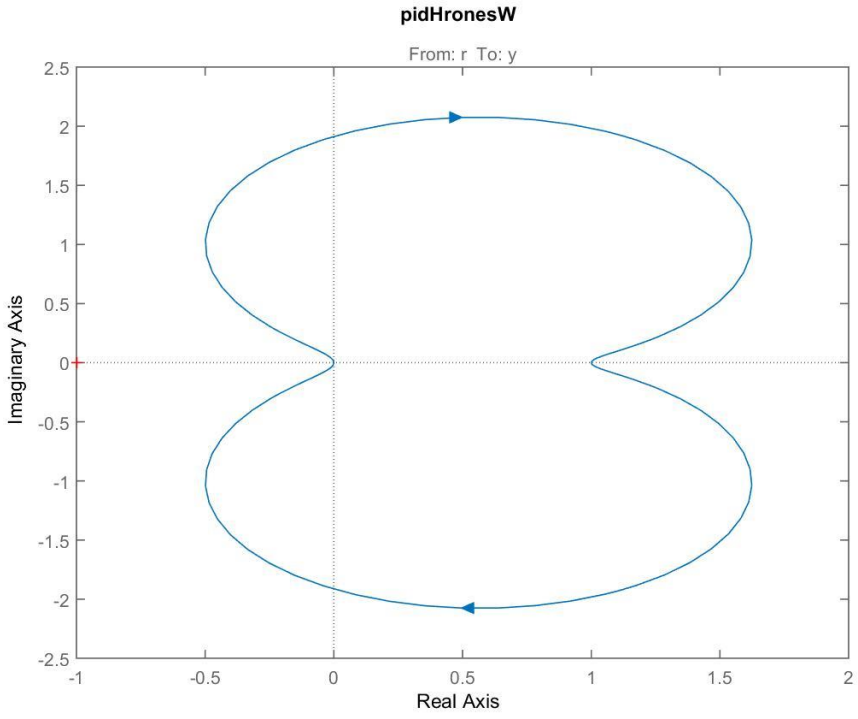


Рис 4.11. – Годограф Найквіста системи з ПІД регулятором.

Налаштувавши регулятори було отримано наступні параметри налаштування:

Таблиця 4.2 – Параметри налаштування регуляторів

	ПІ	ПІД
Kp	5.05	8
Ki	0.000255	0.0006
Kd	0	16703

Для заданих параметрів налаштування ми отримали наступні характеристики системи:

Таблиця 4.3 – Показники якості системи

	ПІ	ПІД
Час наростання, с	11380	8880
Час встановлення, с	121089	123719
Перерегулювання, %	40.83	46.42
Максимальне значення	1.4	1.46

Проаналізувавши значення, можемо зробити висновок, що в цілому даний метод дає кращі показники системи аніж метод Циглера-Нікольса, але все одно дані показники не є досить хорошими. Варто помітити, що в даному випадку використання ПІД регулятора – не є доцільним, оскільки налаштування ПІ регулятора нам дають кращі параметри системи.

4.3 Розрахунок параметрів регулятора за допомогою Sisotool

Для роботи з Control System Toolbox нам потрібно лише задати передавальну функцію в будь-якому з дозволених Matlab’ом виглядів (ss,tf,zpk,frd). Для зручності ми будемо використовувати tf форму, оскільки вона найбільш репрезентативна. Після введення передавальної функції в програмне середовище(або її імпорту з файлу) викликаємо інтерактивне середовище SisoTool за командою: sisotool(W).

Перед нами відкриється вікно редактора (рис. 4.12) з архітектурою системи та різноманітними закладками. Для початку роботи нам потрібно обрати архітектуру системи

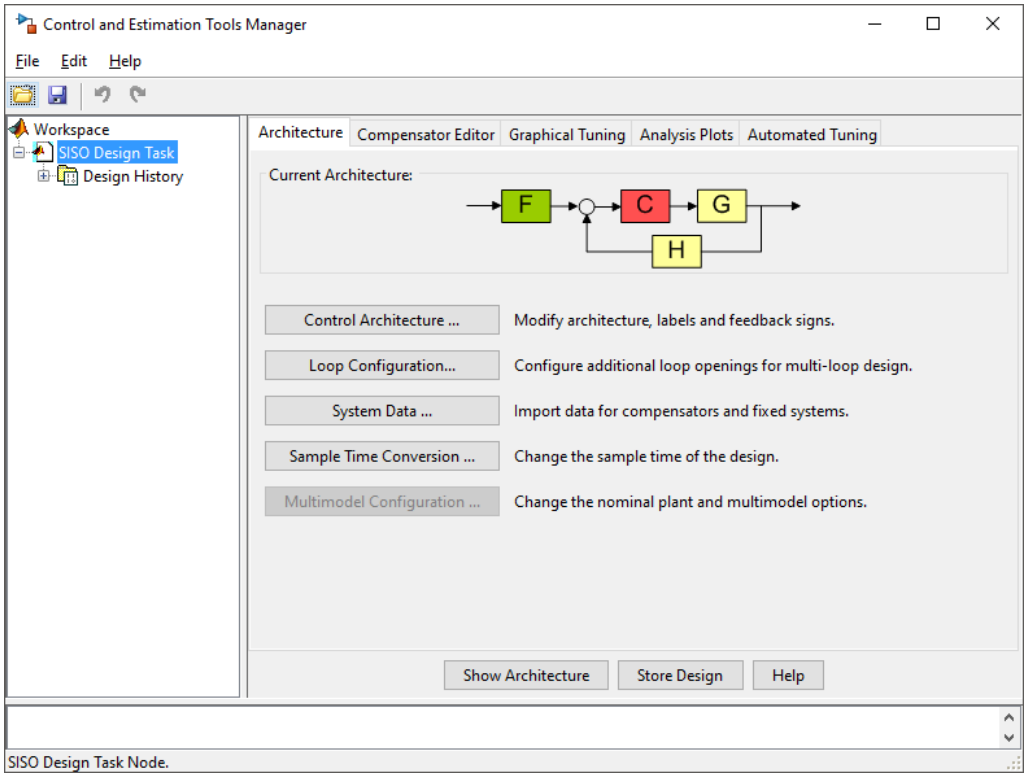


Рис. 4.12. – Вікно редактора

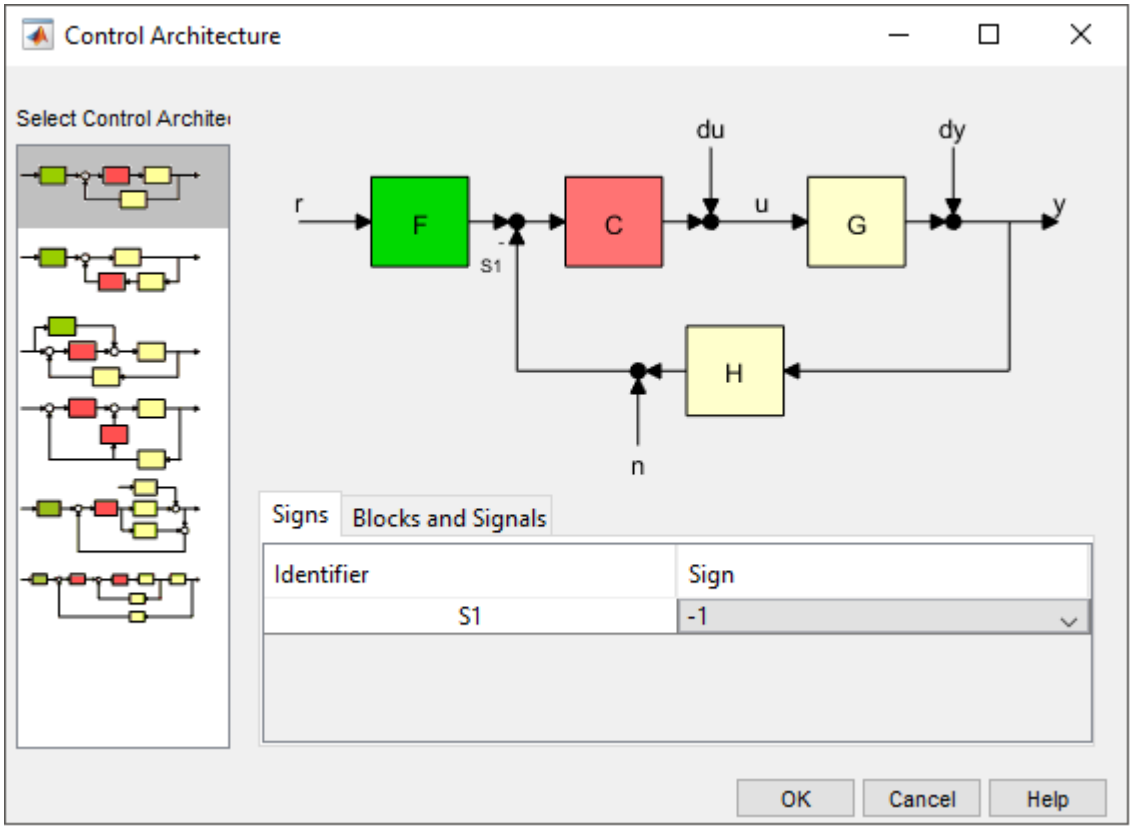


Рис. 4.13. – Вікно вибору архітектури

Тут прийняті наступні позначення:

G – plant (математична модель об’єкта),

H – sensordynamics (давач, що вимірює вихідну величину),

F – prefilter (фільтр),

C – compensator (регулятор).

4.3.1 Приблизна оптимізація коефіцієнту передачі інтегрального регулятора при m-звязках (Approximate MIGO)

Даний метод базується на інтерполяції результатів, отриманих методами чисельної оптимізації. Він був запропонований Хагландом і Острьомом у 2002 році і є деяким синтезом між класичним ATMO та методом MIGO. Для MIGO передавальна функція обєкта має бути відомою, у той час, як AMIGO використовує 3 параметри – коефіцієнт передачі, час запізнення та сталу часу і похідну в точці перегину, які отримуються з перехідної характеристики обєкта з самовирівнюванням.

Метою AMIGO є таке збільшення сталої інтегрування, при якій обмежується функція чутливості для досягнення балансу між якістю керування та робастністю. Метод використовується для обєктів з самовирівнюванням та без нього.

Для спрощення роботи використаємо SISO tool, враховуючи те, що даний метод вже реалізований в цій бібліотеці. Викликавши sisotool в середовищі Matlab за допомогою терміналу, вибираємо Compensator Editor попередньо задавши передавальну функцію в симульовану систему. Обираємо approximate MIGO step response(рис 4.19) попадаємо на наступну сторінку:

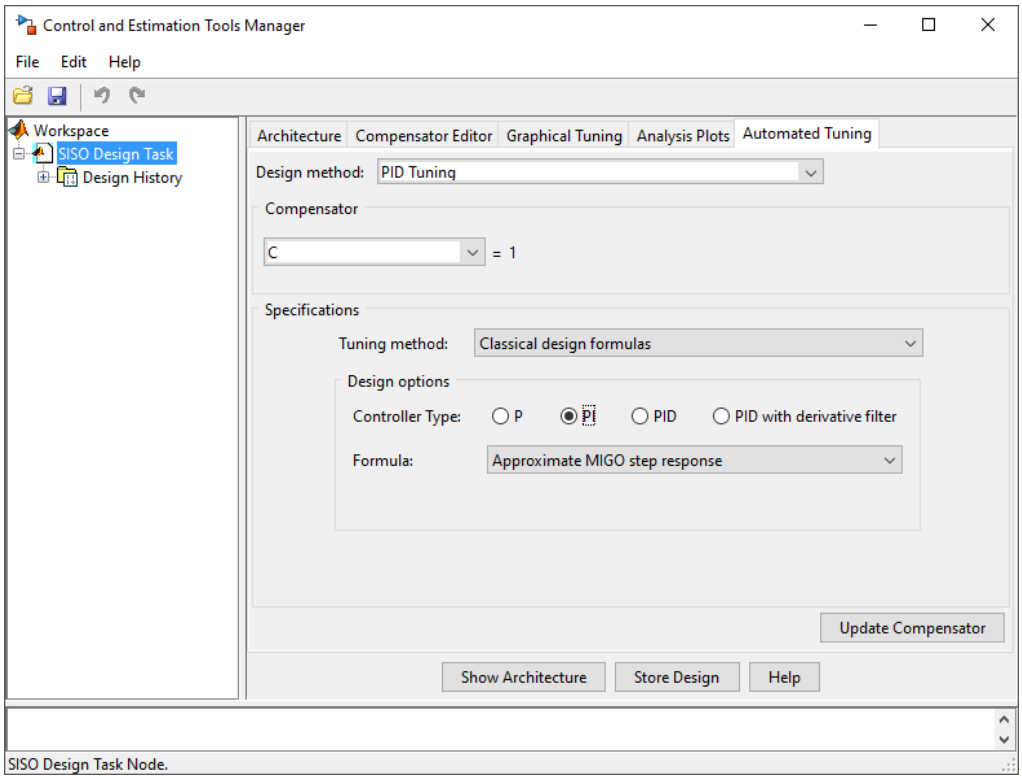


Рис. 4.14. – Вікно налаштування Sisotool

Побудуємо перехідні характеристики для данного методу для ПІ та ПІД регулятора:

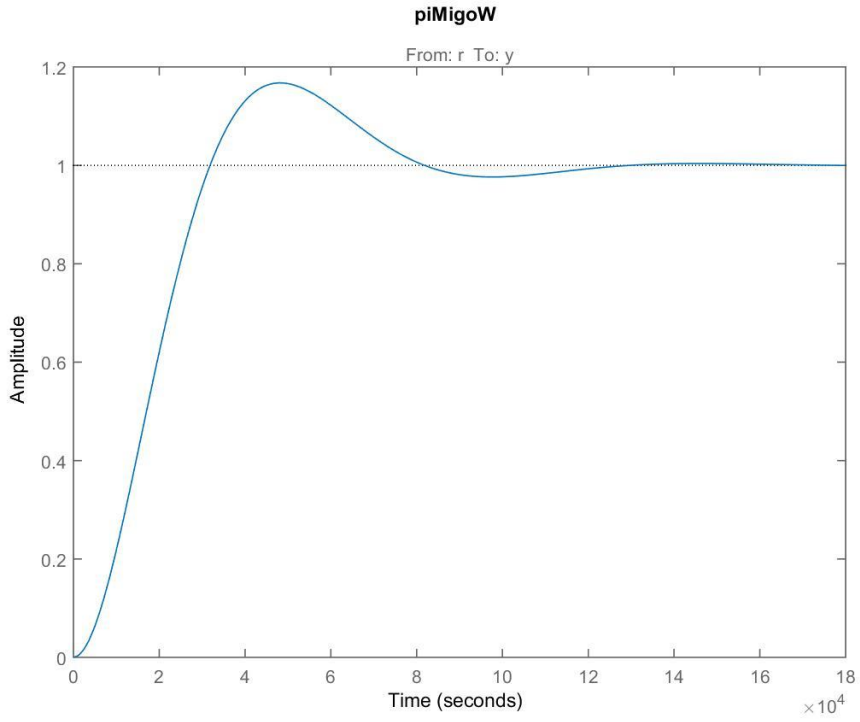


Рис. 4.15 – Перехідна характеристика системи з ПІ регулятором

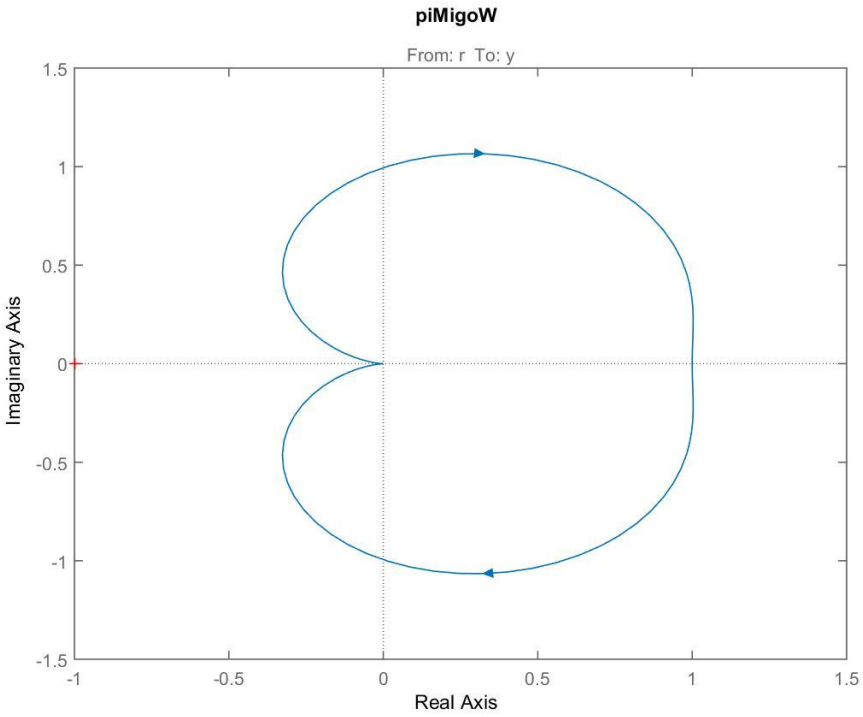


Рис. 4.16. – Годограф Найквіста системи з ПІ регулятором

Виконаємо аналогічні дії для ПІД регулятора:

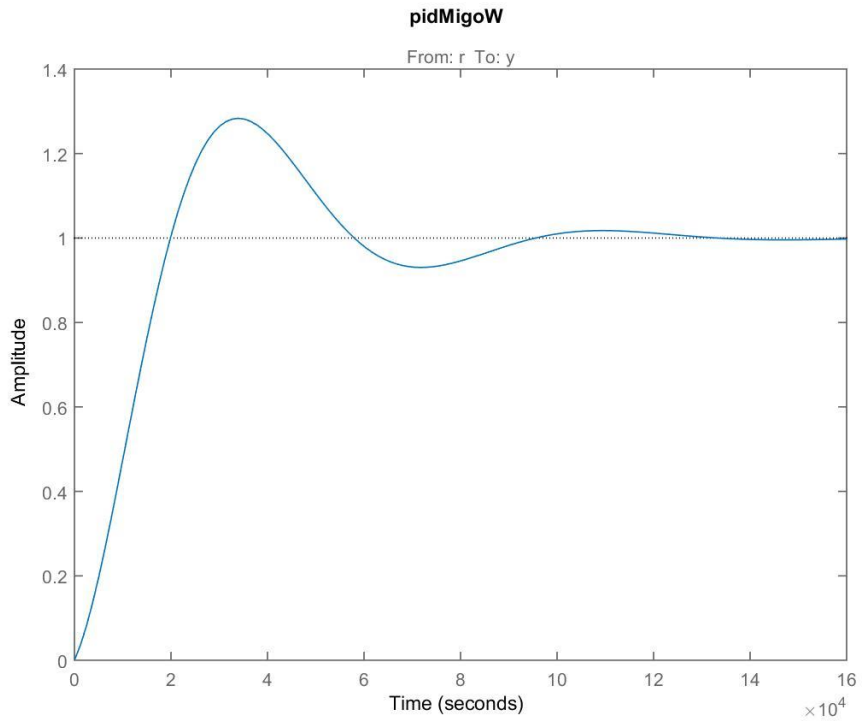


Рис. 4.17. – Перехідна характеристика системи з ПІД регулятором

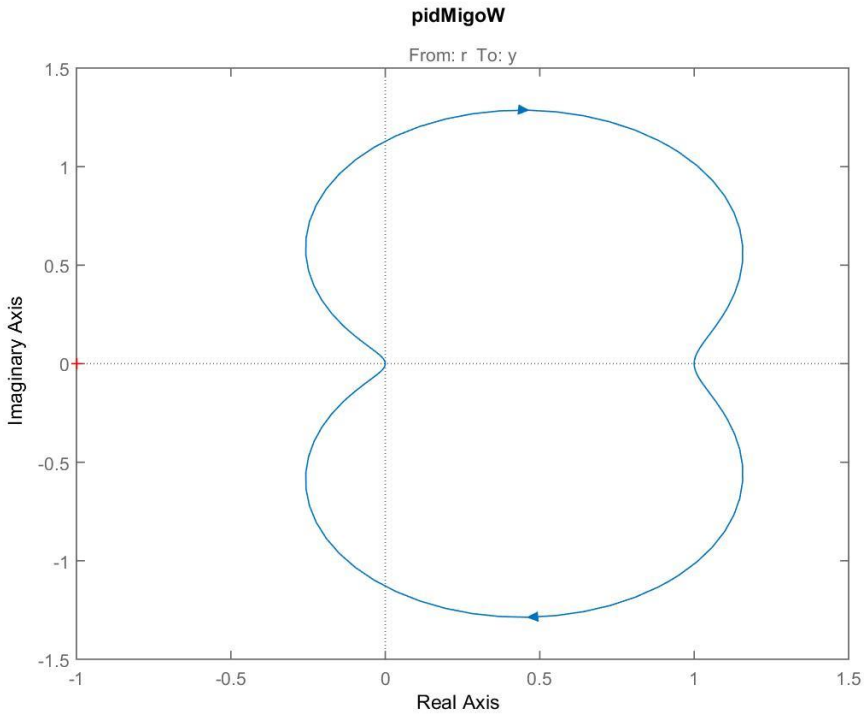


Рис. 4.18. – Годограф Найквіста системи з ПІД регулятором

Експортуємо отриману систему в окремий файл, це нам знадобиться для подальшого аналізу налаштувань. Ми отримали наступні налаштування регуляторів даним методом:

Таблиця 4.4 – Параметри налаштування регуляторів

	ПІ	ПІД
Kp	2.25	4
Ki	8.84e-05	0.00022
Kd	0	9594

Характеристики системи:

Таблиця 4.5 – Параметри якості системи

	ПІ	ПІД
Час наростання, с	21680	14761
Час встановлення, с	105736	89295
Перерегулювання, %	16.7	28.34
Максимальне значення	1.16	1.28

Даний метод дає кращі результаті аніж попередні. ПІ регулятор має перевагу над ПІД в даному випадку. В цілому параметри задовільні.

4.3.2 Синтез робастного регулятора

Робастне керування – сукупність методів теорії автоматтичного керування, метою яких є синтез такого регулятора, який би обеспечував високу якість управління (наприклад, запас стійкості). Якщо обєкт керування відрізняється від розрахованого, або його математична модель не відома, або погано досліджена. Таким чином, робастність означає, малу зміну виходу замкнутої системи керування при малій зміні параметрів обєкта керування.

Системи, котрі володіють властивістю робастності називаються робастними (грубими) системами. Зазвичай робастні контролери використовуються для керування обєктами з невідомою або неповною математичною моделлю і маючими певні невизначеності.

Для проектування робастних систем керування використовуються різноманітні методи оптимального та робастного синтезу, серед яких синтез контролерів в просторах H∞, H2, ЛМН – контролери, μ-контролери.

Метою робастного синтезу є проектування такого контролера, який би задовільняв критерій робастності. Починаючи з 50-х років XX століття було розроблено ряд процедур і алгоритмів, котрі дозволяють вирішити задачу робастного синтезу. Робастні системи керування можуть поєднувати в собі риси як класичного керування, так і адаптивного і нечіткого.

Нижче наведені основні технології синтезу робастних систем керування:

Таблиця 4.6 – оптимальні методи налаштування регуляторів

Назва	Переваги	Недоліки

H∞	Працює як зі стійкістю, так і з чутливістю системи, замкнутий контур завжди стійкий, прямий однопрохідний алгоритм синтеза	Потребує особливої уваги до параметричної робастності обєкту керування
H2	Працює як зі стійкістю, так і з чутливістю системи, замкнутий контур завжди стійкий, точне формування передавальної функції контроллера	Велика кількість ітерацій
LQG	Використання доступної інформації про шуми	Не гарантуються запаси стійкості, треба точна модель об’єкту, велика кількість ітерацій
LQR	Гарантоване забезпечення робастної стійкості, безінерційний регулятор	Треба зворотній зв'язок по всьому вектору станів, треба точна модель обєкту, велика кількість ітерацій
M	Працює з широким классом невизначеностей	Великий порядок контролера

Будемо використовувати автоматичний метод для синтезу робастного регулятора запропонований SISO tool. Для синтезу обираємо баланс між швидкодією та робастністю.

Побудуємо перехідні характеристики системи з ПІ та ПІД регуляторами:

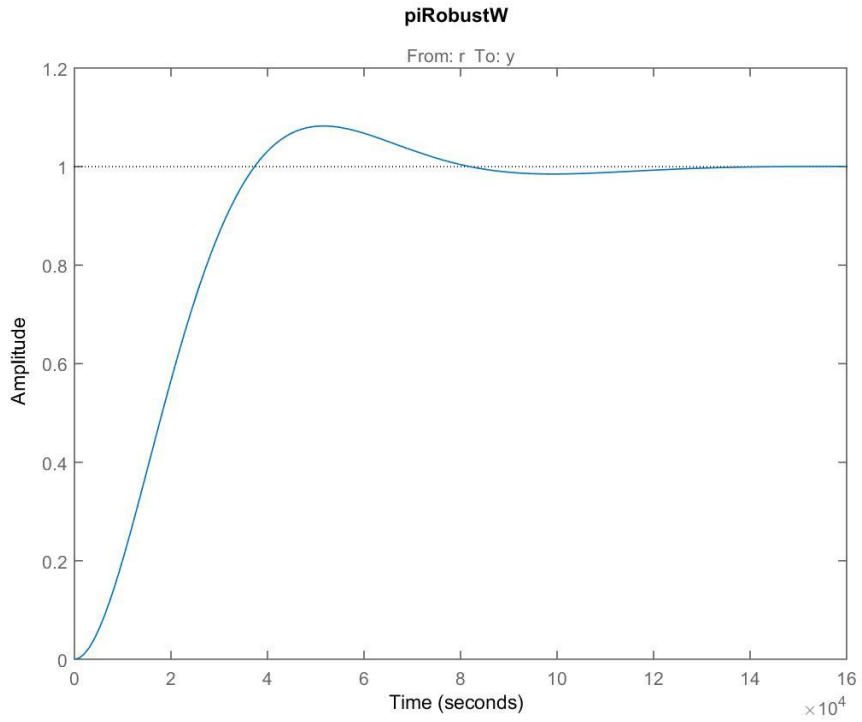


Рис. 4.19. – Перехідна характеристика системи з ПІ регулятором

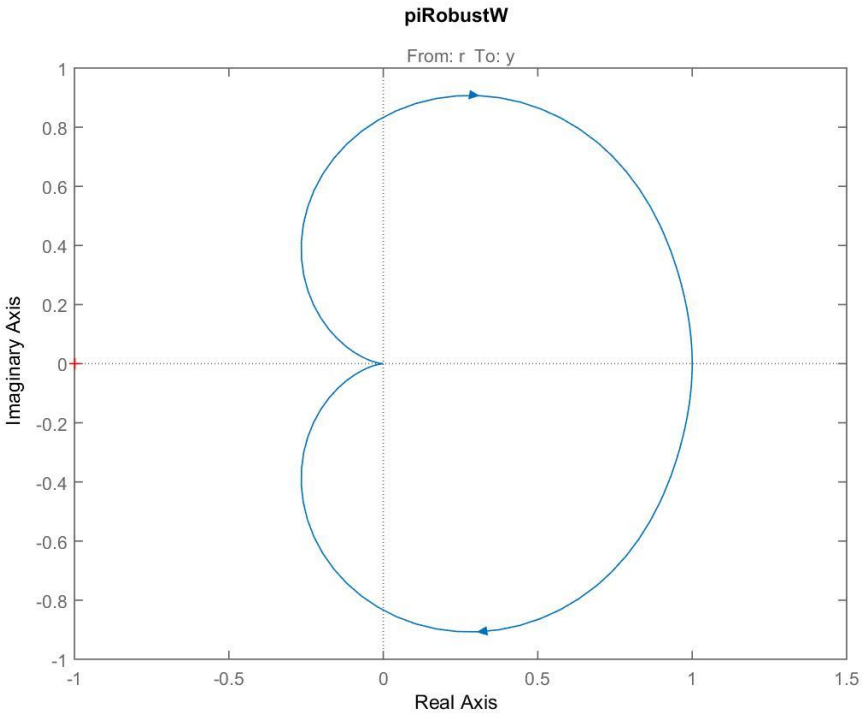


Рис. 4.20 – Годограф Найквіста системи з ПІ регулятором

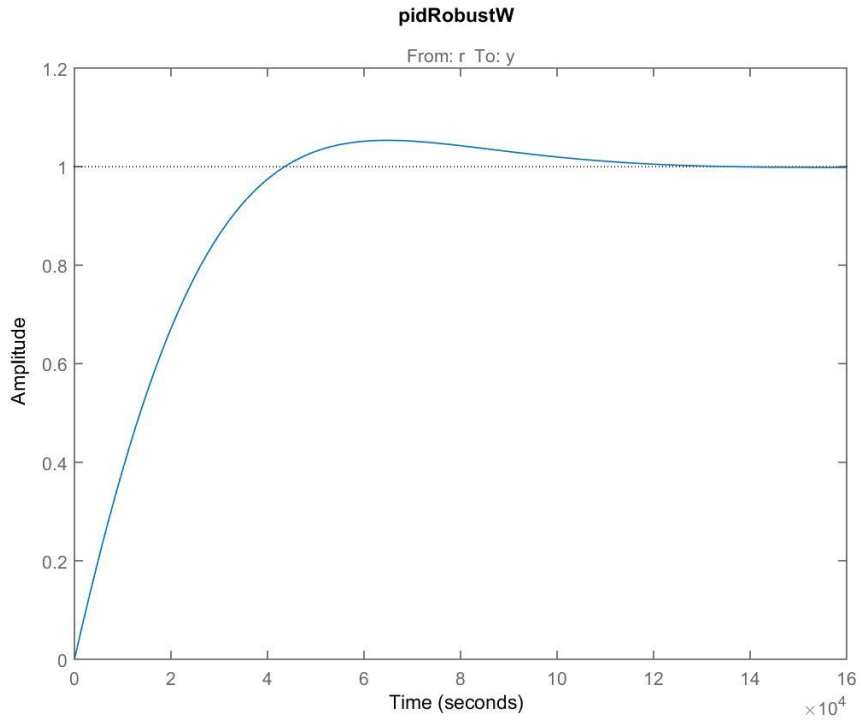


Рис. 4.21 Перехідна характеристика системи з ПІД регулятором

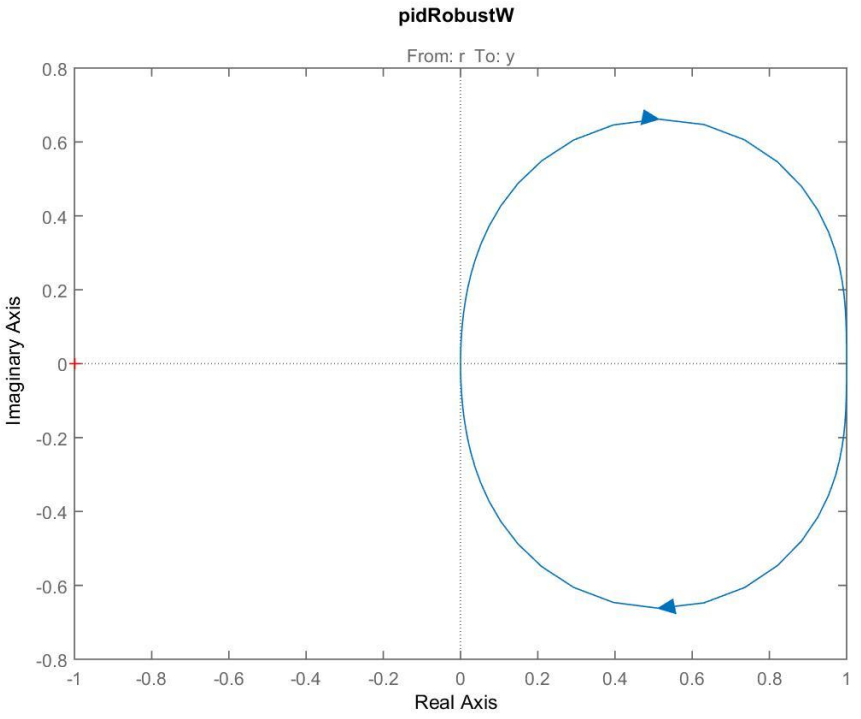


Рис. 4.22 – Годограф Найквіста системи з ПІД регулятором

Отримали наступні налаштування регуляторів:

Таблиця 4.7 – Параметри налаштування регулятора

	ПІ	ПІД

Kp	2.12	2.36
Ki	6.65e-05	8.46e-06
Kd	0	16474

Для даних налаштувань регуляторів маємо наступні характеристики системи:

Таблиця 4.8 – Показники якості системи

	ПІ	ПІД
Час наростання, с	24877	30382
Час встановлення, с	74099	99611
Перерегулювання, %	8.25	5.34
Максимальне значення	1.08	1.053

Бачимо, що отримані характеристики кращі аніж отримані раніше, це повязано з тим, що в даному випадку Matlab використовує оптимальні методи. До того ж потрібно враховувати, що на відміну від інших методів, цей є аналітичним.

4.4. Розрахунок та моделювання системи керування в Simulink

Моделювання є одним із найбільш могутніх засобів дослідження складних систем на сьогоднішній день.

Сьогодні комп'ютерна промисловість пропонує сучасному інженерові цілий ряд різноманітних засобів моделювання, що дозволяють не тільки моделювати складні динамічні системи, але і проводити з ними експерименти. Найбільш повне дослідження загальносистемних проблем виходить у результаті моделювання об'єктів за допомогою сучасних технологій, реалізованих у спеціалізованих обчислювальних пакетах візуального моделювання.

Сучасна комп'ютерна математика пропонує цілий набір інтегрованих програмних систем і пакетів програм для автоматизації математичних розрахунків, одним з яких є MATLAB. MATLAB — одна з найстарших, ретельно пророблених й

перевірених часом систем автоматизації математичних розрахунків, яка побудована на розширеному представленні й застосуванні матричних операцій.

Серед великого числа пакетів візуального моделювання пакет Matlab займає особливе місце. Спочатку орієнтований на дослідницькі проекти, пакет в останні роки став робочим інструментом інженерів, студентів, керівників, фізиків, зв'язківців. Однією з основних причин широкого використання пакета Matlab є великий спектр засобів, що надає користувачеві для рішення різноманітних задач у різних областях людської діяльності. Серед цих засобів особливе місце займає підсистема Simulink.

Simulink - це інтерактивне середовище для моделювання й аналізу широкого класу динамічних систем за допомогою блок-діаграм.

Основні властивості підсистеми Simulink:

- містить у собі велику бібліотеку блоків (безупинні елементи, дискретні елементи, математичні функції, нелінійні елементи, джерела сигналів, засоби відображення, додаткові блоки), які можна використовувати для графічного збирання систем;
- надає можливість моделювання лінійних, нелінійних, безупинних, дискретних і гібридних систем;
- блок-діаграми можуть бути об'єднані в складені блоки, що дозволяє використовувати ієрархічне представлення структури моделі, тим самим забезпечуючи спрощений погляд на компоненти і підсистеми ;
- містить засоби для створення користувальницьких блоків і бібліотек блоків;
- підтримує підсистеми, що працюють за умовами, тригерів.

Simulink забезпечують інтерактивне середовище для моделювання, при цьому поводження моделі і результати її функціонування відображаються в процесі роботи, і існує можливість змінювати параметри моделі навіть у той момент, коли вона виконується. Simulink дозволяє створювати власні блоки і бібліотеки блоків з доступом із програм на Matlab, Fortran чи C, зв'язувати блоки з розробленими раніше програмами на Fortran і C, що містять вже перевірені моделі.

Для забезпечення якості перехідного процесу виконаємо комп'ютерне моделювання звичайного ПІ-регулятора. Для цього у робочому вікні Simulink необхідно зробити систему керування з ПІ-регулятором, яка має вигляд схеми, що складається з таких блоків:

1) *Transfer Fcn* – блок передатної функції Transfer Fcn задає передатну функцію у вигляді відношення поліномів:

$$H(s) = \frac{y(s)}{u(s)} = \frac{num(s)}{den(s)}, \tag{4.9}$$

де *num* - вектор або матриця коефіцієнтів чисельника, den - вектор коефіцієнтів знаменника.

2) *Transport Delay* – забезпечує затримку вхідного сигналу на заданий час.

3) *PID Controller* – з його допомогою можна реалізувати найпопулярніший в промисловості алгоритм управління – ПІД-регулятор. Завдання налаштування ПІД-регулятора в загальному випадку зводиться до підбору трьох коефіцієнтів підсилення. Це можна робити методом проб і помилок, а можна використовувати Simulink Control Design для автоматичної лінеаризації і підбору параметрів, задаючи, наприклад, бажану швидкодію системи. Цей блок після його налаштування, реалізовує передатну функцію об`єкта.

4) *Scope* – будує графіки досліджуваних сигналів у часі. Дозволяє спостерігати за змінами сигналів в процесі моделювання. Для того, щоб відкрити вікно перегляду сигналів необхідно виконати подвійне натискання лівою клавішею "миші" на зображенні блоку. Це можна зробити на будь-якому етапі розрахунку, що є доволі зручним.

5) *Sum* – виконує обчислення суми поточних значень сигналів. Блок може використовуватися для підсумовування скалярних, векторних або матричних сигналів. Типи сигналів повинні збігатися. Якщо кількість входів блоку більше, ніж один, то блок виконує поелементні операції над векторними і матричними сигналами. При цьому кількість елементів в матриці або векторі має бути однаковим.

6) *Constant* – задає постійний за рівнем сигнал. Значення константи може бути дійсним або комплексним числом, обчислюваним виразом, вектором або матрицею.

Побудовану схему системи керування з ПІ-регулятором можна побачити на рис. 4.23. В ролі передатної функції використовується отримана раніше передатна функція по керуванню.

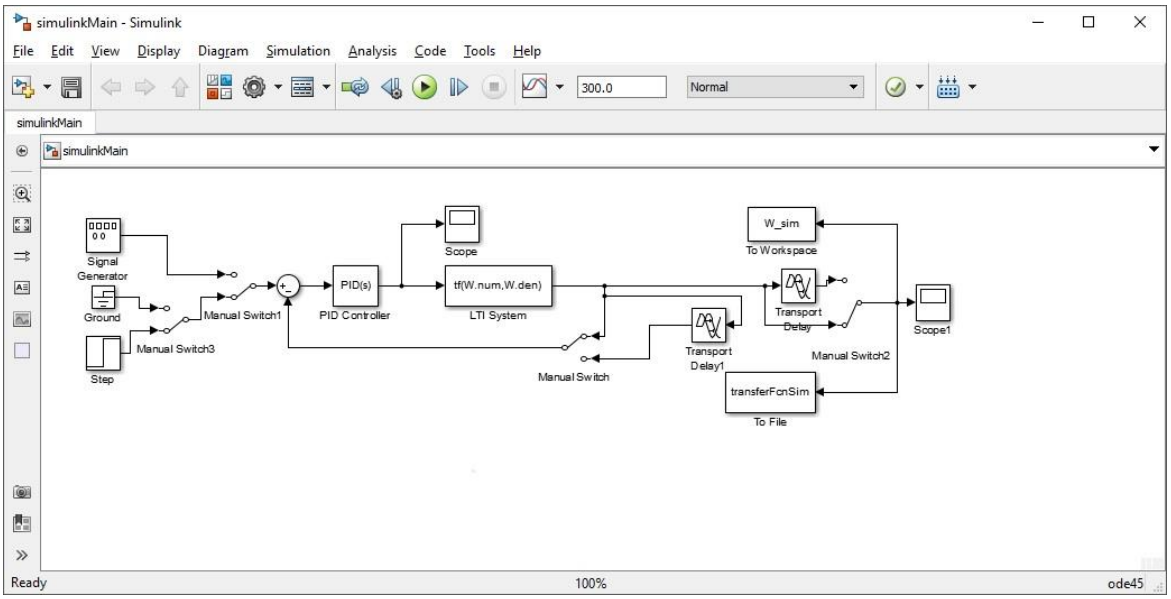


Рис. 4.23 – Схема системи керування з ПІ-регулятором у середовищі Simulink

Натиснувши на блок «*PID Controller*» можна виконати налаштування регулятора. Для того щоб був ПІ-регулятор необхідно вибрати його після побудови схеми у розділі налаштувань «*Controller*». Як видно з наведеного нижче рисунка параметри регулятора вибираються автоматично.

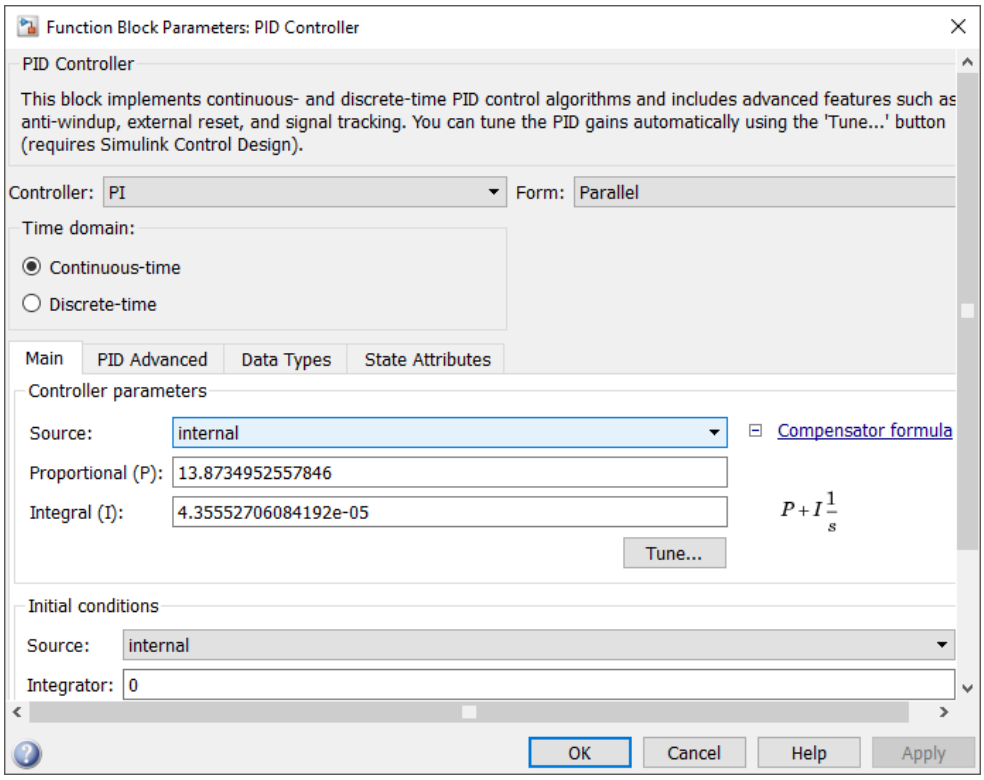


Рис. 4.24 – Вікно налаштувань ПІ-регулятора

Натиснувши на кнопку «Tune», отримаємо зображення графіка якості перехідного процесу:

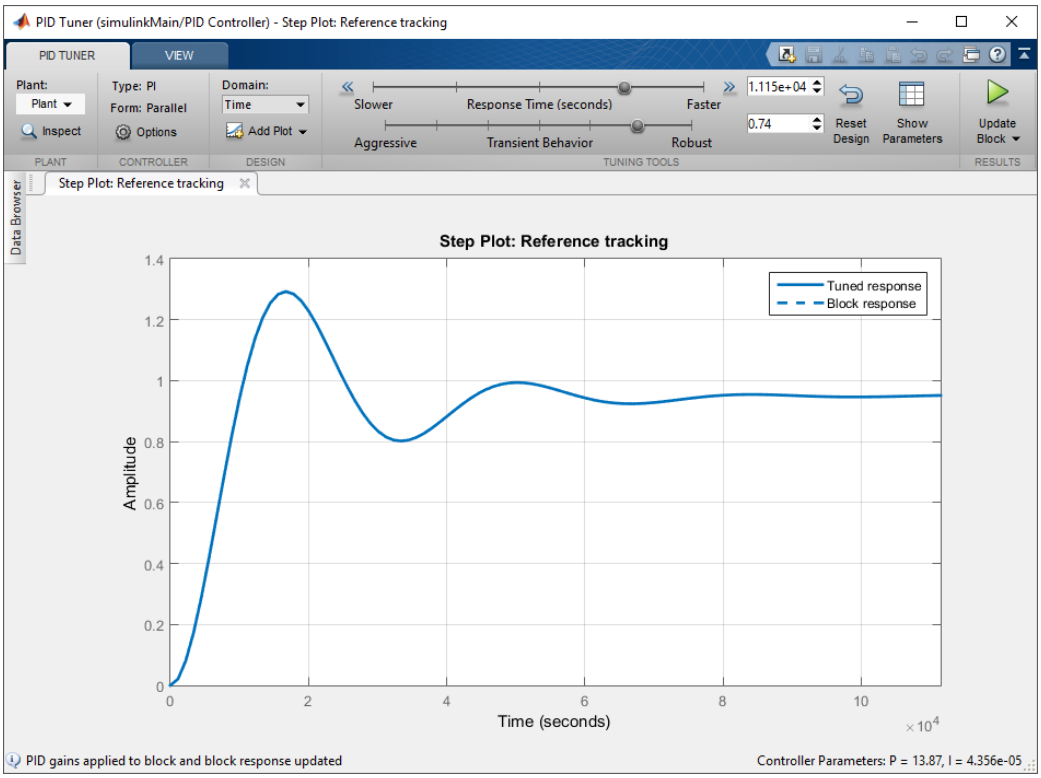


Рис. 4.25. – Графік якості перехідного процесу

У цьому вікні ми можемо налаштувати час та плавність переходу системи, при яких вона вийде на усталений режим. З правої сторони вікна можна спостерігати час наростання (rise time), час встановлення (settling time) та стабільність роботи замкнутого контуру (closed-loop stability). Налаштуємо ПІД регулятор аналогічним методом, маємо:

Параметри налаштування регулятора:

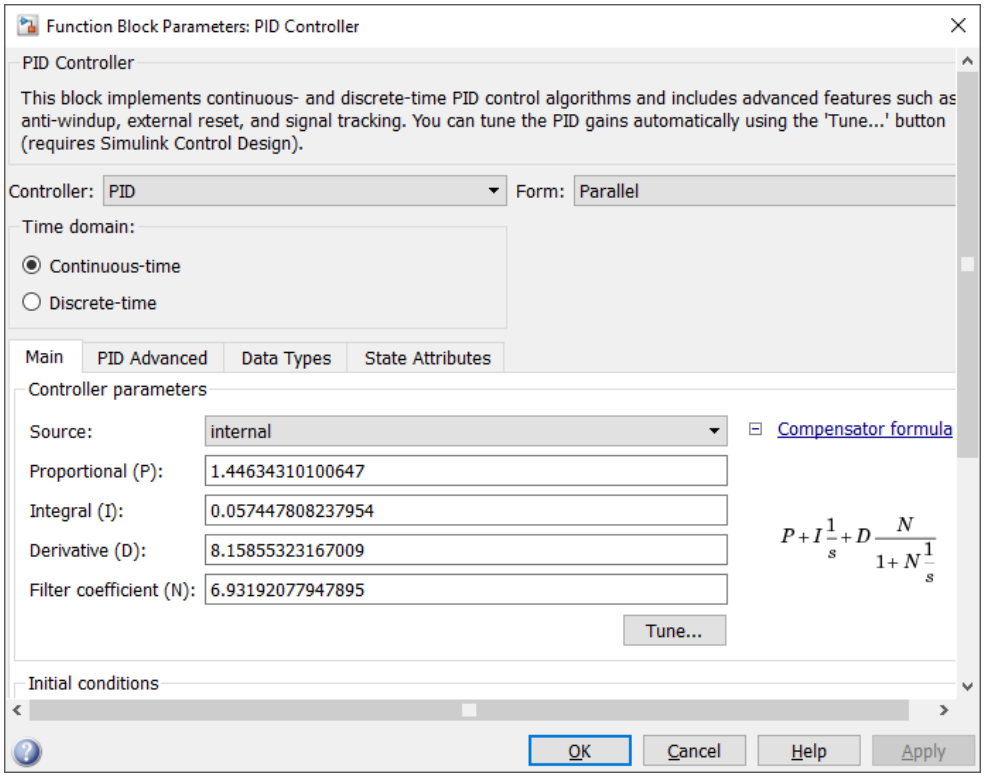


Рис. 4.26. – Вікно параметрів налаштування регулятора

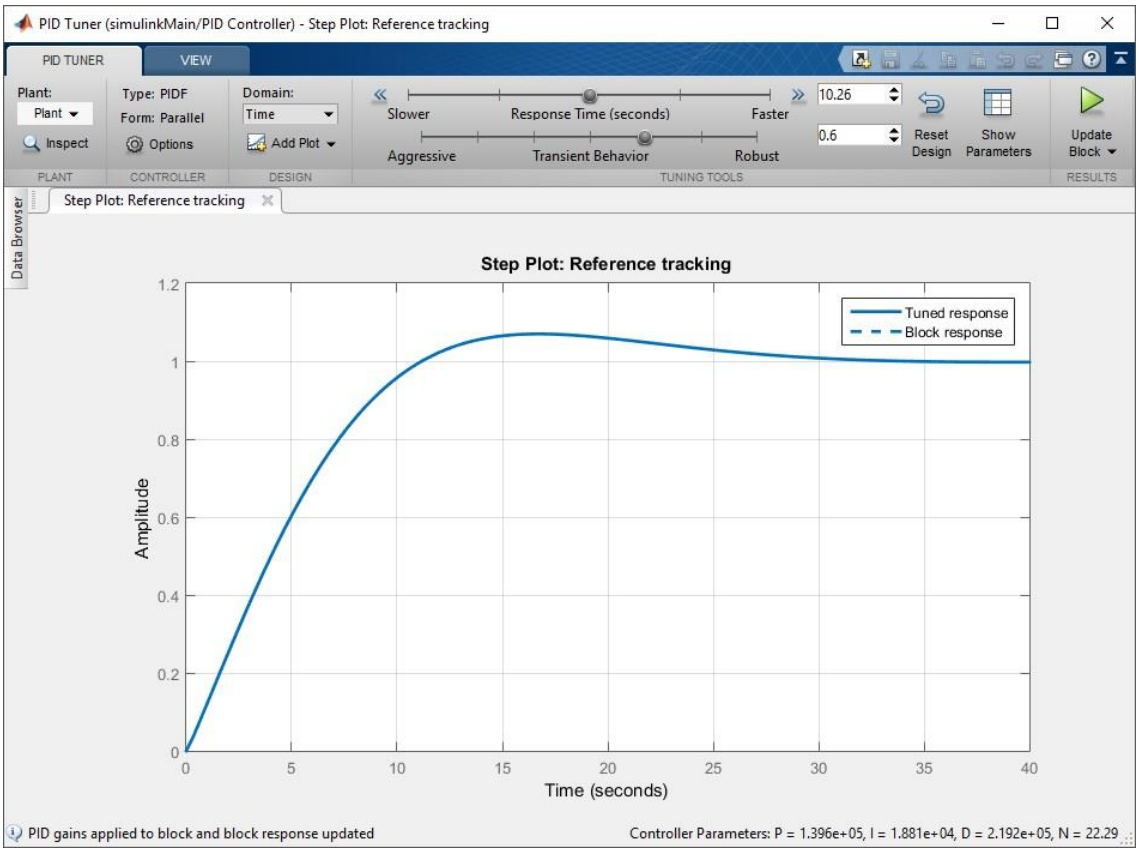


Рис. 4.27. – Перехідна характеристика системи з під регулятором

4.5 Аналіз параметрів налаштування регуляторів

Маючи велику кількість можливостей налаштування регуляторів спеціаліст з автоматизації стикається з питанням, який метод буде найкращим та найпростішим.

Отримавши таку кількість параметрів регуляторів з’явилась необхідність ці данні проаналізувати, та виявити найкращий з методів для даного процесу.

Для вирішення поставленої задачі було розроблено декілька програм, котрі спрощують роботу: для побудови перехідних характеристик та для запису важливих данних в окремий файл та виведення в xls таблицю. Нижче наведений лістінг програм:

Лістинг скрипта iterationsMethod.m, котрий записує необхідні дані в таблицю наведено в додатку 2.

Побудуємо на одному графіку перехідні характеристики для систем з ПІ регулятором та з ПІД:

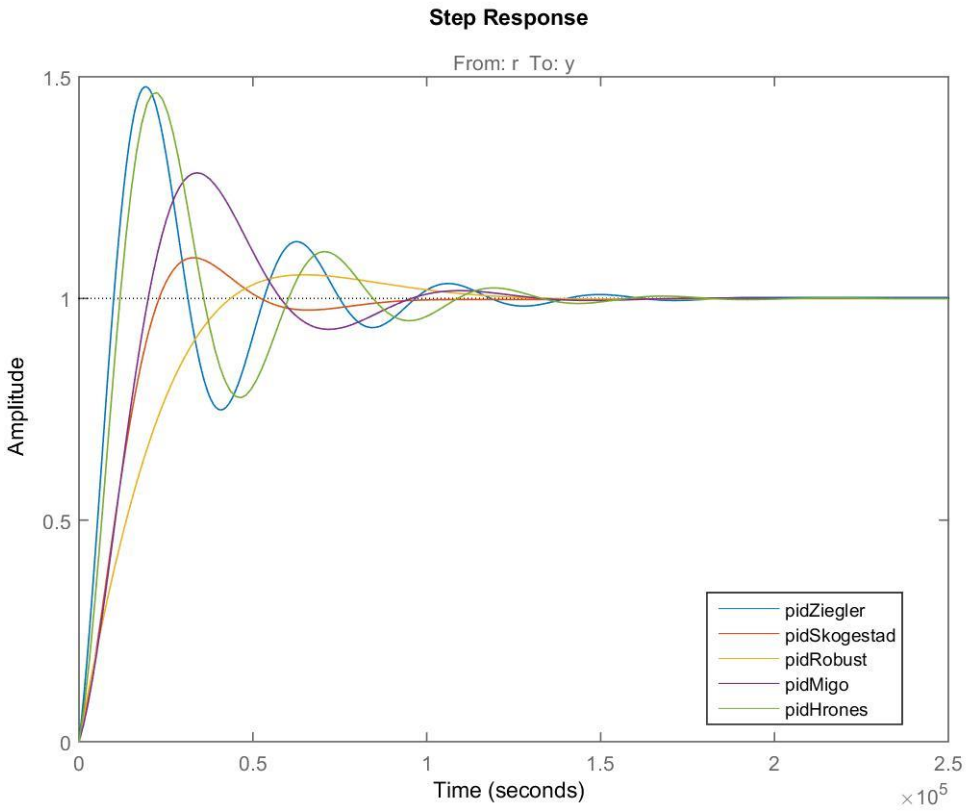


Рис. 4.28. – Перехідні характеристики систем з ПІ регулятором

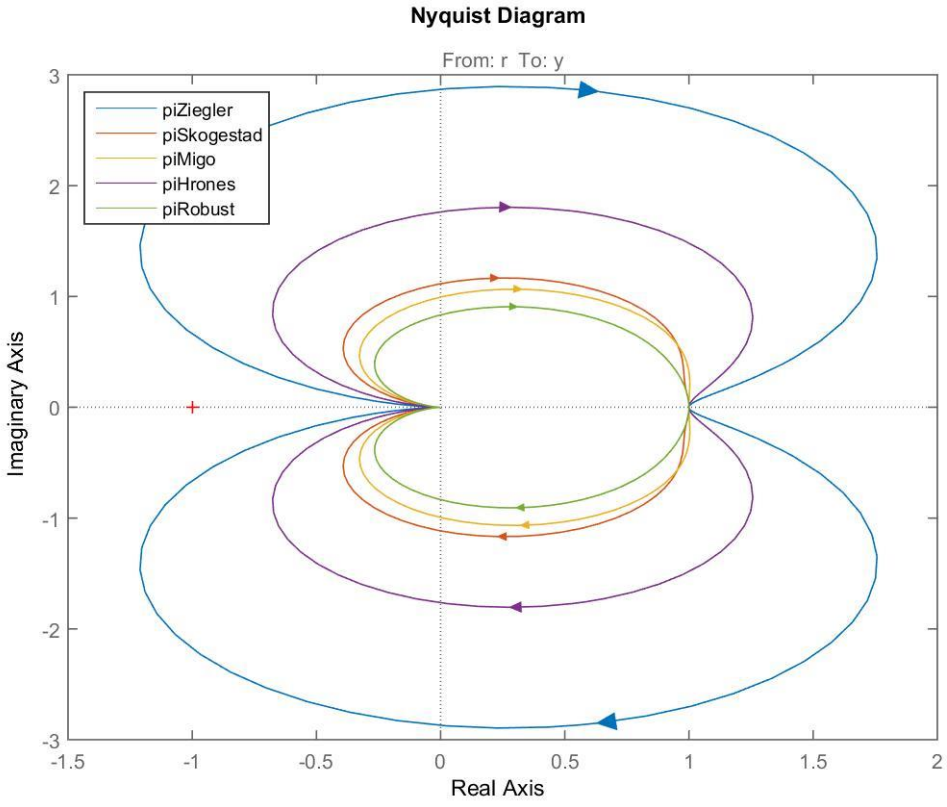


Рис. 4.29. – Годографи Найквіста систем з ПІ регулятором

Для ПІД регуляторів:

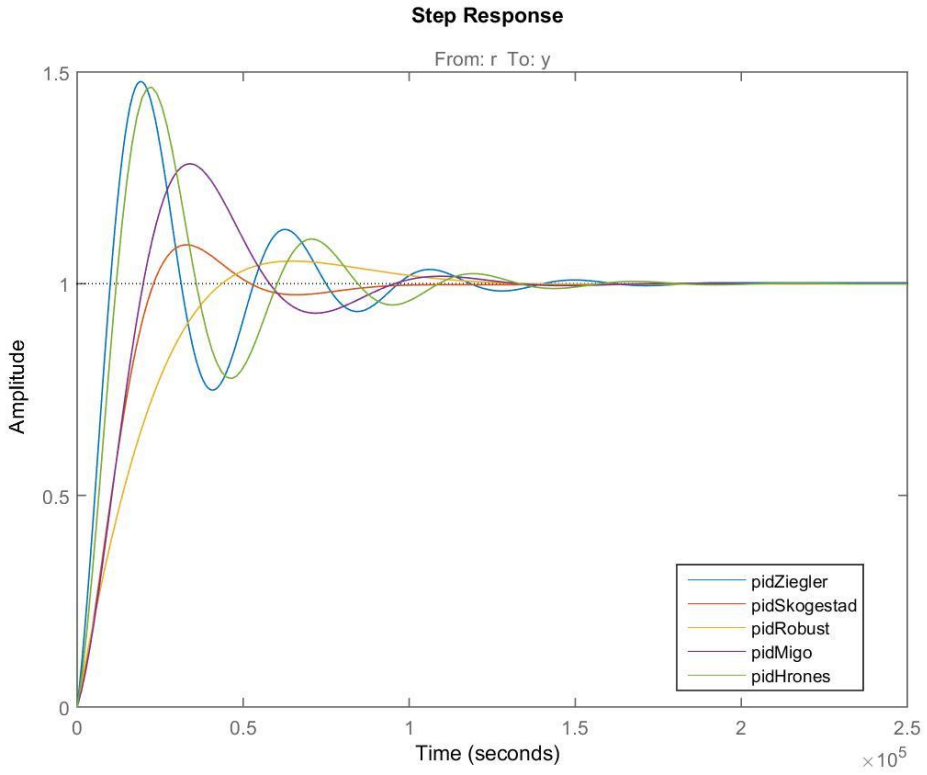


Рис. 4.30. – Перехідні характеристики систем з ПІД регулятором

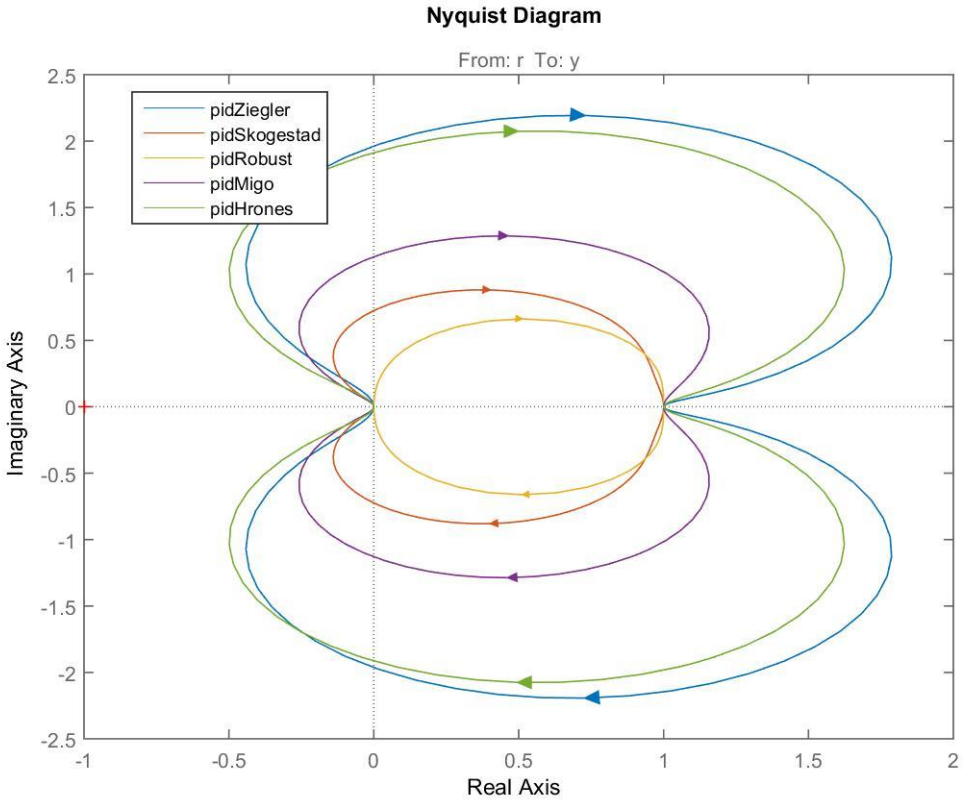


Рис. 4.31. – Годографи Найквіста систем з ПІД регулятором

Сформуємо таблицю налаштувань регуляторів:

Таблиця 4.9 – налаштування регуляторів різними методами

Метод	Кр	Кі	Kd
ПІ Циглер-Нікольс	7.58	0.000509	-
ПІ MIGO	2.24	8.84e-05	-
ПІ CHR	5.05	0.00025	-
ПІ Автоматичний	2.12	6.65e-05	-
ПІД Ц-Н	10.11	0.001018	25517
ПІД MIGO	4.018	0.000222	9545
ПІД CHR	8.007	0.00067	16703
ПІД Автоматичний	2.3617	8.46e-05	16512

Для даних регуляторів, система має наступні характеристики:

Таблиця 4.10 – характеристики систем налаштованих різними методами

Метод	Час наростання, с	Час Встановлення, с	Перерегулювання,%
ПІ Циглер-Нікольс	8493	145087	58.38
ПІ MIGO	21609	105736	16.77
ПІ CHR	11380	121089	40.83
ПІ Автоматичний	24877	74094	8.25
ПІД Ц-Н	7645	112946	47.77
ПІД MIGO	14761	89295	28.34
ПІД CHR	8800	123719	46.42
ПІД Автоматичний	30382	99611	5.34

Проаналізувавши перехідні характеристики та отримані дані можемо сказати, що найкраще з поставленою задачею справився автоматичний метод налаштування, а якщо бути більш конкретними – його ПІ налаштування.

4.6. Синтез лінійно-квадратичного Гауссового регулятора

Лінійно-квадратичний гауссівський регулятор - набір методів і математичного апарату теорії управління для синтезу систем управління з від’ємним зворотним зв'язком для лінійних систем з адитивним гауссовских шумом.Синтез проводиться шляхом мінімізації заданого квадратичного функціоналу.

Лінійно-квадратичний гауссівский (ЛКГ) регулятор відноситься до сучасних методів управління. Методологія синтезу контролера дозволяє віднести системи управління, побудовані на такому принципі, до оптимальних систем, в яких оптимізація проводиться за деяким заданим квадратичним критерієм якості. Також ця теорія бере до уваги наявність збурень у вигляді гауссовського білого шуму. Однак незважаючи на те, що синтез ЛКГ-контролерів передбачає систематичну процедуру розрахунку для оптимізації якості системи, головним його недоліком є те, що в розгляд не приймається робастність системи. Тому ЛКГ-синтез проводиться тільки для систем, що мають надійну і точну лінійну динамічну модель. Для підвищення робастності системи управління застосовують більш складні алгоритми, такі як мінімаксний ЛКГ синтез, або комібінованний ЛКГ синтез. ЛКГ контролери можуть використовуватися як для дискретних, так і для безперервних систем.

Лінійний квадратичний Гауссівський (LQG) регулятор – сучасний метод синтезу оптимальних динамічних регуляторів у просторі станів. Такий регулятор складається з матриці оптимальних коефіцієнтів зворотних зв’язків за змінними стану та естиматора на основі фільтра Калмана.Такий регулятор описується векторно – матричним рівнянням:

$$\hat{x} = [A - LC - (B - LD)K]\hat{x} + Ly$$

$$u = -K\hat{x}$$

На першому етапі синтезу шукаються параметри зворотніх зв’язків за змінними стану, що мінімізують функціонал якості системи :

$$J(u) = \int_0^\infty \{x^T Qx + 2x^T Nu + u^T Ru\}dt$$

Матриця коефіцієнтів зворотніх зв’язків K отримується розв’язанням алгебраїчного рівняння Ріккаті:

$S \times A + A' \times S - (S \times B + N) \times R^{-1} \times (B' \times S + N') + Q = 0$

Отриманні значення коефіцієнтів підсилення називаються LQ-оптимальним підсиленням. На наступному кроці визначаються параметри солвера координат простору стану для реалізації зворотніх зв’язків за змінними стану, які неможливо виміряти безпосередньо. Таким естиматором є фільтр Калмана.

Фільтр Калмана - ефективний рекурсивний фільтр, що оцінює вектор стану динамічної системи, використовуючи ряд неповних та зашумлених вимірювань. Названий на честь Рудольфа Калмана.

Фільтр Калмана широко використовується в інженерних та економетричних роботах: від радарів і систем технічного зору до оцінок параметрів макроекономічних моделей. Калманова фільтрація є важливою частиною теорії управління, відіграє велику роль у створенні систем управління. Спільно з лінійно-квадратичним регулятором фільтр Калмана дозволяє вирішити задачу лінійно-квадратичного гауссівського управління. Фільтр Калмана і лінійно-квадратичний регулятор - можливе рішення більшості фундаментальних завдань в теорії управління. У більшості робіт, кількість параметрів, які задають стан об'єкта, більше, ніж кількість спостережуваних параметрів, доступних для вимірювання. За допомогою моделі об'єкта по ряду доступних вимірів фільтр Калмана дозволяє отримати оцінку внутрішнього стану.

Фільтр Калмана призначений для рекурсивної дооцінки вектора стану апріорно відомою динамічної системи, тобто для розрахунку поточного стану системи необхідно знати поточний вимір, а також попередній стан самого фільтра. Таким чином фільтр Калмана, як і безліч інших рекурсивних фільтрів, реалізований у часовому поданні.

На третьому кроці синтезу на основі одержаних даних з двох попередніх кроків формується LQG – регулятор.

Для синтезу ЛКГ регулятора був використаний Control System Toolbox. Для порівняння налаштуємо два регулятора: один зі збалансованими показниками швидкодії і робастності, інший – з максимальним значенням швидкодії. Значення шуму обираємо середнє, порядок регулятора – другий.

Отримаємо наступні передавальні функції регуляторів:

Збалансований:

```
C =

  -64.553 (s+2.919e-05) (s-8.336)
  -------------------------------
           s (s+752.9)
```

З максимальним значенням швидкодії:

```
C =

  7405 (s^2 + 0.00881s + 3.851e-05)
  ---------------------------------
    s (s^2 + 0.05084s + 0.001292)
```

Побудуємо перехідні характеристки для систем з даними регуляторами:

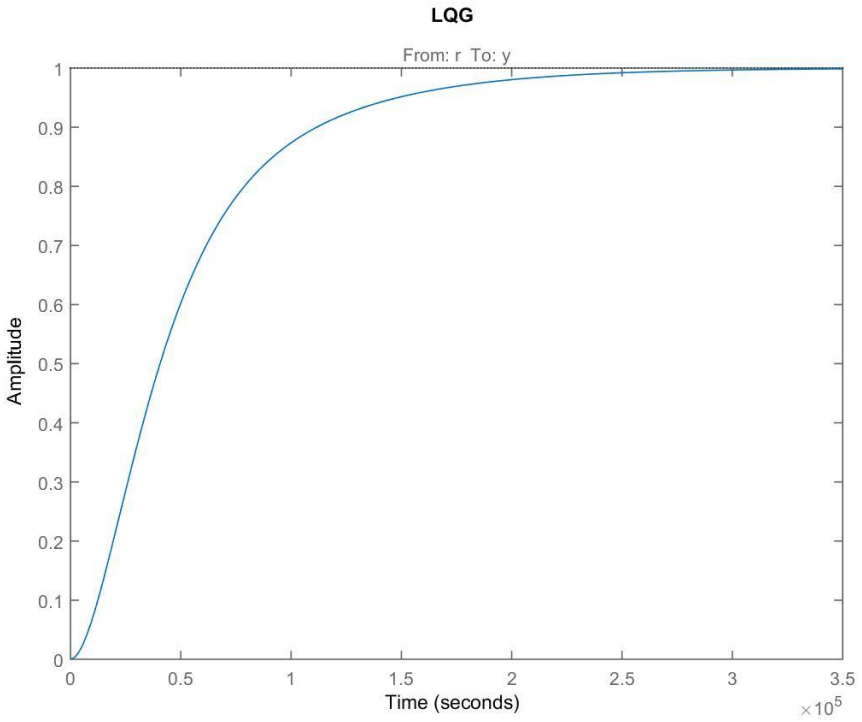


Рис. 4.32. – Перехідна характеристика системи з збалансованим LQG регулятором

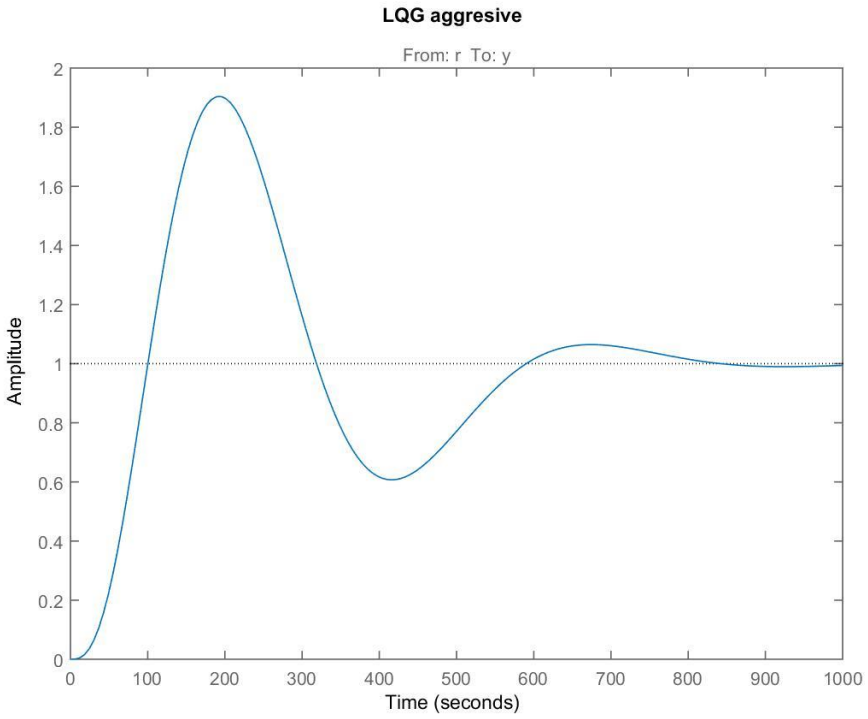


Рис. 4.33. – Перехідна характеристика агрессивного LQG регулятора

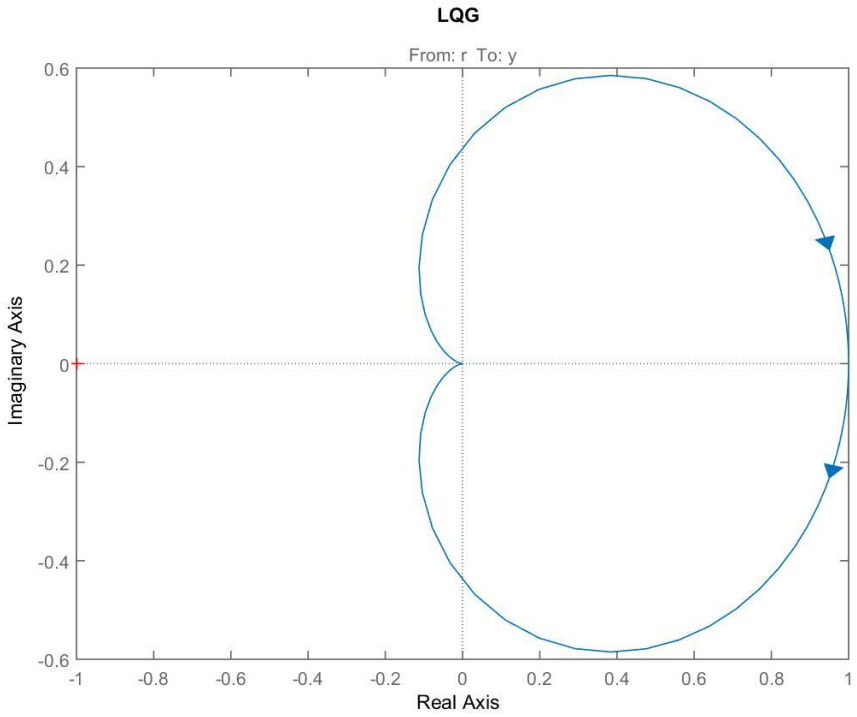


Рис. 4.34. – Годограф Найквіста збалансованого LQG регулятора

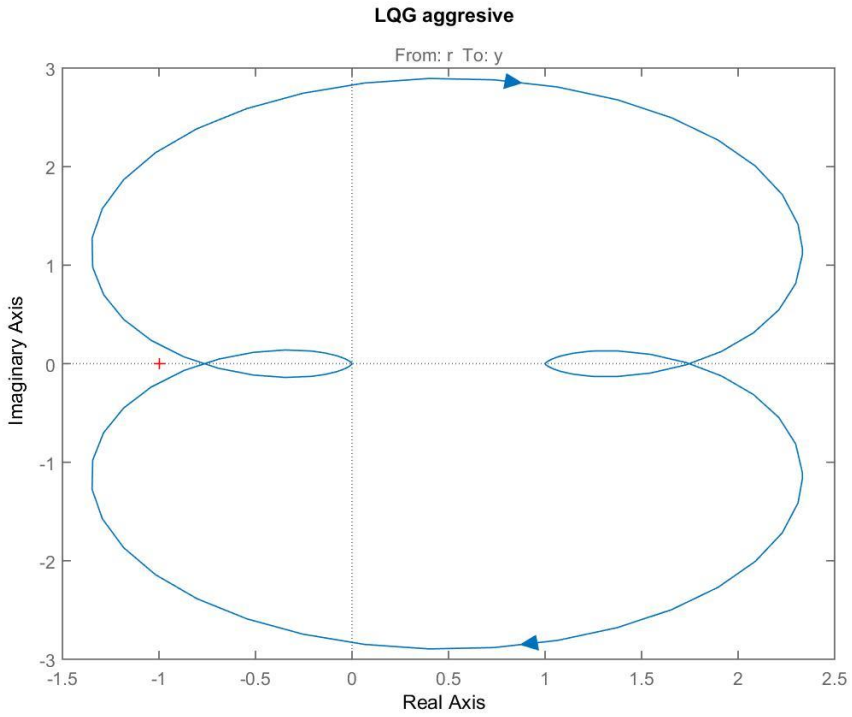


Рис. 4.35. – Годограф Найквіста агрессивного LQG регулятора

Для порівняння отриманих результатів виведемо перехідні характеристики на один графік. Оскільки данний регулятор не є ПІД регулятором, ми не можемо сформувати таблицю, котру формували в п.4

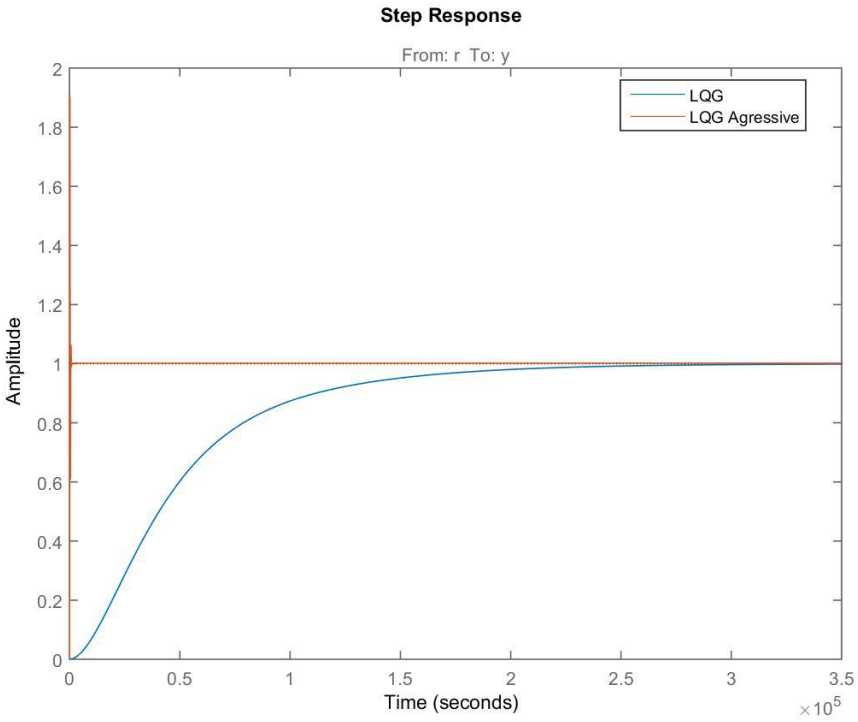


Рис. 4.36. – Перехідні характеристики LQG регуляторів

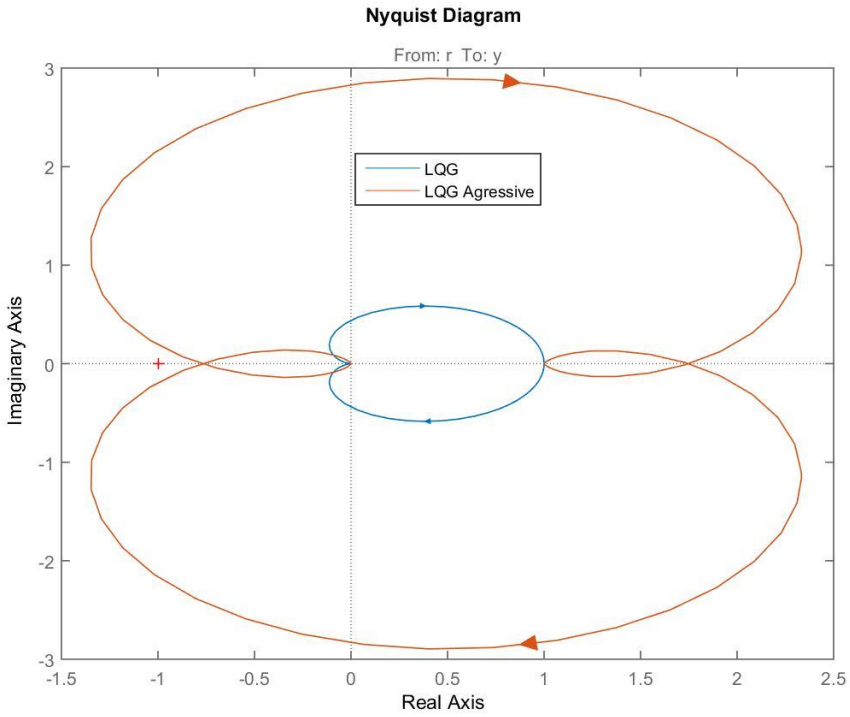


Рис. 4.37. – Годограф Найквіста для двух LQG регуляторів

З отриманих даних можемо зробити висновок: хоча й агресивний регулятор має вражаючий час встановлення, але коливність синтезованої системи надто велика, тому найкращим рішенням буде використовувати збалансований LQG регулятор.

5. Розробка MES системи

Міжнародна асоціація виробників і користувачів систем управління виробництвом (MESA International) визначила в 1994 році модель MESA-11, а в 2004 році модель c-MES, які доповнюють моделі і стандарти управління виробництвом і виробничою діяльністю, що сформувалися за останні десятиліття:

Стандарт ISA-95, «Інтеграція систем управління підприємством і технологічним процесом» («Enterprise-Control System Integration»), який визначає єдиний інтерфейс взаємодії рівнів управління виробництвом і компанією і робочі процеси виробничої діяльності окремого підприємства.

Стандарт ISA-88, «Управління періодичним виробництвом» («Batch Control»), який визначає технології управління періодичним виробництвом, ієрархію рецептур, виробничі дані.

Спільнота Відкритих Додатків (Open Applications Group, OAG): некомерційне промислова спільнота, що має своєю метою просування концепції функціональної сумісності між бізнес-додатками і розробку стандартів бізнес-мов для досягнення зазначеної мети.

Модель процесів ланцюжка поставок (Supply-Chain Operations Reference, SCOR): референтна модель для управління процесами ланцюжка поставок, що зв'язує діяльність постачальника і замовника. Модель SCOR описує бізнес-процеси для всіх фаз виконання вимог замовника. Розділ SCOR «Виготовлення» («Make») присвячений, в основному, виробництва.

5.1 Функкції MES системи

Серед основних завдань MES виділяються: Активація виробничих потужностей на основі детального пооперационного планування виробництва. Відстеження виробничих потужностей.Збір інформації, пов'язаної з виробництвом, від:

- систем автоматизації виробничого процесу;
- датчиків – обладнання;

- персоналу;
- програмних систем.

Відстеження і контроль параметрів якості. Забезпечення персоналу і устаткування інформацією, необхідною для початку процесу виробництва. Встановлення зв'язків між персоналом і обладнанням в рамках виробництва. Встановлення зв'язків між виробництвом і постачальниками, споживачами, інженерним відділом, відділом продажів і менеджментом.

Реагування на:

- Вимоги по номенклатурі виробництва;
- Зміна компонентів, сировини і напівфабрикатів, які застосовує в процесі виробництва;
- Зміна специфікації продуктів;
- Доступність персоналу і виробничих потужностей.

Гарантування відповідності застосовним юридичним актам, наприклад нормам Food and Drug Administration (FDA) США.

Відповідність перерахованим вище індустріальним стандартам.

Основні функції MES:

RAS (англ. Resource Allocation and Status) - контроль стану і розподіл ресурсів. Управління ресурсами: технологічним обладнанням, матеріалами, персоналом, навчанням персоналу, а також іншими об'єктами, такими як документи, які повинні бути в наявності для початку виробничої діяльності. Забезпечує детальну історію ресурсів і гарантує, що обладнання відповідним чином підготовлено для роботи. Контролює стан ресурсів в реальному часі. Управління ресурсами включає резервування і диспетчеризацію, з метою досягнення цілей оперативного планування. Для виробництва хлорбензолу: управління технологічним обладнанням виробництва хлорбензолу.

ODS (англ. Operations / Detail Scheduling) - оперативне детальне планування. Забезпечує впорядкування виробничих завдань, засноване на черговості, атрибутах, характеристиках і рецептах, пов'язаних зі специфікою виробів таких як: форма, колір,

послідовність операцій і ін. І технологією виробництва. Мета - скласти виробниче розклад з мінімальними переналаштування обладнання та паралельною роботою виробничих потужностей для зменшення часу отримання готового продукту і часу простою. Для виробництва хлорбензолу – характеристики та технологія виробництва хлорбензолу.

DPU (англ. Dispatching Production Units) - диспетчеризація виробництва. Управляє потоком одиниць продукції у вигляді завдань, замовлень, серій, партій і замовлення-нарядів. Диспетчерська інформація може надаватися в тій послідовності, в якій робота повинна бути виконана, і змінюється в реальному часі в міру виникнення подій на цеховому рівні. Це дає можливість зміни заданого календарного плану на рівні виробничих цехів. Включає функції усунення браку і переробки відходів, поряд з можливістю контролю трудовитрат в кожній точці процесу з буферизацією даних. Для виробництва хлорбензолу: диспетчер задач.

DOC (англ. Document Control) - Управління документами. Контролює зміст і проходження документів, які повинні супроводжувати випускається виріб, включаючи інструкції і нормативи робіт, способи виконання, креслення, процедури стандартних операцій, програми обробки деталей, записи партій продукції, повідомлення про технічні зміни, передачу інформації від зміни до зміни, а також забезпечує можливість вести планову та звітну цехову документацію. Також включає інструкції з безпеки, контроль захисту навколишнього середовища, державні та необхідні міжнародні стандарти. Зберігає історію проходження і зміни документів. Для виробництва хлорбензолу: вся нормативна база.

DCA (англ. Data Collection / Acquisition) - збір і зберігання даних. Взаємодія інформаційних підсистем з метою отримання, накопичення і передачі технологічних і керуючих даних, циркулюючих в виробничому середовищі підприємства. Функція забезпечує інтерфейс для отримання даних та параметрів технологічних операцій, які використовуються в формах і документах, що прикріплюються до одиниці продукції. Дані можуть бути отримані з цехового рівня як вручну, так і автоматично від обладнання, в необхідному масштабі часу. Для виробницвтва хлорбензолу: збір та зберігання даних в одне сховище.

LM (англ. Labor Management) - управління персоналом. Забезпечує отримання інформації про стан персоналу та управління ним в необхідному масштабі часу. Включає звітність за присутністю і робочого часу, відстеження сертифікації, можливість відстеження невиробничої діяльності, такої, як підготовка матеріалів або інструментальні роботи, в якості основи для обліку витрат за видами діяльності (activity based costing, ABC). Можлива взаємодія з функцією розподілу ресурсів, для формування оптимальних завдань. Для виробництва хлорбензолу – база даних всіх робочих та відстежує їх активність.

QM (англ. Quality Management) - управління якістю. Забезпечує аналіз в реальному часі вимірюваних показників, отриманих від виробництва, для гарантовано правильного управління якістю продукції і визначення проблем, що вимагають втручання обслуговуючого персоналу. Ця функція формує рекомендації щодо усунення проблем, визначає причини браку шляхом аналізу взаємозв'язку симптомів, дій персоналу і результатів цих дій. Може також відстежувати виконання процедур статистичного управління процесом і статистичного управління якістю продукції (SPC / SQC), а також керувати виконанням лабораторних досліджень параметрів продукції. Для цього до складу MES додаються лабораторні інформаційно-керуючі системи (LIMS). Для виробництва хлорбензолу: звязка датчики – аналіз.

PM (англ. Process Management) - управління виробничими процесами. Відстежує виробничий процес і або коригує автоматично, або забезпечує підтримку прийняття рішень оператором для виконання коригувальних дій і вдосконалення виробничої діяльності. Ця діяльність може бути як внутріопераціонной і спрямованої виключно на відслідковують і керовані машини й устаткування, так і межоперационной, що відстежує хід процесу від однієї операції до іншої. Вона може включати управління тривогами для забезпечення гарантованого повідомлення персоналу про зміни в процесі, що виходять за прийнятні межі стійкості. Для виробництва хлорбензолу – автоматичне керування технологічним процесом.

MM (англ. Maintenance Management) - управління технічним обслуговуванням і ремонтом. Відстежує і управляє обслуговуванням устаткування й інструментів.

Забезпечує їх працездатність. Забезпечує планування періодичного і попереджувального ремонтів, ремонту станом. Накопичує і зберігає історію подій, що відбулися (відмови, зменшення продуктивності і ін.) Для використання в діагностуванні виникли і попередження можливих проблем.

PTG (англ. Product Tracking and Genealogy) - відстеження і генеалогія продукції. Забезпечує можливість отримання інформації про стан і місцезнаходження замовлення в кожен момент часу. Інформація про стан може включати дані про те, хто виконує завдання, компонентах, матеріалах і їх постачальників, номері лота, серійний номер, поточних умовах виробництва, а також будь-які тривоги, дані про повторній обробці і інші події, пов'язані з продукту. Функція відстеження в реальному часі створює також архівну запис. Ця запис забезпечує відстеження компонентів і їх використання в кожному кінцевому продукті. Для виробницва хлорбензолу – система, яка дозволяє відстежити кожну партію продукції та її стан.

PA (англ. Performance Analysis) - аналіз продуктивності. Забезпечує формування звітів про фактичні результати виробничої діяльності, порівняння їх з історичними даними та очікуваним комерційним результатом. Результати виробничої діяльності включають такі показники, як коефіцієнт використання ресурсів, доступність ресурсів, час циклу для одиниці продукції, відповідність плану і відповідність стандартам функціонування. Може включати статистичний контроль якості процесів і продукції (SPC / SQC). Систематизує інформацію, отриману від різних функцій, що вимірюють виробничі параметри. Ці результати можуть бути підготовлені в формі звіту або представлені в реальному часі в вигляді поточної оцінки експлуатаційних показників. Для виробництва хлорбензолу – база звітів та порівняння ключових показників якості.

Ми будемо реалізовувати: аналіз продуктивності, збір та зберігання даних, , оперативне детальне планування. Вони повязана наступним чином: спочатку ми збираємо і зберігаємо всі можливі дані, після чого ми проводимо їх аналіз та візуалізацію.

Для більшого розуміння побудуємо таблицю:

Таблиця 5.1 Функції MES системи

№ п/п	Функція	Реалізація
1	Збір та зберігання даних	Збір даних з API та завантаження їх в БД
2	Аналіз продуктивності	Будуємо аналіз на основі історичних подій з БД та візуалізуємо
3	Оперативне детальне планування	Будуємо аналіз на основі даних з БД і тримаємо дані в реальному часі

Для початку нам потрібно визначити джерела даних, після чого організувати збір інформації. Після забору інформації нам потрібно зберігати дані, а потім проводити візуалізацію.

5.2 Джерела даних

В якості джерела даних будемо використовувати RESTful API.

Дані повинні передаватися у вигляді невеликої кількості стандартних форматів (наприклад, HTML, XML, JSON). Будь-який REST протокол (HTTP в тому числі) повинен підтримувати кешування, не повинен залежати від мережевого прошарку, не повинен зберігати інформації про стан між парами «запит-відповідь». Стверджується, що такий підхід забезпечує масштабовність системи і дозволяє їй еволюціонувати з новими вимогами.

В цілому, описані нижче підходи підходять для будь-яких джерел даних, які віддають відповіді JSON обєктами.

5.3 Інтеграція з API

Для інтеграції з API будемо використовувати скрипт, який “збирає” всі JSON файли та з них будує SQL базу даних. Лістинг основної програми повністю наведений в додатку 1:

Візуально, ми будемо впроваджувати наступну схему:

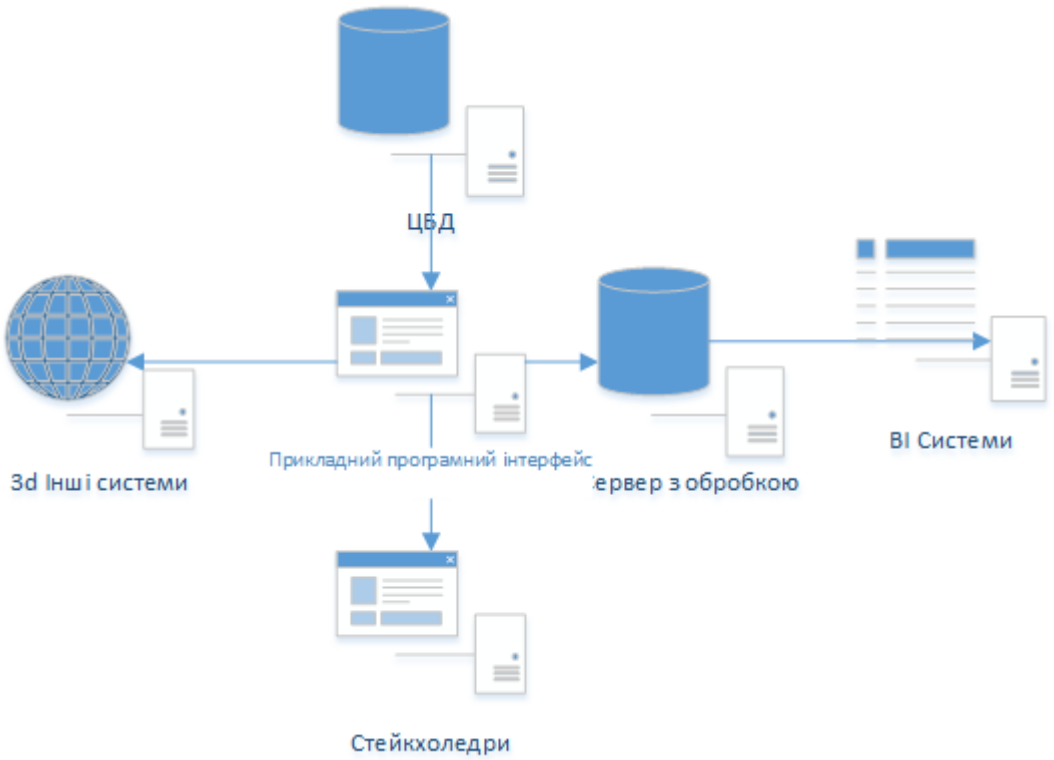


Рис. 5.1. – Схема системи

Де ЦБД – центральна база даних. 3d інші системи – будь які інші системи, які використовують дані для аналізу. Сервер з обробкою – основний сервер, на якому проводяться всі операції над даними. BI системи - системи, котрі віддають дані кінцевому користувачу в легкому для розумінні форматі (графіки, таблиці, тощо). Стейкхолдери – аукціонери, або стейкхолдери в широкому сенсі цього слова.

На рівні баз данних це виглядатиме наступним чином:

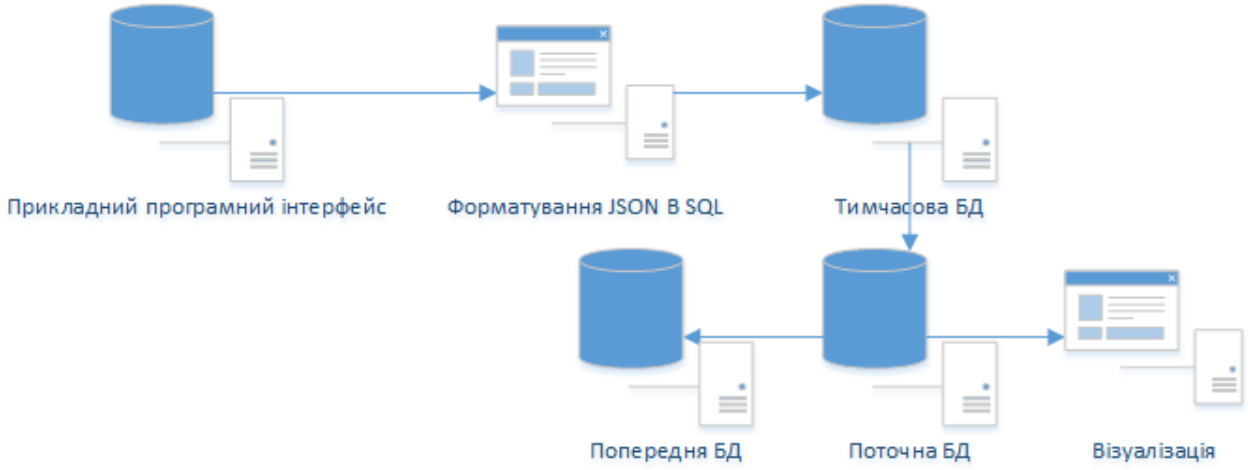


Рис. 5.2. – Схема на рівні баз даних

Прикладний програмний інтерфейс – протокол, через який ми отримуємо дані. Зазвичай в форматі JSON. Форматування JSON в SQL – для того, щоб дані були придатні для обробки, особливо швидкої нам необхідно перевести дані в табличну структуру, а саме – в SQL, після чого ці дані ми завантажуємо в тимчасову БД, допоки всі JSON обєкти не будуть перенесені в тимчасову базу. Після цього тимчасова база стає поточною, а поточна – попередньою. Візуалізація даних будується на основі поточної БД.

Лістинг Python скрипту, для отримання даних:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import sys
import time
import zlib
import socket
import logging
import logging.config
import simplejson as json
import peewee

from argparse import ArgumentParser
from ConfigParser import RawConfigParser

from munch import munchify
from iso8601 import parse_date
from restkit.errors import ResourceError
from openprocurement_client.client import TendersClient
# fix for py2exe
from socketpool import backend_thread
backend_thread.__dict__


logger = logging.getLogger('transfuse')

CHAR_MAX_LENGTH = 250
LONGCHAR_MAX_LENGTH = 1000


class MyConfigParser(RawConfigParser):
        def optionxform(self, optionstr):
        return optionstr

        def test(self, filename, prefix='table:'):
```

```
        """Check config file for model duplicates"""
        parser = MyConfigParser()
        if not parser.read(filename):
        raise ValueError("Can't read config from file %s" % filename)
        for section in parser.sections():
        if section.startswith(prefix) and section in self.sections():
        raise ValueError("Model %s duplicate in file %s" % (section, filename))
        return True


class MyApiClient(TendersClient):
        def __init__(self, key, config):
        params = {'limit': 1000, 'mode': ''}
        if config['mode'] in ('test', '_all_'):
        params['mode'] = config['mode']
        if config['timeout']:
        socket.setdefaulttimeout(float(config['timeout']))
        TendersClient.__init__(self, key, config['host_url'], config['api_version'], params)
        if config.get('resource'):
        self.prefix_path = '/api/{}/{}'.format(config['api_version'], config['resource'])
        self.allow_preload = config.get('preload', None)
        self.api_version = config['api_version']

        def request_cookie(self):
        self.head('/api/{}/spore'.format(self.api_version))

        def preload_tenders(self, feed='', callback=None):
        preload_items = []
        items = True
        if not self.headers.get('Cookie', None):
        self.request_cookie()
        while items:
        items = self.get_tenders(feed=feed)
        if items:
        preload_items.extend(items)
        if not self.allow_preload:
        break
        if items and callback:
        callback(len(preload_items), items[-1])
        return preload_items

        def get_tender(self, tender_id):
        for i in range(5):
        try:
        if not self.headers.get('Cookie', None):
                self.request_cookie()
        return TendersClient.get_tender(self, tender_id)
```

```
            except (socket.error, ResourceError) as e:
            logger.error("get_tender %s reason %s", tender_id, str(e))
            if i > 1:
                        self.headers.pop('Cookie', None)
                        #self.params.pop('offset', None)
            time.sleep(10 * i + 10)
            raise ResourceError("Maximum retry reached")


class BaseTendersModel(peewee.Model):
            class Meta:
            pass

            @classmethod
            def model_name(klass):
            return klass._meta.db_table


class CacheTendersModel(BaseTendersModel):
            tender_id = peewee.CharField(primary_key=True)
            dateModified = peewee.CharField()
            gzip_data = peewee.BlobField()


class TendersToSQL(object):
            client_config = {
            'key': "",
            'host_url': "https://public.api.openprocurement.org",
            'api_version': "0",
            'mode': "_all_",
            'timeout': 30,
            'offset': None,
            'limit': None,
            'resume': False,
            'preload': False,
            }
            server_config = {
            'class': 'MySQLDatabase',
            }
            server_defaults = {
            'MySQLDatabase': {
            'host': 'localhost',
            'user': 'prozorro',
            'passwd': 'prozorro',
            'db': 'prozorro',
            },
            'PostgresqlDatabase': {
```

```
'host': 'localhost',
'user': 'prozorro',
'password': 'prozorro',
'database': 'prozorro',
},
'SqliteDatabase': {
'db': 'prozorro.db',
}
}
table_schema = {
}
field_types = {
'char': (peewee.CharField, {'null': True, 'max_length': CHAR_MAX_LENGTH}),
'longchar': (peewee.CharField, {'null': True, 'max_length': LONGCHAR_MAX_LENGTH}),
'text': (peewee.TextField, {'null': True}),
```

5.4 Збереження даних та візуальна обробка

Для збереження даних та підтримання їх в актуальному вигляді використовуємо bash скрипти.

Лістинг run.sh

```
#!/bin/sh
cd /home/d.usmanov/trans || exit 1
if [ ! -s $1/$2.ini ] ; then
  echo "Usage: $0 config_dir (dir|dir2)"
  exit 1
fi
. env/bin/activate
LOG=log/${1}_${2}.log
while true
do
```

```
  # clear prev logs first
  test -f $LOG.2 && rm -f $LOG.2
  test -f $LOG.1 && mv -f $LOG.1 $LOG.2
  test -f $LOG   && mv -f $LOG $LOG.1
  # start current iteration
  date >>$LOG
  # check config before start
  if [ -f $1/$2.ini ] ; then
        ./trans.py --debug -i ${1}/base.ini ${1}/${2}.ini ${1}/${2}_*.ini >>$LOG 2>&1
        if [ $? -eq 0 ] ; then
        ./tabrename.sh $1 $2 >>$LOG 2>&1
        date >>$LOG
        else
        echo `date` Trans fail, dont rename tables >>$LOG
        fi
  else
        echo `date` File not found $1/$2.ini >>$LOG
  fi
  sleep 600
done

Start.all.sh:
#!/bin/sh
if [ -n $1 ] ; then
  sleep $1
fi
if [ `ps ux | grep -c run.sh` -gt 1 ] ; then
  echo "Already started"
  exit 1
fi
```

```
if [ `ps ux | grep -c trans.py` -gt 1 ] ; then
  echo "Already started"
  exit 1
fi
cd /home/d.usmanov/trans || exit 1
```

Rename_tables.sh:

```
#!/bin/sh
MYHOST=XXX
MYUSER=XXX
MYPASS=XXX
do_mysql() {
  mysql -h $MYHOST -u $MYUSER --password=$MYPASS $1
}

show_tables() {
  echo show tables | do_mysql $1
}
drop_sql() {
  show_tables $1 | awk '{ if ($_ !~ "Tables_in_") print "DROP TABLE "$_";" }'
}
drop_tables() {
  echo "USE $1"
  drop_sql $1 | grep $2
  drop_sql $1 | grep $2 | do_mysql $1
}
rename_sql() {
  show_tables $1 | awk '{ if ($1 !~ "Tables_in_") print "RENAME TABLE '$1'."$_" TO
'$2'."$_";" }'
}
rename_tables() {
```

```
  echo "USE $1"
  rename_sql $1 $2 | grep $3 | grep -v cache
  rename_sql $1 $2 | grep $3 | grep -v cache | do_mysql $1
}
run_script() {
  DB=${1}_curr
  FN=${1}/${2}.sql
  if [ -f $FN ] ; then
        echo "USE $DB RUN $FN"
        cat $FN | do_mysql $DB
  fi
*}
if [ "x$1" = "x" ] ; then
  echo Usage: $0 database table_prefix
  exit
fi

echo "Connect to host=$MYHOST user=$MYUSER password=*** db=$1 prefix=$2"
drop_tables ${1}_prev $2
rename_tables ${1}_curr ${1}_prev $2
#drop_tables ${1}_curr $2
rename_tables ${1}_temp ${1}_curr $2
run_script $1 $2
```

Візуалізацію зроблена за допомогою open source рішення, а саме superset.

Приклад для оперативного планування та аналізу продуктивності зображено нижче.

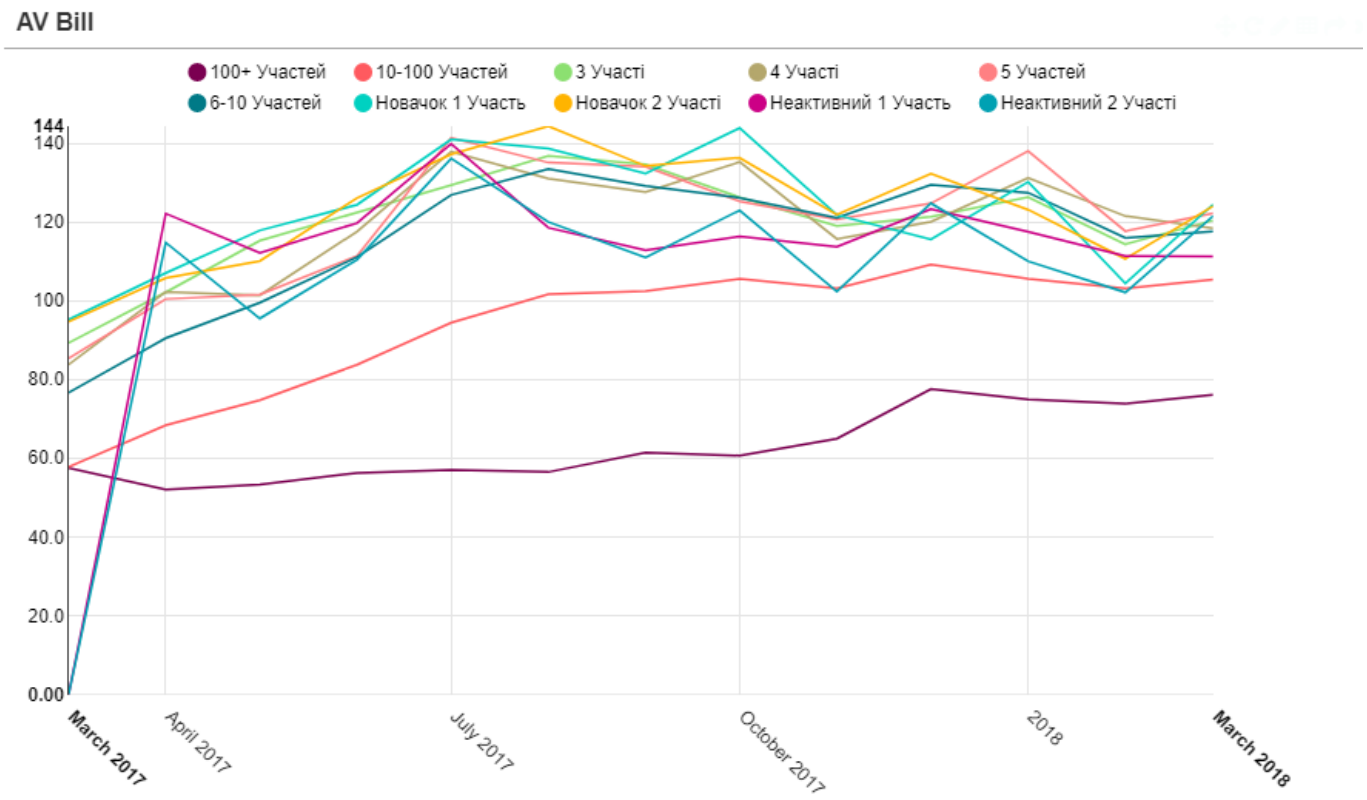


Рис 5.3 – Планування

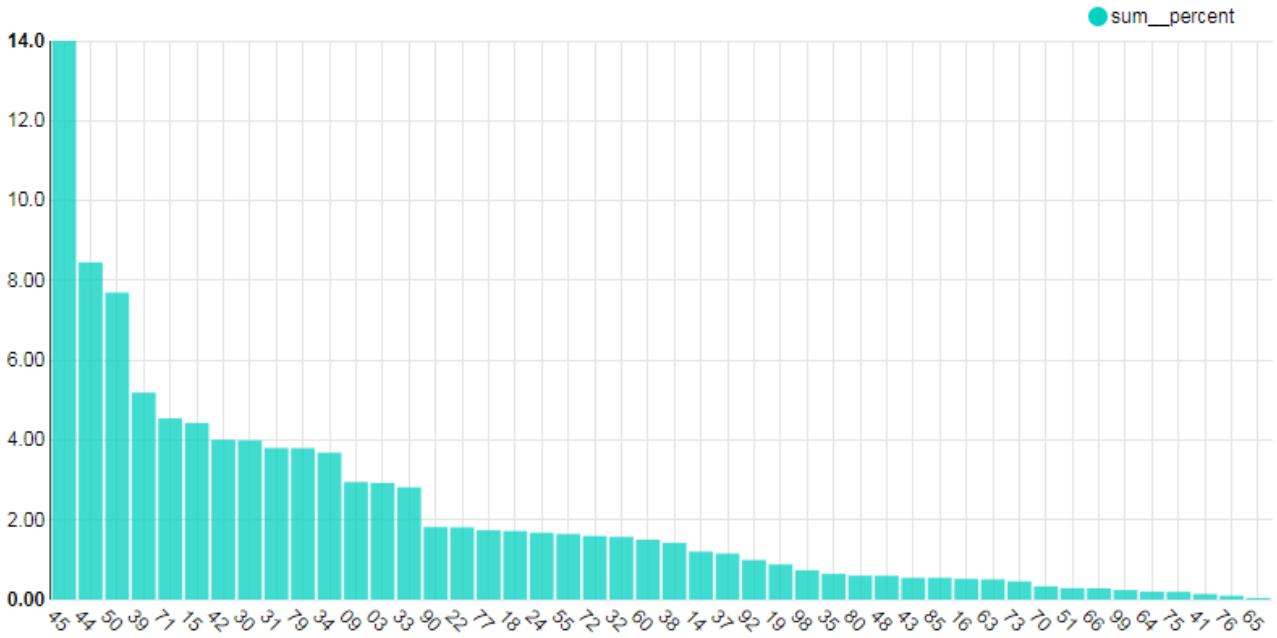


Рис 5.4 – Аналіз продуктивності робітників

6. Розробка стартап проекту

6.1 Вступ до розробки стартап проекту

Стартап - це тільки що створена компанія (можливо навіть не є юридичною особою), яка знаходиться на стадії розвитку і будує свій бізнес на ос-нові нових інноваційних ідей, або на основі технологій, які нещодавно з'явилися.

Однією з основних причин створення, успішного розвитку та подальшого існування стартапів вважають неповороткість і повільність великих ко-рпорацій, які успішно використовують уже наявні продукти, а розробкою і створенням нових майже не займаються. Тому стартапи, завдяки своїй мобі-льності в плані втілення нових ідей складають конкуренцію великим корпораціями.

Основним ресурсом для створення нового стартапу служить хороша новаторська ідея. Власне за свіжими і незвичайними ідеями женеться більшість і часто, купуючи їх, не шкодують великі суми грошей. Сама ідея, що не має ніякого матеріального втілення, а існує тільки на папері, або "на словах" (план стартапу), може коштувати дуже багато. Іншим фактором успішності цієї ідеї є її затребуваність (ступінь необхідності для споживача), адже ідея може бути незвичайною і новою, але користі від неї буде мінімум.

Що стосується України, існує дуже багато різних компаній, які надають свої послуги у вигляді навчання і натхнення, мотивації до успіху підприємців, молоді та дітей. Це дуже зручно і вигідно, тому що людині допомагають реалізувати свій бізнес професіонали. Але будь-який бізнес вимагає зазвичай вкладення чималих коштів, які в нашій країні, знайти дуже важко, враховуючи те, що банківська система в Україні надає жорсткі умови кредитування, а знайти інвесторів і зацікавити в нашій небагатій країні складно, а також існують ризики і тиск з боку конкурентів і влади.

6.2 Опис ідеї стартап проекту

Автоматизація технологічних процесів є ключовою ланкою у загальній системі функціонування будь-якого хімічного виробництва. Сучасна автоматизація – це не лише персональні комп’ютери, контролери, промислові мережі, а і, звичайно ж, програмне забезпечення. Основною проблемою на більшості виробництв України і світу, які навіть оснащені найдорожчими і найточнішими пристроями є складні процеси та проблема їх аналізу та представлення для швидкого та якісного прийняття рішень.

Основною ідеєю стартап проекту є створення онлайн стрімінгової системи аналітики(OLAP) для підтриємства, в частковому випадку – для виробництва хлорбензолу та повязання цих даних з відкритими даними, які знаходяться на ринку.

Можливість застосування таких систем буде у будь-якого хімічного виробництва (і не лише).

На сьогоднішній день відкритих рішень такого типу немає – існують лише закриті корпораційні рішення, які коштують дуже дорого. Ті що відкриті – не мають спрямування на підприємства.

Реалізація такої системи розглядається для вирбництва безперервного хлорування бензолу, оскільки комерційними даними підприємства ми не володіємо буде розглянуто лише на прикладі відкритих даних, які можуть нас цікавити. Проте інтегрувати дане рішення з комерційними даними – достатньо просто.

Таблиця 6.1 – Опис ідеї проету

Зміст ідеї	Напрямки застосування	Вигода для користувача
Сервіс по онлайн аналітиці в реальному часі базуючись на відкритих даних	Бюджетування	Економія коштів підриємства
	Планування	Розширення ринку збуту
	Аналіз ринку	

6.3 Визначення сильних, слабких та нейтральних характеристик ідеї проекту

Визначимо перелік слабких (W), сильних (S) та нейтральних характеристик (N) та властивостей ідеї проекту для є формування його конкурентоспроможності. Для цього використаємо стандарту таблицю для типового WSN аналізу.

Таблиця 6.2 – Визначення сильних, слабких та нейтральних сторін стартап проекту

№ п/ п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	Конкурент 1	Конкурент 2	Конку-рент 3			
1.	Застосування систем в будь-якій сфері виробництва							+
2.	Отримання реальних даних з «полів»							+
3.	Застосування для оперативного планування бюджету						+	
4.	Застосування для оперативного планування продаж							+

6.4 Технологічний аудит проекту

Для того, щоб зрозуміти, як реалізувати ідею технічно нам необхідно провести технологічний аудит проекту за стандартними техніками – для початку цього буде достатньо. Основною ідеєю є розробка сервіса онлайн аналітики, а тому, будемо шукати схожі технології, які дозволять нам вирішити дану задачу.

Таблиця 6.3 – Технічний аудит проекту

№ п/п	**Ідея проекту**	**Технології реалізації**	**Наявність технології**	**Доступність технології**
1	Сервіс по онлайн аналітиці в реальному часі базуючись на відкритих даних	Використання BI модулю superset: (apache/superset)	Так, необхідне допрацювання	Так
2	Сервіс по онлайн аналітиці в реальному часі базуючись на відкритих даних	Використання QlikView/Sense	Так	Частково
3	Сервіс по онлайн аналітиці в реальному	Використання платформи Tableu	Так	Ні

	часі базуючись на відкритих даних			

Виходячи з проаналізованих даних було обрано технологічне рішення на якому буде базуватись наше рішення: apache superset: оскільки воно є безкоштовним і достатньо гнучим для доробок. Також воно є найдоступнішим

6.5 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Визначаємо потенційні групи клієнтів, їх характеристики та формуємо орієнтовний перелік вимог до рекуперацийних систем, як товару для кожної групи (табл. 6.4).

Таблиця 6.4. – Визначення сильних, слабких та нейтральних характеристик проекту

№ п/п	**Потреба, що формує ринок**	**Цільова аудиторія (цільові сегменти ринку)**	**Відмінності у поведінці різних потенційних цільових груп**	**Вимоги споживачів до товару**

1.	Базова потреба, яку задовольняє товар (згідно концепції потенційного товару)	Визначити потенційні цільові групи клієнтів, що можуть бути зацікавлені у задоволені означеної потреби	Вписати фактори, що формують поведінку клієнта (стандарти, технічні регламенти, інші фактори цінового та нецінового характеру) та особливості купівлі та експлуатації	- до продукції - до компанії-постачальника
2.	Необхідність швидкого реагування на зміну ринку	Виробництва, підприємці	Фінансовий план	Отримання даних в реальному часі.
3.	Необіхність ефективного планування бюджету	Виробництва, підприємці	Фінансовий план, ISA-95	Отримання даних для прийняття рішень
4.	Створення централізованої системи для аналітики	Виробництва, підприємці	ISA-95, ДСТУ	Швидкий доступ до всіх даних підтриємства

Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища з використання аналітичних систем. Тому складаємо таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (табл. 6.5).

Таблиця 6.5 – Фактори загроз

№ п/п	**Фактор**	**Зміст загрози**	**Можлива реакція компанії**

1.	Спад виробництва в галузях господарства	Нема потреби у продукті	
2.	Економічне відкриття кордонів	Прихід нових технологій	Використання нових технологій
3.	Зростання інфляції	Падіння платоспроможності в тому числі і підприємств з якими співпрацюємо	Гнучке ціноутворення, планування

Таблиця 6.6 – Фактори можливостей

№ п/п	**Фактор**	**Зміст можливості**	**Можлива реакція компанії**
1.	Політика протекціонізму	Захист вітчизняного виробника внаслідок чого буде розвиватися галузь споживача стартапу.	Залучення у коло споживачів споріднених галузей
2.	Стимулювання розвитку інноваційного підприємництва	Зменшення податкового тиску на стартап.	Масштабування стартапу

Проводимо аналіз (табл. 6.7.) пропозиції: визначаємо загальні риси конкуренції на ринку з використання рекуперацийних систем.

Таблиця 6.7 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	**В чому проявляється дана характеристика**	**Вплив на діяльність підприємства (можливі дії компанії, щоб бути**
Тип конкуренції	Монополістична конкуренція	Конкуренція, ринкова ситуація, за якої відносно велика кількість невеликих виробників пропонують схожу але не ідентичну продукцію.
Рівень конкурентної боротьби	Національний рівень	Національна конкуренція і конкурентоспроможність окремих фірм, підприємств і організацій відбувається і проявляється на національному
За галузевою ознакою	Міжгалузева	Застосування аналітичних систем можливе на будь-яких підтриєствах
Конкуренція за видами товарів	Товарно-видова конкуренція	Відстеження тенденцій на ринку з можливістю появи на ринку продуктів-замінників.
За характером конкурентних переваг	Цінова	Гнучке ціноутворення з урахуванням динаміки попиту. Удосконалення технології, що спрямована на підвищення базових переваг.
За інтенсивністю - марочна/не марочна	Немарочна	Забезпечення масштабованості стартапу в найближчій перспективі для створення стійкого сприйняття стартапу як окремої бізнес одиниці.

Проводимо оцінку привабливості стратегічної зони господарювання (СЗГ) стартапу із застосуванням методу Дельфі. На першому етапі була про-ведена оцінка зміни в прогнозованому збільшенні стратегічної зони господа-рювання (G) методом Дельфі (табл. 6.8.).

Таблиця 6.8 – Аналіз зміни в прогнозованому збільшенні СЗГ (G)

Параметри	***Шкала інтенсивності***										
	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5
1. Темп збільшення галузі											
2. Приріст числа споживачів											
3. Динаміка розширення ринку											
4. Ступінь оновлення продукції											
5. Ступінь оновлення технології											
6. Рівень насичення попиту											
7. Суспільне сприйняття товару											
8. Державне регулювання збільшення											
9. Збільшення числа конкурентів											
10. Ступінь застарівання продукції											
Загальна оцінка змін	+10										

На другому етапі проведено оцінку прогнозованих тенденцій зміни рентабельності СЗГ (Р) (табл. 6.9)

Таблиця 6.9 – Аналіз оцінки змін рентабельності СЗГ (Р)

Параметри	***Шкала інтенсивності***										
	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5
1. Темп збільшення галузі											
2. Приріст числа споживачів											
3. Динаміка розширення ринку											
4. Ступінь оновлення продукції											
5. Ступінь оновлення технології											
6. Рівень насичення попиту											
7. Суспільне сприйняття товару											
8.Державне регулювання збільшення											
9. Збільшення числа конкурентів											
10. Ступінь застарівання продукції											
Загальна оцінка змін рентабельності на перспективу	+17										

На третьому етапі визначимо рівень впливу загроз (-Qi) і використання можливостей (+Qi) зовнішнього середовища для підприємства в даній СЗГ. Оцінювання кожного чинника відбувається за шкалою від -5 до +5 балів. Зважена оцінка визначається як результат множення оцінки чинни-ка, значущості групи факторів та значущості чинника в групі, до якої даний чинник належить. Негативна зважена оцінка розглядається як потенційна за-гроза, а позитивна - як можливість для організації в даній СЗГ.

Таблиця 6.10– Оцінка зовнішніх погроз і можливостей для підприємства

Група чинників	Чинники	Значущість групи чинників	Значущість чинника в групі	Оцінка чинника	Зважена оцінка	
					загрози	можливості
1	2	3	4	5	6	7
1. Економічні	Темпи інфляції	2,8	0,14	4		1,7
	Стабільність курсу Гривни		0,39	3		2,55
	Тарифи на транспорт і енергоресурси		0,21	-3	-1,675	
	Податкові ставки		0,23	-2	-1	
2. Політичні	Державне регулювання ринків	0,8	0,2	4		0,48
	Інвестиційна політика		0,32	3		0,56
	Політична Стабільність		0,19	-3	-0,38	
	Законодавча база для регулювання підприємницької Діяльності		0,27	0	0	
3. Ринкові	Інтенсивність Конкуренції	1,6	0,14	2		0,59
	Ціни на чинники виробництва		0,26	1		0,38
	Укладення прямих договорів		0,45	-4	-2,45	
	Динаміка попиту і пропозиції		0,16	3		0,64
4. Чинники конкуренції	Агресивність конкуренції	1,9	0,38	4		2,1
	Тенденції зміни числа конкурентів		0,49	-3	-2,035	
	Переваги лідерів галузі		0,22	3		0,8
5. Виробничо-технологічні	Стан виробництва	2,8	0,23	-2	-1,15	
	Можливість нових технологічних		0,36	1		0,8
	Рівень технології конкурентів		0,19	3		1,376
	Доступність ресурсів		0,32	-2	-1,53	
6. Соціальні	Соціальна напруженість в галузі	1,5	0,35	2		1,05
			0,27	4		1,62
	Підвищення кваліфікації кадрів		0,38	-2	-1,14	

	Юридичні обмеження в бізнесі					
Підсумкова оцінка		11,4	-	-	-11,36	14,646

На підставі отриманих результатів розраховано значення привабливості СЗГ за формулою:

Привабливість СЗГ = αG + βP + γ (O - T),

де α = 0.4, β=0.3, γ=0.3;

G - зміни в прогнозованому збільшенні СЗГ;

P - зміни в рентабельності СЗГ;

+Qi та -Qi - відповідно оцінки можливостей та загроз для організації.

Привабливість СЗГ = 0,4 · 10 + 0,3 · 17 + 0,3 · (14,646 – 11,36) = 10,0858.

На четвертому етапі оцінюємо (табл. 6.11) рівень привабливості за відповідною шкалою.

Таблиця 6.11 – Шкала оцінювання привабливості СЗГ

Значення	**Характеристика**
-50…-46	Дуже непривабливa
-45…-36	Високо непривабливa
-35…-26	Достатньо непривабливa
-25…-16	Помірно непривабливa
-15…-6	Майже непривабливa
-5…+5	Відсутність привабливості
+6…+15	Майже привабливa
+16…+25	Помірно привабливa
+26…+35	Достатньо привабливa
+36…+45	Високо привабливa
+46…+50	Дуже привабливa

Значення привабливості СЗГ (11.5035) потрапляє в інтервал від 6 до 15, що може вважатися майже привабливим для підприємства. Позитивно слід оцінювати спрямованість показників G та P, які свідчать про непогані перспективи діяльності підприємства в даній СЗГ.

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних та слабких сторін, загроз та можливостей (табл. 6.5.9).

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення

Значення привабливості СЗГ (11.5035) потрапляє в інтервал від 6 до 15, що може вважатися майже привабливим для підприємства. Позитивно слід оцінювати спрямованість показників G та P, які свідчать про непогані перспективи діяльності підприємства в даній СЗГ.

Фінальним етапом ринкового аналізу можливостей впровадження про-екту є складання SWOT-аналізу (матриці аналізу сильних та слабких сторін, загроз та можливостей (табл. 6.12).

Перелік ринкових загроз та ринкових можливостей складається на осно-ві аналізу факторів загроз та факторів можливостей маркетингового середо-вища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення.

Таблиця 6.12 – SWOT- аналіз стартап-проекту

Сильні сторони: • контроль за здійсненням витрат, пошук можливостей щодо їхнього зниження; • інвестиційна привабливість підприємства; • зважена цінова політика; • врахування потреб споживачів.	*Слабкі сторони:* • частка ринку; • результативність рекламної політики; • організація системи комунікацій.
Можливості: • зростання грошових доходів населення; • застосування сучасних технологій організації товароруху; • впровадження різних форм організації	*Загрози:* • недосконаліть та змінюваність законодавства; • інфляційні процеси; • високий рівень безробіття.

торгівлі.	

На основі SWOT-аналізу розробляються альтернативи ринкової пове-дінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтов-ний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок.

Визначені альтернативи аналізуються (табл. 6.13) з точки зору строків та ймовірності отримання ресурсів.

Таблиця 6.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової Поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Стратегія нейтралізації ринкових загроз сильними сторонами стартапу.	Висока	1 рік
2.	Стратегія підсилення сильних сторін за рахунок ринкових можливостей.	Висока	6 місяців
3.	Стратегія компенсації слабких сторін наявними ринковими можливостями.	Середня	1 рік
4.	Стратегія виходу з ринку	Низька	не має

Обрано стратегію нейтралізації ринкових загроз сильними сторонами стартапу

6.6 Розроблення ринкової стратегії

Розроблення ринкової стратегії (табл. 6.14) передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів.

Таблиця 6.14 – Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи	Готовність споживача прийняти продукт	Орієнтовний попит	Інтесивність конкуренції	Простота входу у сегмент
1	Виробництво(продавець)	Готові	Високий	Середня	Низька
2	Виробництво(покупець)	Готові	Високий	Висока	Низька
3	Моніторингові групи	Готові	Середній	Висока	Висока

Проаналізувавши потенційні групи споживачів обирано наступні три цільові групи та визначано стратегію охоплення ринку. Оскільки компанія працює із кількома сегментами, розробляючи для них окремо програми ринкового впливу, то використовуємо стратегію диференційованого маркетингу.

Для роботи в обраних сегментах ринку сформулюємо базову стратегію розвитку, а саме стратегію диференціації

Таблиця 6.15 – Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	**Стратегія охоплення ринку**	**Ключові конкурентоспроможні позиції відповідно до обраної альтернативи**
Стратегія диференціації	Ринкове позиціонування	• по відношенню до прямих конкурентів диференціація знижує ступінь замінності товару, посилює прихильність марці, зменшує чутливість до ціни і тим самим підвищує рентабельність; • прихильність клієнтів послабляє їх тиск на фірму і перешкоджає приходу на ринок нових конкурентів; • підвищена рентабельність збільшує стійкість до можливого зростання витрат в результаті дій сильного постачальника; • відмітні властивості товару і завойована прихильність клієнтів захищають фірму і від товарів-замінників.

Стратегія диференціації передбачає надання товару важливих з точки зору споживача відмітних властивостей, які роблять товар відмінним від то-варів конкурентів. Така відмінність може базуватися на об'єктивних або суб'єктивних, відчутних і невідчутних властивостях товару (у ширшому ро-зумінні – комплексі маркетингу), бути реальною або уявною. Інструментом реалізації стратегії диференціації є ринкове позиціонування.

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки розробляємо страте-гію позиціонування (табл. 6.16), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Таблиця 6.16 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	**Базова стратегія розвитку**	**Ключові конкурентоспроможні позиції власного стартап- проекту**	**Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)**
Комп'ютерно-інформаційна технологія повинна мати можливість: • графічного представлення даних; • зображення всіх необхідних компонентів (технологічний апарат та регулятор); • оцінки роботи отриманої системи; • поточне виведення всіх значень параметрів.	Стратегія диференціації	До конкурентних переваг слід віднести: • оптимальна ціна за необхідний програмний пакет; • швидкодія роботи; • багаторівнева система безпеки; • більш мобільна система управління в порівнянні з конкурентами.	• Доступний продукт в даному сегменті; • Використання інформаційних технологій для імітаційного моделювання контуру; регулювання температури; • Візуальна оцінка роботи системи з регулятором.

6.7 Розроблення маркетингової програми стартап проекту

Першим кроком є формування маркетингової концепції до товару, який отримає споживач. Для цього у таблиці 6.17 підсумовуємо результати попереднього аналізу конкурентоспроможності товару.

Таблиця 6.17 – Визначення ключових переваг концепції потенційного товару

Потреба	**Вигода, яку пропонує Товар**	**Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити**
Необхідність швидкого реагування на зміну ринку	Аналітика в реальному часі	Можливість аналітики в реальному часі
	Всі дані в одному місці	Всі дані зберігаються в одній БД
	Оцінка роботи підприємства	Можливість оцінити якість підприємства
	Оцінка планування	Можливість гнучкого планування

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій (табл. 6.18), що спирається на попередньо обра-ну основу для позиціонування, визначену специфіку поведінки клієнтів.

Таблиця 6.18 – Концепція маркетингових комунікацій

№ п/п	**Специфіка поведінки цільових клієнтів**	**Канали комунікацій, якими користуються цільові клієнти**	**Ключові позиції, обрані для позиціонування**	**Завдання рекламного повідомлення**
1.	Обережний вибір потенційних контрагентів, що зумовлено особливістю ринку з використання Інформаційних технологій	Інтернет-розсилки	Технологія	Привернути увагу до систем з аналітикою в реальному часі
2.		Контекстна реклама	Технологія	
3.		Спеціалізовані виставки, форуми	Технологія	

6.8 Реалізація рішення

Рішення було реалізовано за допомогою фреймворку superset. Виглядає воно наступним чином. Для цього була проведена процедура отримання даних з прикладного програмного інтерфейсу та інтегрування її в три існуючі бази даних. Приклади отриманих результатів наведені нижче.

Аналіз середнього чеку по сегментам постачальників підприємства наведений на Рис 6.8.1

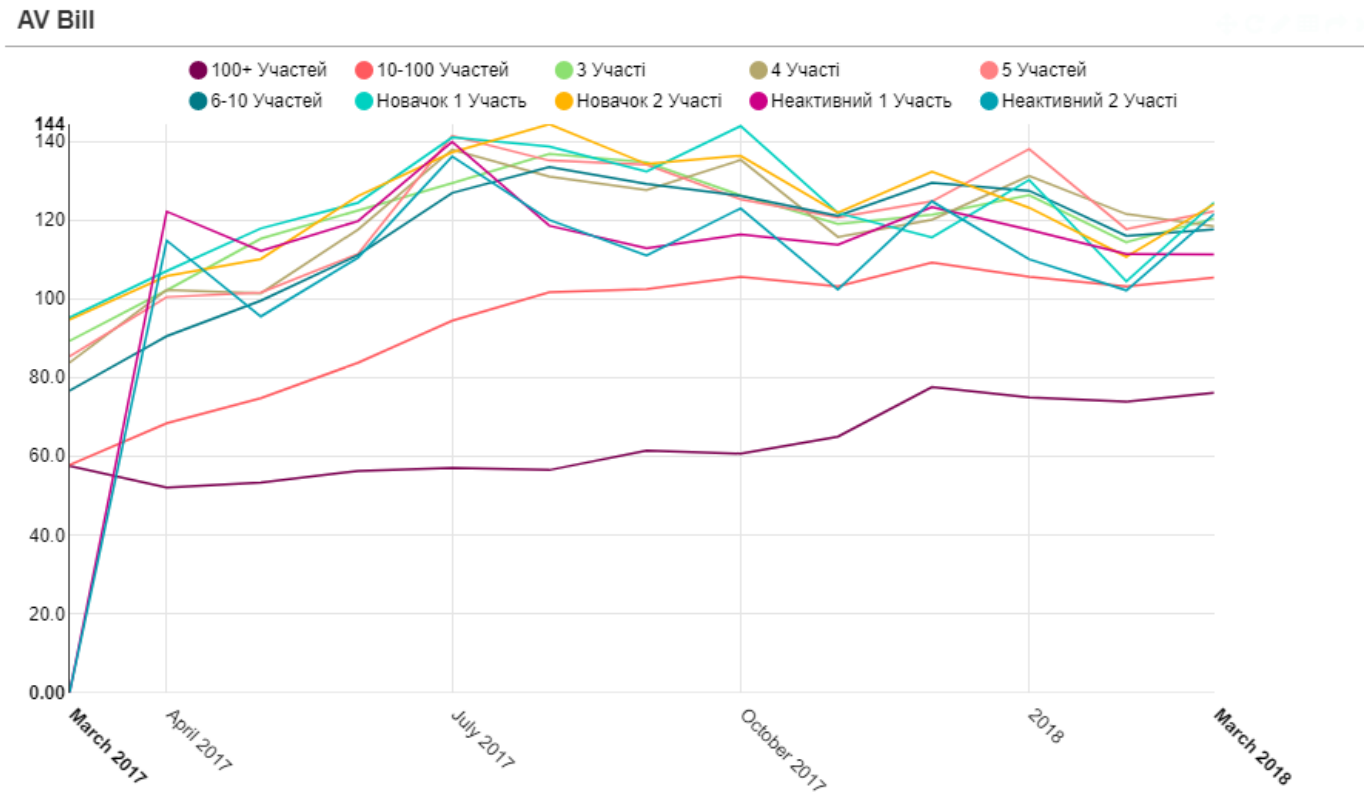


Рис 6.8.1 – Аналіз середнього чеку по сегментам постачальників підприємства

Динаміка клієнтської бази

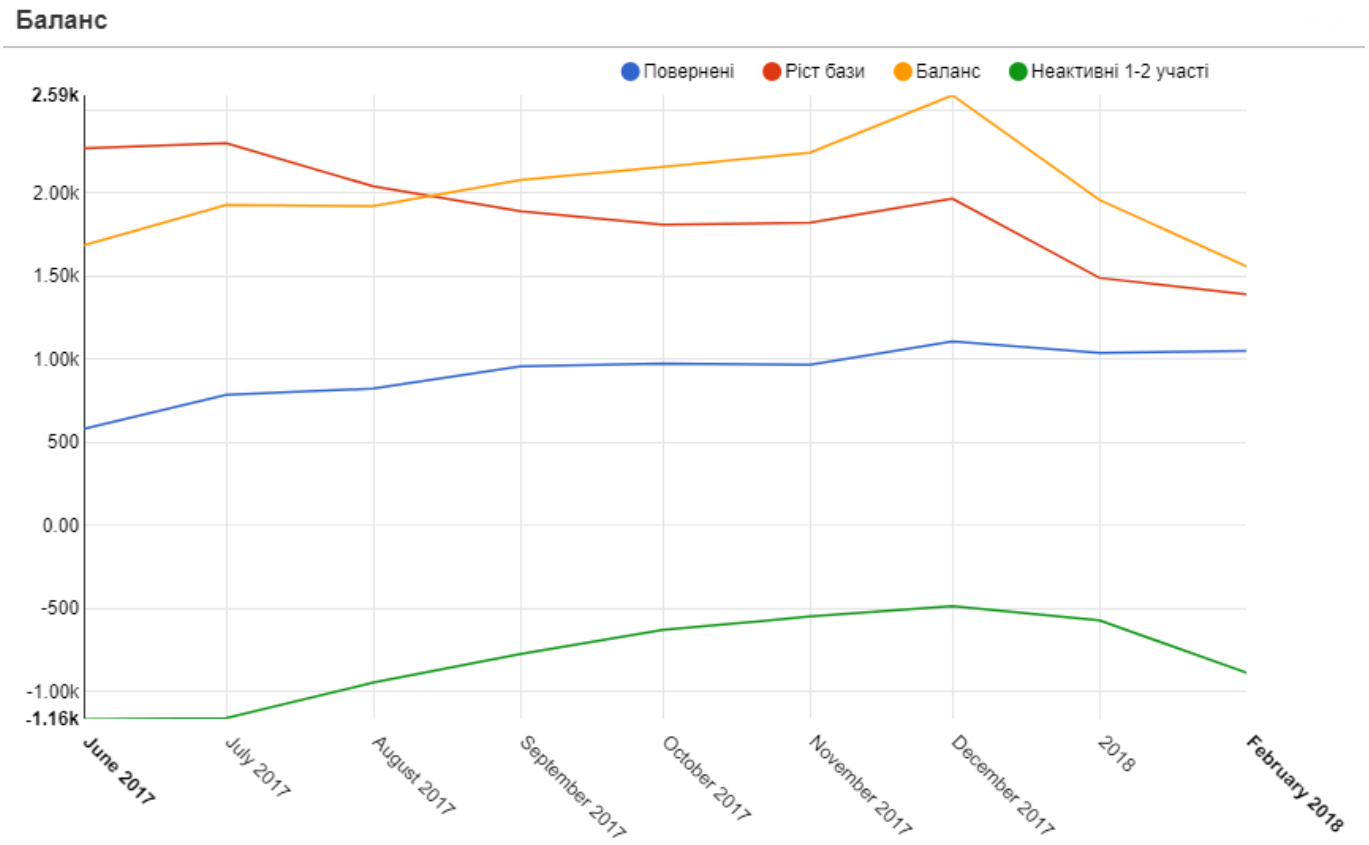


Рис 6.8.2 – Динаміка клієнтської бази

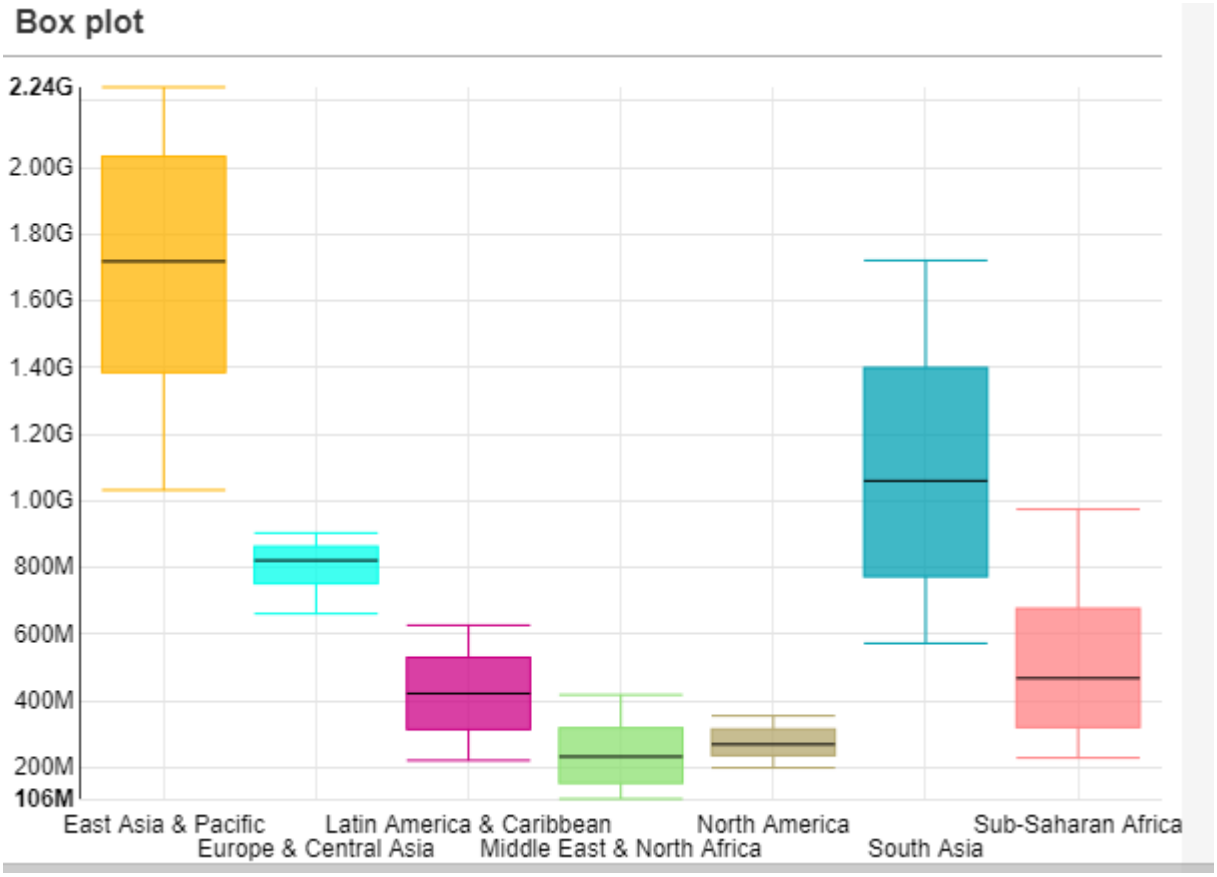


Рис 6.8.3 – Розподіл постачальників в грн

6.9 Висновки

Переглянувши всі позитивні і негативні риси аналітичних систем для технологічних процесів, можем зробити висновок, що представлена ідея є актуальною та цілком адекватною для застосування. Технологія є доступною в плані ціни та застосування на власних підприємствах чи устано-вах.

Оскільки ринок монополістична конкуренція, то бар'єри входу на ринок – високі. Наявна конкуренція високої концентрації середнього рівня інтенсивності. Для ринкової реалізації проекту обрано стратегію диференційованого маркетингу, яка передбачає чітку ідентифікацію окремих сегментів, з якими планує взаємодіяти стартап, зокрема це галузі хімічного виробництва, та й в цілому будь-яке підприєствот. Доцільна подальша імплементація проекту.

Розвиток запропонованого стартап-проекту буде пов'язаний з можливістю інтенсифікації економічного розвитку в Україні, бо життєздатність стартапу

визначатиметься темпами зростання хімічної промисловості та споріднених галузей, де необхідний моніторинг та управління процесами в апаратах.

Висновки

При виконанні магістерської дисертації розглянуто автоматизацію виробництва процесу отримання хлорбензолу безперервним хлоруванням бензолу.

На сучасному етапі розвитку виробництва автоматизація даного процесу, як і будь-яких інших складних процесів, дозволяє отримати високі показники продуктивності установки і якості кінцевого продукту. Автоматизація зокрема передбачає контроль, керування, сигналізацію та блокування технологічних параметрів за допомогою відповідних автоматичних пристроїв без безпосередньої участі людини, але і під її контролем.

Даний проект є надійним, з точки зору капіталовкладень, так як оснащення цеху сучасним технологічним обладнанням та сучасними технічними засобами автоматизації веде до економії енергоресурсів і призводить до швидкої окупності капіталовкладень.

В дипломному проекті була розроблена математична модель процесу абсорбції, здійснена розробка системи керування за ПІ та ПІД-законом регулювання та досліджені питання стосовно модернізації технологічного обладнання. Було розроблено багаторівневу систему управлінням процесом безперервного хлорування бензолу, налаштовано отримання та інтеграцію даних. Було розроблено веб сервіс для проведення онлайн аналізу з будь-якої точки світу. В розділі стартап проекту було розроблено стартап проект та наведені приклади рішення, також обгрунтовано необхідність продукту на ринці.

При иконанні дипломного проекту та оформлення проектної документації застосовано програмні середовища MS Office, Visio, MATLAB, Python, SQL.

Література

1. Лукінюк М.В. Технологічні вимірювання та прилади : Навч. посіб. для. курс. проектування. К.: "ПОЛІПАРНАС" , 2002. - 257с: іл.
2. Бабіченко А.К., Тушинський В.І., Михайлов В.С. Промислові засоби автоматизації. Ч. 1. Вимірювальні пристрої / За заг. ред. Бабіченка А.К.: Навч. посібник. - Харків: НТУ "ХПГ, 2001 р. - 470 с.
3. А.И. Емельянов, О.В. Капник «Проектирование автоматизированных систем управления технологическими процессами»,Москва «Энергия», 1974г.
4. Клюев А. С. «Проектирование систем автоматизации технологических процессов». Справочное пособие - М.: Машиностроение, 1980. - с. 214.
5. Дытнерский Ю.И. Процессы и аппараты химической технологии. – Москва «Химия», 1995. – 260с.
6. Основные процессы и аппараты химической технологии: Касаткин А.Г. - Москва, 1988. - 832с;
7. Жученко А.И., Кубрак Н.А., Голинко И.М. Динамика объектов с распределенными параметрами: Учебное пособие. – К.: ЕКСМО, 2005.- 121с.
8. Жученко А. І., Кваско М.З., Кубрак Н. А. Ідентифікація динамічних характеристик. Комп’ютерні методи. К.:ВІПОЛ, 2000. – 182с., іл.
9. Симановский А. Ю. Методика настройки регуляторов. Инструкция. – К.: МИКРОЛ, 2004. – 64с.
10. Бояринов Л.И., Кафаров В.В. Методы оптимизации в химической технологии. – М., Химия, 1975.
11. Реклейтис Г., Рейвиндран А., Рэгсдел К. Оптимизация в технике: В 2-х кн. – М.: Мир, 1986 – 349 с.
12. Гил Ф., Мюррей У., Райт М. Практическая оптимизация – М.: Мир, 1985. – 509с.
13. Ладіева Л.Р. Оптимальне керування системами. Навчальний посібник. - К.: НМЦ ВО, 2000.-187с.

14. Майника Э. Алгоритмы оптимизации на сетях и графах: Пер. с англ. – М.: Мир, 1981. – 323с.

15. Гроп. Методи идентификации систем. М.: Мир. 1979. - 302с.

16. Эйкофф П. Основы идентификации систем управления. –М., Мир, 1975.

17. Таха Х.А. Введение в исследование операций – М.: Издательский дом «Вильямс», 2005 – 912с.

18. Зайченко Ю.П. Дослідження операцій: підручник.– К.: Вид. дім "Слово", 2006 – 816с.

19. Хант Э. Искусственный интеллект. – М.: Мир, 1978.- 558с.

20. Сигеру Омату, Нейроуправление и его приложения / Сигеру Омату, Марзуки Халид, Рубия Юсоф – М.: ИПРЖР, 2000. – 272 с. – ISBN: 5-93108-006-6.

21. Хайкин Саймон Нейронные сети: полный курс, 2е издание / Пер. с англ. – М.: Издательский дом "Вильямс", 2006. 1104 с.

22. Вентцель Е.С. Теория вероятностей. – М.: Высш. шк., 2001.–575 с.

23. Гмурман В.Е. Теория вероятностей и математическая статистика. –М.: Высш. шк., 2000. – 479 с.

24. Айвазян С. А. Прикладная статистика: Основы моделирования и первичная обработка данных. Справочное изд. / С. А. Айвазян, И. С. Енюков, Л. Д. Мешалкин. – М.: Финансы и статистика, 1983. – 471с.

25. Барсегян А.А., Куприянов М.С., Степаненко В.В Методы и модели анализа данных: OLAP и Data Mining. – СПб.: БХВ-Петербург, 2004.–336с.

26. Усманов Д.О., Ковалюк Д.О. Багаторівнева система керування виробництвом безперервного хлорування бензолу – Автоматизація та комп’ютерно-інтегровані технології [Текст]: Матеріали V Міжнародної науково-практичної конференції молодих учених, аспірантів і студентів (АКІТ-2018). - Київ, 11-12 квітня 2018 р. – К.: КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2018. – С. 131-132.

Додатки

Додаток 1

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import sys
import time
import zlib
import socket
import logging
import logging.config
import simplejson as json
import peewee

from argparse import ArgumentParser
from ConfigParser import RawConfigParser

from munch import munchify
from iso8601 import parse_date
from restkit.errors import ResourceError
from openprocurement_client.client import TendersClient
# fix for py2exe
from socketpool import backend_thread
backend_thread.__dict__


logger = logging.getLogger('transfuse')

CHAR_MAX_LENGTH = 250
LONGCHAR_MAX_LENGTH = 1000


class MyConfigParser(RawConfigParser):
    def optionxform(self, optionstr):
        return optionstr

    def test(self, filename, prefix='table:'):
        """Check config file for model duplicates"""
        parser = MyConfigParser()
        if not parser.read(filename):
            raise ValueError("Can't read config from file %s" % filename)
        for section in parser.sections():
            if section.startswith(prefix) and section in self.sections():
                raise ValueError("Model %s duplicate in file %s" % (section, filename))
        return True


class MyApiClient(TendersClient):
    def __init__(self, key, config):
        params = {'limit': 1000, 'mode': ''}
        if config['mode'] in ('test', '_all_'):
            params['mode'] = config['mode']
        if config['timeout']:
            socket.setdefaulttimeout(float(config['timeout']))
        TendersClient.__init__(self, key, config['host_url'], config['api_version'], params)
        if config.get('resource'):
            self.prefix_path = '/api/{}/{}'.format(config['api_version'], config['resource'])
        self.allow_preload = config.get('preload', None)
        self.api_version = config['api_version']
```

```
    def request_cookie(self):
        self.head('/api/{}/spore'.format(self.api_version))

    def preload_tenders(self, feed='', callback=None):
        preload_items = []
        items = True
        if not self.headers.get('Cookie', None):
            self.request_cookie()
        while items:
            items = self.get_tenders(feed=feed)
            if items:
                preload_items.extend(items)
            if not self.allow_preload:
                break
            if items and callback:
                callback(len(preload_items), items[-1])
        return preload_items

    def get_tender(self, tender_id):
        for i in range(5):
            try:
                if not self.headers.get('Cookie', None):
                    self.request_cookie()
                return TendersClient.get_tender(self, tender_id)
            except (socket.error, ResourceError) as e:
                logger.error("get_tender %s reason %s", tender_id, str(e))
                if i > 1:
                    self.headers.pop('Cookie', None)
                    #self.params.pop('offset', None)
                time.sleep(10 * i + 10)
        raise ResourceError("Maximum retry reached")


class BaseTendersModel(peewee.Model):
    class Meta:
        pass

    @classmethod
    def model_name(klass):
        return klass._meta.db_table


class CacheTendersModel(BaseTendersModel):
    tender_id = peewee.CharField(primary_key=True)
    dateModified = peewee.CharField()
    gzip_data = peewee.BlobField()


class TendersToSQL(object):
    client_config = {
        'key': "",
        'host_url': "https://public.api.openprocurement.org",
        'api_version': "0",
        'mode': "_all_",
        'timeout': 30,
        'offset': None,
        'limit': None,
        'resume': False,
        'preload': False,
    }
    server_config = {
        'class': 'MySQLDatabase',
```

```
}
server_defaults = {
    'MySQLDatabase': {
        'host': 'localhost',
        'user': 'prozorro',
        'passwd': 'prozorro',
        'db': 'prozorro',
    },
    'PostgresqlDatabase': {
        'host': 'localhost',
        'user': 'prozorro',
        'password': 'prozorro',
        'database': 'prozorro',
    },
    'SqliteDatabase': {
        'db': 'prozorro.db',
    }
}
table_schema = {
}
field_types = {
    'char': (peewee.CharField, {'null': True, 'max_length': CHAR_MAX_LENGTH}),
    'longchar': (peewee.CharField, {'null': True, 'max_length': LONGCHAR_MAX_LENGTH}),
    'text': (peewee.TextField, {'null': True}),
    'date': (peewee.DateTimeField, {'null': True}),
    'int': (peewee.IntegerField, {'null': True}),
    'bigint': (peewee.BigIntegerField, {'null': True}),
    'float': (peewee.FloatField, {'null': True}),
    'decimal': (peewee.DecimalField, {'null': True, 'max_digits': 16, 'decimal_places': 2}),
    'bool': (peewee.BooleanField, {'null': True})
}
allowed_fieldopts = ['null', 'index', 'unique', 'primary_key']

def __init__(self, config, args):
    db_class = config.get('server', 'class')
    if db_class in self.server_defaults:
        self.server_config.update(self.server_defaults[db_class])
    self.server_config.update(config.items('server'))
    self.client_config.update(config.items('client'))
    # update config from args
    self.update_config(args)
    # create client
    api_key = self.client_config.pop('key')
    logger.info("Create client %s", self.client_config)
    self.client = MyApiClient(api_key, self.client_config)
    # log connection config w/o password
    safe_config = dict(self.server_config)
    safe_config.pop('passwd', None)
    safe_config.pop('password', None)
    logger.info("Connect server %s", safe_config)
    # create database connection
    db_class = peewee.__dict__.get(self.server_config.pop('class'))
    self.db_init = self.server_config.pop('init', '').strip(' \"')
    self.db_name = self.server_config.pop('db', None)
    if not self.db_name and self.server_config.get('database'):
        self.db_name = self.server_config.pop('database')
    self.database = db_class(self.db_name, **self.server_config)
    if self.db_init:
        self.database.execute_sql(self.db_init)
    # create model class
    self.create_models(config)
    # create cache model
    self.init_cache(config)
```

```
def update_config(self, args):
    for key in ('offset', 'limit', 'resume'):
        if getattr(args, key, None):
            self.client_config[key] = getattr(args, key)
    for key in ('no_cache', 'drop_cache', 'fill_cache'):
        setattr(self, key, getattr(args, key, False))
    self.ignore_errors = args.ignore

def init_cache(self, config):
    self.cache_model = None
    if self.no_cache:
        return
    if not config.has_option('cache', 'table'):
        return
    cache_table = config.get('cache', 'table')
    if not cache_table:
        return
    logger.info("Init cache table `%s`", cache_table)
    self.cache_model = CacheTendersModel
    self.cache_model._meta.database = self.database
    self.cache_model._meta.db_table = cache_table
    self.cache_max_size = 0xfff0
    self.cache_hit_count = 0
    self.cache_miss_count = 0
    try:
        self.cache_model.select().count()
        cache_table_exists = True
    except:
        cache_table_exists = False
        self.database.rollback()
    if self.drop_cache and cache_table_exists:
        logger.warning("Drop cache table `%s`", cache_table)
        self.cache_model.drop_table()
        cache_table_exists = False
    if not cache_table_exists:
        logger.info("Create cache table `%s`", cache_table)
        self.cache_model.create_table()

@staticmethod
def field_name(name):
    return name.replace('.', '_').replace('(', '_').replace(')', '').strip()

def create_table(self, model_class):
    logger.warning("Drop & Create table `%s`", model_class._meta.db_table)
    with self.database.transaction():
        try:
            model_class.select().count()
            model_class.drop_table()
        except:
            self.database.rollback()
    with self.database.transaction():
        model_class.create_table()

def init_model(self, table_name, table_schema):
    logger.info("Create model %s", table_name)
    if table_name in self.models:
        raise IndexError('Model %s already exists', table_name)

    fields = dict()
    parsed_schema = list()
    table_options = {'__name__': table_name}
    has_primary_key = False
```

```
    for key, val in table_schema:
        if key.startswith('__'):
            if key == '__iter__':
                table_options['__path__'] = val
                val = val.split('.')
            table_options[key] = val
            continue
        name = self.field_name(key)
        logger.debug("+ %s %s", name, val)
        opts = [s.strip() for s in val.split(',')]
        # [table:model_name]
        # field = type,flags,max_length
        if opts[0] not in self.field_types:
            raise TypeError("Unknown type '%s' for field '%s'" % (opts[0], key))
        fieldtype, fieldopts = self.field_types.get(opts[0])
        if len(opts) > 1:
            if opts[1] not in self.allowed_fieldopts:
                raise ValueError("Unknown option '%s' for field '%s'" % (opts[1], key))
            fieldopts = dict(fieldopts)
            fieldopts[ opts[1] ] = True
            if opts[1] == 'primary_key':
                has_primary_key = True
        if len(opts) > 2:
            fieldopts['max_length'] = int(opts[2])
        fields[name] = fieldtype(**fieldopts)
        # parse field path
        funcs = key.replace(')', '').split('(')
        chain = funcs.pop().split('.')
        parsed_schema.append((name, chain, funcs, opts[0]))

    if not has_primary_key:
        fields['pk_id'] = peewee.PrimaryKeyField(primary_key=True)
    class_name = "%sModel" % table_name.title()
    model_class = type(class_name, (BaseTendersModel,), fields)
    model_class._meta.table_options = table_options
    model_class._meta.table_schema = parsed_schema
    model_class._meta.database = self.database
    model_class._meta.db_table = table_name
    self.models[table_name] = model_class
    if not self.client_config.get('resume', False):
        self.create_table(model_class)

def create_models(self, config):
    self.models = dict()
    for section in config.sections():
        if section.startswith('table:'):
            table_schema = config.items(section)
            table, name = section.split(':', 2)
            self.init_model(name, table_schema)

def apply_func(self, fn, data):
    if fn == 'count':
        return len(data)
    if fn == 'sum':
        return sum(data)
    if fn == 'min':
        return min(data)
    if fn == 'max':
        return max(data)
    if fn == 'avg':
        return sum(data) / len(data)
    raise ValueError("Unknown function %s" % fn)
```

```
def field_value(self, chain, funcs, data):
    for key in chain:
        if isinstance(data, list):
            res = list()
            for item in data:
                val = item.get(key)
                # make val to be always a list
                if not isinstance(val, list):
                    val = [val]
                # and solve the problem only for list
                for vi in val:
                    if isinstance(vi, dict):
                        vi['parent'] = item
                    if vi is not None:
                        res.append(vi)
            data = res
        elif data:
            data = data.get(key)
        if data is None:
            return
    for fn in funcs:
        data = self.apply_func(fn, data)
    return data

@staticmethod
def parse_iso_datetime(value):
    return parse_date(value).replace(tzinfo=None)

def process_model_item(self, model_class, data):
    table_schema = model_class._meta.table_schema
    fields = dict()
    for field_info in table_schema:
        name, chain, funcs, ftype = field_info
        try:
            value = self.field_value(chain, funcs, data)
            if isinstance(value, list):
                value = value[0] if len(value) else None
            if value is None:
                continue
            if ftype in ['char', 'longchar', 'text']:
                value = unicode(value)
            if ftype == 'char' and len(value) > CHAR_MAX_LENGTH:
                raise ValueError("Value too long, use longchar or text")
            if ftype == 'longchar' and len(value) > LONGCHAR_MAX_LENGTH:
                value = value[:LONGCHAR_MAX_LENGTH]
            if ftype == 'date':
                value = self.parse_iso_datetime(value)
            fields[name] = value
        except Exception as e:
            message = "%s on model [%s] field %s itemID:%s" % (str(e),
                str(model_class.model_name()), name, data.get('id'))
            raise type(e), type(e)(message), sys.exc_info()[2]
    item = model_class(**fields)
    item.save(force_insert=True)

def process_model_data(self, model_class, data):
    table_options = model_class._meta.table_options
    if table_options.get('__iter__'):
        iter_name = table_options['__iter__']
        iter_path = table_options['__path__']
        iter_list = self.field_value(iter_name, [], data)
        root_name = table_options.get('__root__', 'root')
```

```
            if iter_list:
                for item in iter_list:
                    logger.info("+ Child %s %s", item.get('id', '-'), iter_path)
                    item[root_name] = data
                    self.process_model_item(model_class, item)
        else:
            return self.process_model_item(model_class, data)

    def process_tender(self, tender):
        data = self.get_from_cache(tender)
        if not data:
            data = self.client.get_tender(tender.id)['data']
            self.save_to_cache(tender, data)
        if self.fill_cache:
            return
        with self.database.transaction():
            try:
                for model_name, model_class in self.models.items():
                    self.process_model_data(model_class, data)
            except Exception as e:
                message = str(e) + " rootID:%s" % data.get('id')
                if self.ignore_errors:
                    self.database.rollback()
                    logger.error(message)
                    return
                raise type(e), type(e)(message), sys.exc_info()[2]

    def get_from_cache(self, tender):
        if not self.cache_model:
            return None
        total_count = self.cache_hit_count + self.cache_miss_count
        if total_count > 0 and total_count % 1000 == 0:
            usage = 100.0 * self.cache_hit_count / total_count
            logger.info("-- Cache hit %d miss %d usage %1.0f %%",
                self.cache_hit_count, self.cache_miss_count, usage)
        try:
            item = self.cache_model.get(
                self.cache_model.tender_id == tender.id,
                self.cache_model.dateModified == tender.dateModified)
        except self.cache_model.DoesNotExist:
            self.cache_miss_count += 1
            return None
        self.cache_hit_count += 1
        return munchify(json.loads(zlib.decompress(item.gzip_data)))

    def save_to_cache(self, tender, data):
        if not self.cache_model:
            return False
        gzip_data = zlib.compress(json.dumps(data))
        if len(gzip_data) > self.cache_max_size:
            logger.warning("Too big for cache %s", tender.id)
            return False
        cache_item = self.cache_model(tender_id=tender.id,
                dateModified=tender.dateModified,
                gzip_data=gzip_data)
        try:
            cache_item.save(force_insert=True)
        except peewee.IntegrityError:
            cache_item.save()
        return True

    def onpreload(self, count, last):
        logger.info("Preload %d last %s", count, last['dateModified'])
```

```
    def run(self):
        offset = self.client_config.get('offset', '')
        limit = int(self.client_config.get('limit') or 0)
        self.total_processed = 0

        if offset:
            self.client.params['offset'] = offset

        while True:
            tenders_list = self.client.preload_tenders(callback=self.onpreload)

            for tender in tenders_list:
                if offset and offset > tender.dateModified:
                    logger.debug("Ignore %s %s", tender.id, tender.dateModified)
                    continue

                dateModified = tender.dateModified.replace('T', ' ')[:19]
                logger.info("Process %s %s", tender.id, dateModified)
                self.process_tender(tender)
                self.total_processed += 1

                if limit and self.total_processed >= limit:
                    logger.info("Reached limit, stop.")
                    return

            if not tenders_list:
                logger.info("No more records.")
                break

    def run_debug(self):
        with open('debug/tender.json') as f:
            tender = json.load(f)
        data = munchify(tender['data'])
        for model_name, model_class in self.models.items():
            self.process_model_data(model_class, data)


def run_app(args):
    config = MyConfigParser(allow_no_value=True)
    for inifile in args.config:
        config.test(inifile)
        config.read(inifile)

    app = TendersToSQL(config, args)
    return app.run()


def main():
    description = "Prozorro API to SQL server bridge, v0.4"
    parser = ArgumentParser(description=description)
    parser.add_argument('config', nargs='+', help='ini file(s)')
    parser.add_argument('-o', '--offset', type=str, help='client api offset')
    parser.add_argument('-l', '--limit', type=int, help='client api limit')
    parser.add_argument('-r', '--resume', action='store_true', help='dont drop table')
    parser.add_argument('-i', '--ignore', action='store_true', help='ignore errors')
    parser.add_argument('-x', '--drop-cache', action='store_true', help="clear cache")
    parser.add_argument('-f', '--fill-cache', action='store_true', help="only save to cache")
    parser.add_argument('-n', '--no-cache', action='store_true', help="don't use cache")
    parser.add_argument('-d', '--debug', action='store_true', help='print traceback')
    parser.add_argument('-p', '--pause', action='store_true', help='pause before exit')
    args = parser.parse_args()
```

```
    for inifile in args.config:
        logging.config.fileConfig(inifile)
        break
    if args.debug:
        logger.setLevel(logging.DEBUG)

    exit_code = 0

    try:
        run_app(args)
    except KeyboardInterrupt:
        logger.error("Keyboard Interrupt")
        pass
    except Exception as e:
        if args.debug:
            logger.exception("Got Exception")
        else:
            logger.error(e)
        exit_code = 1
    if args.pause:
        print "Press Enter to continue..."
        raw_input()
    return exit_code
if __name__ == '__main__':
    sys.exit(main())
```

Додаток 2

```
regOptions = ['PI_Hrones    ';
              'PI_MIGO      ';
              'PI_Robust    ';
              'PI_Skogestad ';
              'PI_Ziegler   ';
              'PID_Hrones   ';
              'PID_MIGO     ';
              'PID_Robust   ';
              'PID_Skogestad';
              'PID_Ziegler  '];
PropsData = {'KpData      ',
             'KiData      ',
             'KdData      ',
             'RiseTime    ',
             'SettlingTime',
             'SettlingMin ',
             'SettlingMax ',
             'Overshoot   ',
             'Peak        ',
             'PeakTime    '};
methodsStr = cellstr(regOptions);
defaultSize = size(regOptions);
MainSize = defaultSize(1);

%Variable init
i            = 1;
NewProps     = [];
KpData       = [];
KiData       = [];
KdData       = [];
RiseTime     = [];
SettlingTime = [];
SettlingMin  = [];
SettlingMax  = [];
Overshoot    = [];
Peak         = [];
PeakTime     = [];

while i < MainSize+1
    load(strcat(methodsStr{i},'.mat'));
    %Compensator data
    [Kp,Ki,Kd] = piddata(C);
    StepData = stepinfo(T_r2y);
    save(strcat('compensatorData/',methodsStr{i}),'Kp','Ki','Kd','StepData');
    %All possible Properties Of StepData
    Props = [Kp;
             Ki;
             Kd;

             StepData.RiseTime;
             StepData.SettlingTime;
             StepData.SettlingMin;
             StepData.SettlingMax;
             StepData.Overshoot;
             StepData.Peak;
             StepData.PeakTime];
    KpData       = [KpData;Kp];
    KiData       = [KiData;Ki];
```

```
    KdData       = [KdData;Kd];
    RiseTime     = [RiseTime;    StepData.RiseTime];
    SettlingTime = [SettlingTime;StepData.SettlingTime];
    SettlingMin  = [SettlingMin; StepData.SettlingMin];
    SettlingMax  = [SettlingMax; StepData.SettlingMax];
    Overshoot    = [Overshoot;   StepData.Overshoot];
    Peak         = [Peak;        StepData.Peak];
    PeakTime     = [PeakTime;    StepData.PeakTime];
    NewProps =
[KpData,KiData,KdData,RiseTime,SettlingTime,SettlingMin,SettlingMax,Overshoot,Peak,Pe
akTime];
    i = i+1;
end
NewData{11,11}     = [];
NewData(1,1)       = {'-'};
NewData(2:11,1)    = methodsStr;
NewData(1,2:11)    = PropsData;
NewData(2:11,2:11) = num2cell(NewProps);
%Saving data to xls file
xlswrite('compensatorData/compensatorData.xls',NewData)
%Linear Quadratic Gaussian
load('LQG_Fast.mat');
StepData = stepinfo(T_r2y);
save('compensatorData/LQG_Fast.mat','C','StepData');
%First try of Quadratic Gaussian
load('LQG_Fast.mat');
StepData = stepinfo(T_r2y);
save('compensatorData/LQG_Second.mat','C','StepData');
```

impulse.m(будує імпульсні характеристики систем):

```
close all;
load('PID_Ziegler.mat');
pidZieglerW = T_r2y;
impulse(pidZieglerW);
title('pidZieglerW');
saveas(gcf,'plots/impulsepidZiegler','jpg');
figure;
load('PID_Skogestad.mat');
pidSkogestadW = T_r2y;
impulse(pidSkogestadW);
title('pidSkogestadW');
saveas(gcf,'plots/impulsepidSkogestad','jpg');
figure;
load('PID_Robust.mat');
pidRobustW = T_r2y;
impulse(pidRobustW);
title('pidRobustW');
saveas(gcf,'plots/impulsepidRobust','jpg');
figure;
load('PID_MIGO.mat');
pidMigoW = T_r2y;
impulse(pidMigoW);
title('pidMigoW');
saveas(gcf,'plots/impulsepidMIGO','jpg');
figure;
load('PID_Hrones.mat');
pidHronesW = T_r2y;
impulse(pidHronesW);
title('pidHronesW');
saveas(gcf,'plots/impulsepidHrones','jpg');
figure;
```

```
load('PI_Ziegler.mat');
piZieglerW = T_r2y;
impulse(piZieglerW);
title('piZieglerW');
saveas(gcf,'plots/impulsepiZiegler','jpg');
figure;
load('PI_Skogestad.mat');
piSkogestadW = T_r2y;
impulse(piSkogestadW);
title('piSkogestadW');
saveas(gcf,'plots/impulsepiSkogestad','jpg');
figure;
load('PI_MIGO.mat');
piMigoW = T_r2y;
impulse(piMigoW);
title('piMigoW');
saveas(gcf,'plots/impulsepiMIGO','jpg');
figure;
load('PI_Hrones.mat');
piHronesW = T_r2y;
impulse(piHronesW);
title('piHronesW');
saveas(gcf,'plots/impulsepiHrones','jpg');
figure;
load('PI_Robust.mat');
piRobustW = T_r2y;
impulse(piRobustW);
title('piRobustW');
saveas(gcf,'plots/impulsepiRobust','jpg');
figure;
load('LQG_Fast.mat');
lqgAgressive = T_r2y;
impulse(lqgAgressive);
title('LQG aggresive');
saveas(gcf,'plots/impulseLQGaggresive','jpg');
figure;
load('LQG_Second.mat');
LQG = T_r2y;
impulse(LQG);
title('LQG');
saveas(gcf,'plots/impulseLQGaggresive','jpg');
%close all;
impulse(pidZieglerW, pidSkogestadW, pidRobustW,pidMigoW,pidHronesW);
legend('pidZiegler','pidSkogestad','pidRobust','pidMigo','pidHrones','Location','Best
');
saveas(gcf,'plots/impulsepidmain','jpg');
figure;
impulse(piZieglerW,piSkogestadW,piMigoW,piHronesW,piRobustW);
legend('piZiegler','piSkogestad','piMigo','piHrones','piRobust','Location','Best');
saveas(gcf,'plots/impulsepiMain','jpg');
figure;
impulse(LQG,lqgAgressive);
legend('LQG','LQG Agressive','Location','Best');
saveas(gcf,'plots/impulseLQGcompare','jpg');
close all;
```

expand.m(будує перехідну систему для кожної системи):

```
close all;
load('PID_Ziegler.mat');
pidZieglerW = T_r2y;
step(pidZieglerW);
title('pidZieglerW');
```

```
saveas(gcf,'plots/steppidZiegler','jpg');
figure;
load('PID_Skogestad.mat');
pidSkogestadW = T_r2y;
step(pidSkogestadW);
title('pidSkogestadW');
saveas(gcf,'plots/steppidSkogestad','jpg');
figure;
load('PID_Robust.mat');
pidRobustW = T_r2y;
step(pidRobustW);
title('pidRobustW');
saveas(gcf,'plots/steppidRobust','jpg');
figure;
load('PID_MIGO.mat');
pidMigoW = T_r2y;
step(pidMigoW);
title('pidMigoW');
saveas(gcf,'plots/steppidMIGO','jpg');
figure;
load('PID_Hrones.mat');
pidHronesW = T_r2y;
step(pidHronesW);
title('pidHronesW');
saveas(gcf,'plots/steppidHrones','jpg');
figure;
load('PI_Ziegler.mat');
piZieglerW = T_r2y;
step(piZieglerW);
title('piZieglerW');
saveas(gcf,'plots/steppiZiegler','jpg');
figure;
load('PI_Skogestad.mat');
piSkogestadW = T_r2y;
step(piSkogestadW);
title('piSkogestadW');
saveas(gcf,'plots/steppiSkogestad','jpg');
figure;
load('PI_MIGO.mat');
piMigoW = T_r2y;
step(piMigoW);
title('piMigoW');
saveas(gcf,'plots/steppiMIGO','jpg');
figure;
load('PI_Hrones.mat');
piHronesW = T_r2y;
step(piHronesW);
title('piHronesW');
saveas(gcf,'plots/steppiHrones','jpg');
figure;
load('PI_Robust.mat');
piRobustW = T_r2y;
step(piRobustW);
title('piRobustW');
saveas(gcf,'plots/steppiRobust','jpg');
figure;
load('LQG_Fast.mat');
lqgAgressive = T_r2y;
step(lqgAgressive);
title('LQG aggresive');
saveas(gcf,'plots/stepLQGaggresive','jpg');
figure;
load('LQG_Second.mat');
LQG = T_r2y;
```

```
step(LQG);
title('LQG');
saveas(gcf,'plots/stepLQGaggresive','jpg');
%close all;
load('transferFunction.mat')
step(pidZieglerW, pidSkogestadW, pidRobustW,pidMigoW,pidHronesW,W);
legend('pidZiegler','pidSkogestad','pidRobust','pidMigo','pidHrones','Ordinary','Loca
tion','Best');
saveas(gcf,'plots/steppidmain','jpg');
figure;
step(piZieglerW,piSkogestadW,piMigoW,piHronesW,piRobustW,W);
legend('piZiegler','piSkogestad','piMigo','piHrones','piRobust','Ordinary','Location'
,'Best');
saveas(gcf,'plots/steppiMain','jpg');
figure;
step(LQG,lqgAgressive);
legend('LQG','LQG Agressive','Location','Best');
saveas(gcf,'plots/stepLQGcompare','jpg');
close all;
```

nyquistNew.m (будує та зберігає годограф Найквіста для кожної системи):

```
close all;
load('PID_Ziegler.mat');
pidZieglerW = T_r2y;
nyquist(pidZieglerW);
title('pidZieglerW');
saveas(gcf,'plots/nyquistpidZiegler','jpg');
figure;
load('PID_Skogestad.mat');
pidSkogestadW = T_r2y;
nyquist(pidSkogestadW);
title('pidSkogestadW');
saveas(gcf,'plots/nyquistpidSkogestad','jpg');
figure;
load('PID_Robust.mat');
pidRobustW = T_r2y;
nyquist(pidRobustW);
title('pidRobustW');
saveas(gcf,'plots/nyquistpidRobust','jpg');
figure;
load('PID_MIGO.mat');
pidMigoW = T_r2y;
nyquist(pidMigoW);
title('pidMigoW');
saveas(gcf,'plots/nyquistpidMIGO','jpg');
figure;
load('PID_Hrones.mat');
pidHronesW = T_r2y;
nyquist(pidHronesW);
title('pidHronesW');
saveas(gcf,'plots/nyquistpidHrones','jpg');
figure;
load('PI_Ziegler.mat');
piZieglerW = T_r2y;
nyquist(piZieglerW);
title('piZieglerW');
saveas(gcf,'plots/nyquistpiZiegler','jpg');
figure;
load('PI_Skogestad.mat');
piSkogestadW = T_r2y;
nyquist(piSkogestadW);
title('piSkogestadW');
```

```
saveas(gcf,'plots/nyquistpiSkogestad','jpg');
figure;
load('PI_MIGO.mat');
piMigoW = T_r2y;
nyquist(piMigoW);
title('piMigoW');
saveas(gcf,'plots/nyquistpiMIGO','jpg');
figure;
load('PI_Hrones.mat');
piHronesW = T_r2y;
nyquist(piHronesW);
title('piHronesW');
saveas(gcf,'plots/nyquistpiHrones','jpg');
figure;
load('PI_Robust.mat');
piRobustW = T_r2y;
nyquist(piRobustW);
title('piRobustW');
saveas(gcf,'plots/nyquistpiRobust','jpg');
figure;
load('LQG_Fast.mat');
lqgAgressive = T_r2y;
nyquist(lqgAgressive);
title('LQG aggresive');
saveas(gcf,'plots/nyquistLQGaggresive','jpg');
figure;
load('LQG_Second.mat');
LQG = T_r2y;
nyquist(LQG);
title('LQG');
saveas(gcf,'plots/nyquistLQGaggresive','jpg');
%close all;
nyquist(pidZieglerW, pidSkogestadW, pidRobustW,pidMigoW,pidHronesW);
legend('pidZiegler','pidSkogestad','pidRobust','pidMigo','pidHrones','Location','Best
');
saveas(gcf,'plots/nyquistpidmain','jpg');
figure;
nyquist(piZieglerW,piSkogestadW,piMigoW,piHronesW,piRobustW);
legend('piZiegler','piSkogestad','piMigo','piHrones','piRobust','Location','Best');
saveas(gcf,'plots/nyquistpiMain','jpg');
figure;
nyquist(LQG,lqgAgressive);
legend('LQG','LQG Agressive','Location','Best');
saveas(gcf,'plots/nyquistLQGcompare','jpg');
close all;
```