Донбаська державна машинобудівна академія
Кафедра автоматизації виробничих процесів

МЕТОДИЧНІ ВКАЗІВКИ

ДО КУРСОВОГО ПРОЕКТУВАННЯ ПО ДИСЦИПЛІНІ

«ЦИФРОВІ СИСТЕМИ КЕРУВАННЯ І ОБРОБКИ ІНФОРМАЦІЇ»

(для студентів спеціальності 151
«Автоматизація та комп‘ютерно-інтегровані технології»)
Кваліфікаційний рівень - магістр

Краматорськ 2018

УДК 681.3.06 (075.8)

Методичні вказівки до курсового проектування по дисципліні «Цифрові системи керування й обробки інформації» (для студентів спеціальності 151 «Автоматизація та комп‘ютерно-інтегровані технології»). Кваліфікаційний рівень – магістр. / Укл. О. О. Сердюк – Краматорськ: ДДМА, 2018. – 64 с.

Приводяться структура й склад курсового проекту. Викладаються методики рішення найбільш важливих завдань проектування систем керування й обробки інформації – аналізу об'єкта керування, математичного опису й моделювання системи, а також рекомендації з розробки програмного забезпечення й створення документації до проекту.
У додатках представлені необхідні довідкові матеріали.

Укладач СЕРДЮК Олександр Олександрович, доцент,

ВСТУП

Курсовий проект по дисципліні «Цифрові системи керування й обробки інформації» виконується з метою придбання вмінь і навичок системного сприйняття завдань проектування та розкриття взаємозв'язку між концепцією керування й конструкторською концепцією в рамках конкретного завдання автоматизації виробництва.

Результати захисту курсового проекту визначають рівні теоретичної й практичної підготовки студента до виконання ***науково-дослідних та інженерних завдань***, установлених кваліфікаційною характеристикою спеціальності стосовно до області проектування систем автоматизації.

Предметом проектування є один з наступних варіантів:

- *Перший варіант* – цифрова система керування обладнанням на технологічному рівні, що працює в ***режимі реального часу***.
- *Другий варіант* – автоматизована виконавча система виробництва MES-рівня, що забезпечує оперативну обробку інформації ***в режимі машинного часу***.

Зміст курсового проекту повинен повністю розкривати сутність розв'язку завдань, поставлених в індивідуальній темі проекту.

Як методологічний напрямок повинен бути прийнятий ***системний підхід***, який дозволяє розкрити цілісність процесу в умовах різноманіття типів зв'язків, а також об'єднати цифрові й аналогові канали, механічні й електронні пристрої, матеріальні та інформаційні потоки, програмні й апаратні засоби.

Відповідно до цього студенти повинні виконати наступне:

1. На основі системного аналізу технологічного процесу визначити проблемні ситуації, які стримують вирішення завдань підвищення ефективності процесу.
2. На основі огляду літературних джерел розробити пропозиції щодо удосконалення існуючої системи автоматизації.
3. Виконати моделювання системи керування або предметної області.
4. Для системи керування обладнанням розробити схеми з'єднань апаратних компонентів системи. Для автоматизованої виконавчої системи розробити модель програмного додатку та модель бази даних.
5. Розробити програмне забезпечення системи автоматизації.
6. Створити документацію проекту відповідно до вимог стандартів.

На всіх етапах проектування студенти повинні використовувати в якості інструментів проектування спеціальні програмні засоби, що забезпечують сучасні вимоги до якості проекту й скорочення часу на його розробку.

Тематика курсових проектів формується кафедрою й затверджується деканом факультету. При формуванні тематики враховуються теми дипломних робіт, визначені при проходженні практики. Зразкова тематика курсових проектів наведена в додатку А.

Завдання на курсове проектування розробляється керівником курсового проекту. У завданні визначаються тема проекту, вихідні дані для розробки проекту, зміст пояснювальної записки й графічної частини проекту, строк виконання. Зразок бланка завдання наведений у додатку Б.

Результати проектування повинні бути представлені в розрахунково-пояснювальній записці обсягом до 40 сторінок тексту на аркушах формату А4 і в графічній частині обсягом 4-5 аркушів формату А3 або А4.

Розрахунково-пояснювальна записка повинна бути оформлена відповідно до ГОСТ 2.105-85 «Общие требования к текстовым документам», а також ДСТУ 3008-95 «Документація. Звіти в сфері науки й техніки». Розрахунково-пояснювальна записка повинна містити титульний лист, завдання, реферат, зміст, основну частину, перелік посилань і додатки. Зразок титульного листа розрахунково-пояснювальної записки представлений у додатку В.

Рекомендується наступна структура й обсяг кожного елемента основної частини розрахунково-пояснювальної записки:

Для проекту цифрової системи керування обладнанням

Вступ (1 стор.).

1. Аналіз технологічного процесу й постановка завдань проектування (5 стор.).
2. Визначення основних параметрів системи (8-10 стор.).
3. Проектування системи (10-12 стор.).
4. Моделювання системи (5 стор.)
5. Розробка програмного забезпечення (5-8 стор.).

Висновки (1 стор.).

Графічна частина складається із схем та плакатів. Схеми повинні бути виконані з дотриманням вимог ГОСТ 2.743-82 «Схемы цифровой техники», ГОСТ 2.759-82 «Схемы аналоговой техники», ГОСТ 2.721-74 «Обозначения условные графические общего применения», ГОСТ 19002-80, 19003-80 «Правила выполнения блок-схем алгоритмов», ГОСТ 2.702-75 «Правила выполнения электрических схем», ГОСТ 2.710-81 «Обозначения буквенно-цифровые в электрических схемах». Плакати використовуються для представлення результатів теоретичних та експериментальних досліджень. Плакат повинен мати назву (зверху), текстові та графічні елементи на плакаті не повинні викликати напруження у сприйманні.

Для проекту автоматизованої системи виробництва (розробка програмного додатка і бази даних)

Вступ (1 стор.).

1. Аналіз предметної області, літературний огляд й постановка завдань проектування (5 стор.).
2. Розробка моделі нового бізнес-процесу (IDEF0-модель, 5-7 стор.).
3. Моделювання програмного додатка – діаграма варіантів використання, діаграма класів, діаграма поведінки (UML-моделі, 10-15 стор.) .
4. Моделювання бази даних (ER-моделі, 6-8 стор.)
5. Розробка схеми розгортання системи (сервер, станція, мережа)

Висновки (1 стор.).

Графічна частина такого проекту складається із діаграм і плакатів. Діаграми являються програмними документами і тому оснащуються основним підписом як теоретичне креслення.

Графічні зображення на діаграмах повинні відповідати загально прийнятій нотації діаграм IDEF0, UML та ER.

1 АНАЛІЗ БАЗОВОГО ПРОЦЕСУ ТА ПОСТАНОВКА ЗАВДАНЬ ПРОЕКТУВАННЯ

Рекомендований перелік питань, які повинні бути розглянуті у першому розділі, наведено в таблиці 1.1.

Таблиця 1.1

Варіант предмету проектування	Перелік питань першого розділу
Система керування обладнанням технологічного рівня	1.1 Аналіз існуючого технологічного процесу і його недоліків 1.2 Аналіз сучасних підходів і технологічних рішень (літературний огляд) 1.3 Аналіз вимог до нової системи керування й розробка завдань проектування
Автоматизована виконавча система виробництва (інформаційна система)	1.1 Аналіз предметної області (існуючого процесу організації й керування виробництвом) 1.2 Аналіз сучасних підходів до створення інформаційних систем (літературний огляд) 1.3 Аналіз вимог до нової інформаційної системи й розробка завдань проектування

Приступаючи до проектування автоматизованої системи, необхідно поставити ***цілі проектування***.

Цілі проектування зазвичай перебувають у площині ***економічних показників***, які, у свою чергу, визначаються однією із двох складових економічної ефективності – прямим ефектом або непрямим ефектом.

Для систем автоматичного керування обладнанням показниками ефективності є, головним чином, трудові й вартісні витрати, що ставляться до прямого ефекту. На величину цих витрат впливають наступні виробничі показники:

1. *Точність контролю, обліку й підтримки технологічних параметрів продукції.* Автоматизовані системи керування дозволяють зменшити трудові й вартісні витрати, збільшивши при цьому прибуток. Так, наприклад, якщо на прокатному стані підтримувати товщину смуги в межах нижнього поля допуску, то можна суттєво знизити витрати металу на погонний метр смуги. Економічно вигідне підвищення точності систем виміру й обліку споживання енергетичних ресурсів, тому що це сприяє зменшенню упущеної вигоди. Завдання підвищення точності повинні вирішуватися шляхом

ретельного аналізу всіх погрішностей і наступної розробки комплексних заходів.

2. *Витрата енергетичних ресурсів (електроенергія, газ, вода і т.д.), сировини й матеріалів на технологічні потреби*. Автоматизовані системи керування дозволяють застосувати найбільш ефективні алгоритми керування, які оптимізують споживання енергетичних ресурсів, сировини й матеріалів, що сприяє зменшенню їх витрат.

3. *Продуктивність процесу*. Продуктивність – це показник, який визначається як величина, зворотна сумі втрат часу на здійснення робочих рухів і виконання допоміжних операцій. Тому для підвищення продуктивності необхідно провести ретельний аналіз втрат часу й виключити або зменшити причини цих втрат.

4. *Коефіцієнт технічного використання*. Коефіцієнт технічного використання відбиває частку часу знаходження встаткування в працездатному стані щодо деякого часу експлуатації, наприклад, одного року. Він характеризує простої устаткування, пов'язані з його обслуговуванням і ремонтом. Системи автоматичного керування дозволяють підвищити *надійність* роботи устаткування й скоротити втрати часу на відновлення працездатності, що сприяє збільшенню дійсного фонду часу роботи устаткування, а також зменшенню матеріальних витрат, пов'язаних з обслуговуванням і ремонтом.

Ефективними засобами підвищення надійності є:

- вибір елементів з найменшою ймовірністю відмови;
- проектування засобів захисту від аварій;
- розробка розвинутої системи діагностики.

Для автоматизованої виконавчої системи виробництва (MES-системи) економічна ефективність визначається непрямим ефектом, інтегральним показником якого є *рентабельність*. Рентабельність відбиває ступінь ефективності використання матеріальних, трудових і грошових ресурсів виробництва. MES-системи поліпшують фінансові показники підприємства шляхом підвищення віддачі основних фондів, зменшення відсотка браку продукції, прискорення обігу коштів, забезпечення своєчасності поставок, розширення мережі клієнтів.

Передпроектний аналіз

Проектування системи автоматизації повинне базуватися на результатах системного аналізу технологічного процесу або предметної області програмного додатка. При виконанні цієї роботи потрібно визначити мету, об'єкт, предмет і зміст аналізу.

Метою аналізу технологічного процесу (предметної області) є конкретизація вимог до автоматизованої системи керування, а також

знаходження способів досягнення бажаного результату при мінімальних витратах.

Предметом аналізу повинен бути:

- ***об'єкт автоматизації*** (виробничий процес або предметна область організаційної діяльності);
- ***засоби автоматизації*** (апаратура керування, програмне забезпечення, канали передачі й т.п.).

Зміст аналізу – це перелік питань або завдань, які повинні бути вирішені. При аналізі технологічного процесу потрібно встановити недоліки існуючих засобів керування, а також визначити можливі способи їх усунення з урахуванням сучасних досягнень науково-технічного прогресу. При виконанні аналізу предметної області необхідно визначити недоліки в організації виконуваних робіт, процеси, не охоплені оперативним контролем, а також створити специфікацію функцій інформаційної системи.

Особливо уважно слід вивчити вимоги до технологічного процесу й особливості керування процесом.

Вимоги до технологічного процесу можна знайти в технічній документації – технічних умовах, галузевих нормативних документах і стандартах, в описах технологічних процесів, кресленнях, керівних матеріалах і т.п. Для обліку особливостей керування слід з'ясувати думки фахівців.

При розробці способів усунення виявлених недоліків не можна покладатися тільки на власні знання, а слід використати передовий досвід. Слід також враховувати, що в інвестиційному проекті найбільша увага приділяється його життєвому циклу – інвестори завжди вимагають надійного захисту своїх інвестицій на тривалий час (принцип «вкластися й не думати про модернізацію десять-п'ятнадцять років»). Тому розроблювач проекту повинен застосовувати в проекті перспективні рішення в організації апаратних і програмних засобів системи керування.

Саме ці обов'язки ставлять перед будь-яким проектувальником завдання аналізу наукової літератури й інтернет-джерел.

Підсумком виконання першого розділу є:

1. Специфікація вимог до проектованої системи (як повинна бути влаштована система і які функції вона повинна виконувати?);
2. Перелік завдань проектування (що потрібно зробити для забезпечення вимог до нової системи керування?)

Формулюючи завдання проектування, розроблювач проекту, по суті, повинен логічно зв'язати висновки аналізу технологічного процесу зі змістом наступних розділів пояснювальної записки.

Завдання проектування рекомендується формулювати так, як показано в таблиці 1.2.

Таблиця 1.2

Предмет проектування	**Перелік завдань**
Система керування устаткуванням технологічного рівня	1 Розрахунки основних параметрів системи керування й устаткування 2 Розробка структурної схеми системи 3 Конфігурування системи керування 4 Моделювання процесу керування виконавчим пристроєм 5 Розробка принципіальних схем 6 Розробка структури програмного забезпечення й програмного коду (для однієї із задач керування)
Автоматизована виконавча система керування виробництвом (із проектуванням програмного додатка)	1 Розробка IDEF0-моделей предметної області «як є» і «як повинно бути». 2 Розробка UML-моделей для програми: • діаграми варіантів використання програми ; • діаграми послідовності для одного із варіантів використання; • діаграми класів програми. 3 Розробка ER-діаграми бази даних. 4 Розробка схеми розгортання додатка (апаратні засоби та їх зв‘язки)

При розробці проекту слід ***забезпечити логічну зв‘язність*** у розв'язанні завдань проектування. Рекомендується наступний порядок виконання проекту (залежно від теми).

При проектуванні системи керування устаткуванням необхідно спочатку розрахувати час робочого циклу, а також параметри периферійних пристроїв, що забезпечують необхідну точність і динаміку.

Після цих розрахунків можна приступити до вибору апаратури керування, вимірювальних перетворювачів і виконавчих засобів, а також до розробки структурної схеми.

Для підтвердження правильності компонувальних рішень необхідно виконати конфігурування системи. Конфігурування дозволяє визначити також адреси всіх каналів сигнальних модулів. Перевірка правильності конфігурації проводиться командою *Consistency Check*.

Якість керування оцінюється шляхом моделювання каналу керування, який включає регулятори, перетворювач енергії, двигун, а також пристрої

зворотного зв'язку. Параметрами оцінки є стійкість процесу, швидкодія й точність системи керування.

Якщо результат моделювання позитивний, виконується розробка схем з'єднань засобів керування з виконавчими пристроями, а також схем підключень каналів уведення-виводу.

На завершення проводиться розробка структури програми користувача й створення програмного коду для однієї із задач керування.

При ***проектуванні програмного додатку***, що створюється для розв‘язання деяких виробничих завдань, в якості парадигми програмування слід прийняти об'єктно-орієнтоване програмування, базовими концепціями якого є *об'єкти* й *класи*.

На першому етапі необхідно побудувати ***функціональну модель*** предметної області. Функціональне моделювання виконується із застосуванням технології IDEF0.

Спочатку треба розробити модель функцій існуючої організації бізнес-процесів – модель «як є» (по англ. «AS IS»). Ця модель піддається ретельному аналізу з метою встановлення «вузьких» місць, які можна усунути за допомогою комп‘ютерних засобів, точніше програмного додатка. Для уточнення функцій програмного додатка треба розробити другу модель – модель – «як повинно бути» (по англ. «TO BE»).

В основу цієї моделі повинні бути покладені класи, які, по суті, становлять словник предметної області. Кожний клас повинен реалізувати певну поведінку, а сукупність класів створити бажану поведінку системи. Завдання полягає в тому, щоб бажана поведінка, яка на початковому етапі ще не зовсім зрозуміла, була реалізована в повному обсязі відповідно до функцій системи. Тому після побудови моделі TO BE необхідно створити ще ряд моделей, які дозволять уточнити функції системи, доки вони не задовольнять користувачів.

Після ідентифікації сутностей можна перейти до побудови інфологічної моделі бази даних і створенню даталогічних (логічної й фізичної) моделей, які «розуміє» система керування базою даних (СУБД). Для забезпечення функціонування бази даних необхідно розробити інтерфейси користувачів, а також створити відповідні сценарії (командні файли).

Уточнення поведінки об'єктів здійснюється за допомогою діаграм послідовностей. Після їх побудови слід перейти до моделювання статичної структури додатка – до діаграми класів. На діаграмах класів повинні бути вказані всі необхідні для роботи додатка відомості – імена змінних, типи даних, операції й методи, типи взаємодій об'єктів.

Моделювання програми здійснюється з використанням універсальної мови моделювання UML і CASE-інструментів. Застосування CASE-інструментів, наприклад, програми Rational Rose, дозволяє прискорити створення працюючого прототипу додатка й одержати необхідну програмну документацію. При цьому, якщо код буде неповним, його можна буде дописати звичайним шляхом, а потім знову регенерувати в модель.

Відповідно до вимог стандартів OPC (OLE Process Control – відкриті системи процесів керування) програмний додаток повинен функціонувати в інформаційному просторі, тобто він повинен бути інтегрованим в систему інформаційного забезпечення підприємства. Тому на завершальному етапі необхідно передбачити побудову діаграми розгортання програмної системи, яка являє собою модель взаємодії додатка із зовнішнім середовищем.

Розгортання системи в клієнт-серверній архітектурі потребує відображенням наступних компонентів:

- *серверна частина* (для зберігання й обробки інформації);
- *клієнтська частина* (робочий інструмент користувача);
- *мережа*, яка забезпечує взаємодію між клієнтом і сервером відповідно до заданих специфікацій.

2 ВИЗНАЧЕННЯ ОСНОВНИХ ПАРАМЕТРІВ СИСТЕМИУПРАВЛІННЯ ОБЛАДНАННЯМ

Процес проектування системи управління обладнанням починається з визначення основних параметрів майбутньої системи:

1. Визначення тривалості робочого циклу програми управління.
2. Розрахунки потрібної точності й швидкодії вимірювально-перетворювальних каналів
3. Розрахунки динамічних параметрів виконавчих пристроїв.

При розробці проекту слід враховувати, що системи автоматичного управління являються ієрархічними і мають дві важливі властивості:

1 *Властивість пріоритетів* – в ієрархічній системі команди верхнього рівня обов'язкові для виконання нижчестоящим рівнем.

2 *Властивість взаємозалежності* – функціональні можливості й характеристики нижчестоящого рівня повинні бути узгоджені із завданнями й командами вищого рівня.

Облік цих властивостей вимагає зосередити увагу, у першу чергу, на нижньому рівні, тому що саме тут формуються динамічні характеристики системи і її точність. На цьому рівні, перш за все, потрібно ретельно проаналізувати та визначити період дискретності системи управління, тривалість якого пов‘язана із часом виконання програми.

2.1 Розрахунки тривалості робочого циклу програми

Тривалість робочого циклу користувацької програми системи керування залежить від швидкості зміни технологічного параметра, яким управляє система, і припустимої помилки керування. Чим вище швидкість зміни технологічного параметра, тим менше повинне бути значення тривалості робочого циклу виконання користувацької програми. Тому при керуванні, наприклад, тепловим процесом, де регулюється температура нагрівання, час робочого циклу може становити одиниці секунд, а при керуванні процесом з високою динамікою, наприклад, приводом подачі верстата з ЧПУ, робочий цикл зазвичай має тривалість 0,1-10 мс.

Якщо від виконавчого пристрою, наприклад, приводу лінійних переміщень верстата, не потрібна висока динаміка, то розрахунки тривалості робочого циклу T_0 ведуться по формулі:

$$T_0 \le \sqrt{\frac{2\varepsilon_c}{a_{\text{доп}}}} \qquad (2.1)$$

де ε_c – задана величина відхилення поточної координати при максимальній швидкості ($V_{\max}$) у *мм*; $a_{доп}$ – допустима величина прискорення у *м/с*^{2}.

Якщо ж виконавчий пристрій працює в умовах високих вимог до динаміки, то розрахунки тривалості робочого циклу програми керування слід вести з урахуванням необхідної ***смуги частот системи керування***.

На рисунку 2.1 зображена типова форма бажаної логарифмічної амплітудно-частотної характеристики (ЛАЧХ) цифрової системи керування швидкістю руху. Така характеристика властива для двухкоординатних рухів робочих вузлів верстата із ЧПУ (супорту або столу) при виконанні кругових переміщень, при яких швидкості по координатах змінюються по синусоїдальному закону.

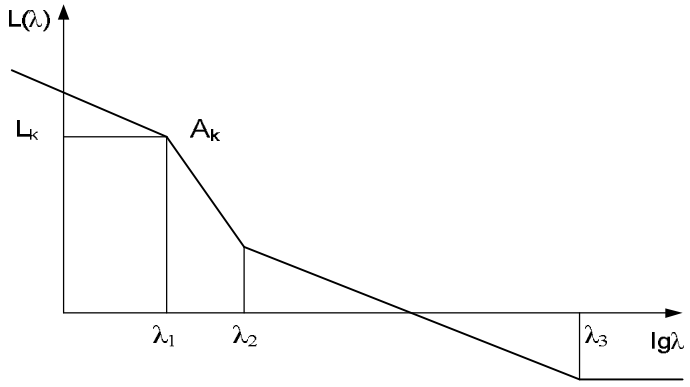

Рисунок 2.1 – Бажана форма ЛАЧХ цифрової системи управління швидкістю руху

Наведена форма ЛАЧХ має наступні особливості:

- можливість усунути позиційну помилку координатного руху – перша асимптота має нахил до осі частот 20 дБ/дек;
- обмеження швидкісної помилки – перша асимптота повинна зайняти певне положення на осі відносної амплітуди $L(\lambda)$;
- забезпечення стійкої роботи системи управління – ЛАЧХ системи перетинає вісь частот асимптототою з нахилом 20 дБ/дек;
- забезпечення необхідної смуги частот і показника коливальності *M* – задається певна довжина асимптоти в частотному діапазоні $\lambda_2 \Lambda\ \lambda_3$.

Для наведеної вище форми бажана ЛАЧХ описується дискретною частотною характеристикою (ДЧХ):

$$W_{ж}(j\lambda)=\frac{K\cdot(1+\tau_2\cdot j\lambda)}{j\lambda\cdot(1+\tau_1\cdot j\lambda)}\cdot\frac{\left(1-j\lambda\frac{T_0}{2}\right)^{q_1}}{\left(1+j\lambda\frac{T_0}{2}\right)^{q_2}}, \qquad (2.2)$$

де λ – псевдочастота; $K=K_1$; $\tau_1=\frac{1}{\lambda_1}$; $\tau_2=\frac{1}{\lambda_2}$; λ_3 – основні параметри, обумовлені вимогами до системи керування; $\frac{\left(1-j\lambda\frac{T_0}{2}\right)^{q_1}}{\left(1+j\lambda\frac{T_0}{2}\right)^{q_2}}$ – характеристика запізнювання вхідних і вихідних сигналів.

Для визначення основних параметрів ДЧХ необхідно перетворити задані параметри технологічного процесу (максимальну швидкість $V_{\max}$ та допустиме прискорення $a_{доп}$) в еквівалентні параметри гармонійного сигналу (амплітуду a_m та частоту ω), а потім визначити положення A_k критичної точки заборонної області ЛАЧХ.

Таке перетворення параметрів можливе у випадку, коли траєкторію руху інструмента, наприклад, фрези, представити функцією:

$$\alpha(t)=\alpha_m\sin\omega t, \qquad (2.3)$$

де $a(t)$ – поточне значення координати; a_m – амплітуда (радіус кругового контуру); ω – кутова швидкість руху.

Перша й друга похідні (швидкість і прискорення) гармонійного сигналу визначаються відомими виразами:

$$\dot{\alpha}_m=\alpha_m\omega, \qquad \ddot{\alpha}_m=\alpha_m\omega^2,$$

де індекси m позначають максимальні значення.

Звідси можна визначити еквівалентні параметри гармонійного руху – частоту $\omega_э$ й амплітуду $a_{эт}$:

$$\omega_э=\frac{\ddot{\alpha}_m}{\dot{\alpha}_m}, \qquad \alpha_{эт}=\frac{\dot{\alpha}_m^2}{\ddot{\alpha}_m}.$$

Для низькочастотної ділянки ЛАЧХ слушне допущення $\lambda\approx\omega$.

Тоді:

$$W\ (j\lambda) \approx W\ (j\omega). \tag{2.4}$$

Якщо відоме значення допустимого відхилення $\theta_m[n]$, то повинна бути виконана умова:

$$|W\ (j\lambda)| > \frac{\alpha_{эт}\ [n]}{\theta_m\ [n]}. \tag{2.5}$$

Для відносної амплітуди $L(\lambda_k)$ ця умова запишеться в наступному виді:

$$L\ (\lambda_k) = 20\ \lg\ |W\ (j\lambda_k)|\ > 20\ \lg\ \frac{\ddot{\alpha}_m^{\,2}}{\ddot{\alpha}_m \cdot \theta_m[n]}, \tag{2.6}$$

У системах керування значення максимальної швидкості $\dot{\alpha}_m = V_{\max}$, допустиме прискорення $\ddot{\alpha}_m = a_{\text{доп}}$, а також допустима величина відхилення траєкторії $\theta_m[n] = \varepsilon_c$ повинні бути відомі.

Тоді, враховуючи умови перетворення, для забезпечення необхідної точності у сталому режимі бажана ЛАЧХ повинна проходити вище критичної точки A_k з координатами:

$$\lambda_k = \frac{a_{\text{доп}}}{V_{\max}}; \tag{2.7}$$

$$L\ (\lambda_k) = 20\ \lg\ \frac{V_{\max}^{\,2}}{a_{\text{доп}}\ \varepsilon_c}. \tag{2.8}$$

При цьому заборонна область обмежується по швидкості першою асимптотою, яка проводиться вліво від точки A_k з нахилом –20 дБ/дек, а по прискоренню – другою асимптотою, яка проводиться вправо від точки A_k із нахилом –40 дБ/дек.

Положення заборонної області показано на рисунку 2.2.

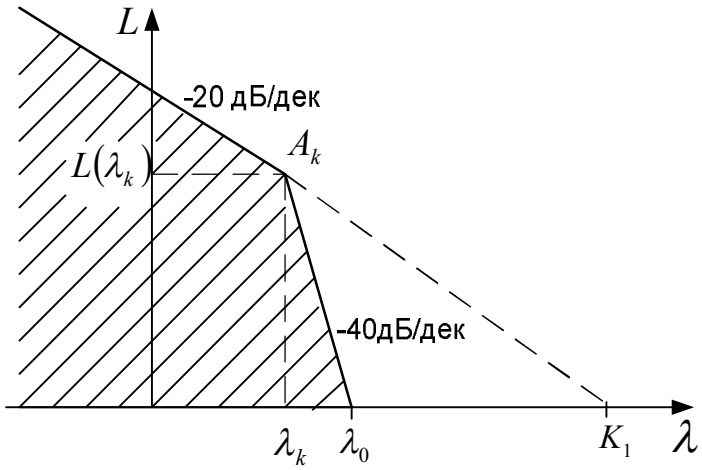

Рисунок 2.2 – Побудова заборонної області за критерієм точності

Швидкісна помилка ε_c й відповідна їй максимальна швидкість $V_{\max}$ визначають необхідну добротність системи по швидкості K_1:

$$K_1 = \frac{V_{\text{мах}}}{\varepsilon_{\text{с}}}. \qquad (2.9)$$

Значення K_1 відповідає точці перетинання з віссю λ лінії, яка продовжує першу низькочастотну асимптоту (рис. 2.2).

При побудові слід дотримуватися наступного порядку.

1. Перша низькочастотна асимптота бажаної ЛАЧХ проводиться з нахилом –20 дБ/дек вище точки A_k на 3 дБ, щоб забезпечити запас стійкості. Підйом характеристики приводить до збільшення коефіцієнта добротності по швидкості на величину $\sqrt{2}$:

$$K_1 = \sqrt{2}\frac{V_{\max}}{\varepsilon_c}. \qquad (2.10)$$

2. Друга асимптота проводиться з нахилом –40 дБ/дек від точки сполучення з координатами (λ_k ; $L(\lambda_k)+3$) до точки перетинання з віссю λ, яка визначає базову частоту λ_0 заборонної області:

$$\lambda_0 = \sqrt{\frac{a_{\text{доп}}}{\varepsilon_c}\sqrt{2}}. \qquad (2.11)$$

3. По заданому показникові коливальності M (зазвичай М=1,2…1,7) визначається частота сполучення другої і третьої асимптот:

$$\lambda_2 = \lambda_0 \sqrt{\frac{M-1}{M}}. \qquad (2.12)$$

4. Третя асимптота з нахилом –20 дБ/дек проводиться від точки λ_2 до точки λ_3, яка також залежить від показника коливальності *M*:

$$\lambda_3 = \lambda_2 \frac{M+1}{M-1}. \qquad (2.13)$$

Четверта асимптота у цифрових системах керування проводиться паралельно осі частот.

Остаточний вид логарифмічної частотної характеристики представлений на рисунку 2.3.

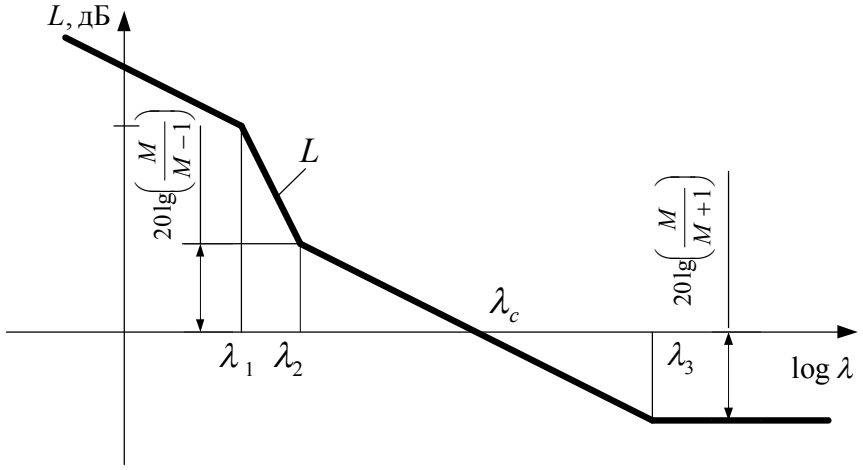

Рисунок 2.3 – Остаточний вид ЛАЧХ для проектованої СУ

У верхньому діапазоні частота λ_3 визначається значенням періоду дискретності T_0:

$$\lambda_3 = 2/T_0.$$

При виборі періоду дискретності слід керуватися вираженням:

$$T_0 \le \frac{2}{\lambda_3}. \qquad (2.14)$$

У наступних розрахунках повинне бути прийняте таке значення T_0, яке задовольняє умові (2.14).

2.2 Розрахунки точності й швидкодії вимірювальних каналів

Визначення розрядності АЦП або ЦАП проводиться з умови забезпечення точності перетворення.

Величина похибки перетворення залежить від швидкості зміни сигналу на вході АЦП або на виході ЦАП.

Для гармонійного сигналу $a(t)$:

$$\alpha(t) = \alpha_m \sin \omega t \tag{2.15}$$

значення похибки визначається з нерівності:

$$\left|\Delta_g(T_0)\right| \le a_m\, \omega T_0 . \tag{2.16}$$

Приклад. Нехай амплітуда $a_m = 1$, швидкість $\omega = 20\ \text{с}^{-1}$, а період дискретності $T = 0{,}01\ \text{с}$. Тоді відносна величина помилки складе:

$$\Delta_g = 1 \cdot 20 \cdot 0{,}01 = 0{,}2 .$$

Як видно із прикладу, навіть при невеликих значеннях частоти сигналу значення похибки становить 0,2, тобто 20% від рівня сигналу, що в багатьох випадках неприпустимо. При збільшенні частоти сигналу це значення буде збільшуватися пропорційно. Єдиним способом зменшення помилки є зменшення періоду дискретизації безперервного сигналу, тобто тривалості робочого циклу системи керування. Тому, наприклад, у системі регулювання вихідного струму частотного перетворювача для забезпечення необхідної точності дискретність повинна бути на рівні 2 мкс при частоті вихідної напруги перетворювача 400-600 Гц (кутова швидкість $\omega \approx 3000\ \text{с}^{-1}$).

Однак похибка дискретизації – це не єдина складова погрішності перетворення. Друга складова цієї погрішності – розв'язна здатність, або абсолютна помилка перетворювача. Зазвичай вона ухвалюється рівній ціні однієї дискрети, тобто 1 у молодшому розряді коду.

Для вибору розрядності коду використовується така характеристика, як динамічний діапазон.

Приклад. Необхідно визначити розрядність ЦАП для подачі сигналу на аналоговий регулятор сили струму, який здійснює регулювання струму від 0 до 600 А з точністю 2 А. Тоді динамічний діапазон буде рівний:

$$D = \frac{600}{2} = 300.$$

Необхідна розрядність (число розрядів) ЦАП визначається цілим числом *ent* (фр. *антье*):

$$b = ent[\log_2 D] + 1,$$

де додаткова 1 означає заміну дробової частини цілим.

Для наведеного вище прикладу необхідна розрядність ЦАП рівна:

$$b = ent\left[\log_2 300\right] + 1 = 9 \text{ (розрядів).}$$

У такий же спосіб визначається й розрядність АЦП.

Приклад. Нехай аналоговий сигнал деякого вимірювального перетворювача змінюється в межах ±5 В. Необхідна точність його вистави повинна становити не більш 1 мВ. Тоді динамічний діапазон буде дорівнювати відношенню подвійної амплітуди до точності вистави:

$$D = \frac{2 \cdot 5}{0{,}001} = 10000.$$

Кількість розрядів АЦП, необхідна для такої вистави, дорівнює:

$$b = ent\left[\log_2 10000\right] + 1 = 14 \text{ (розрядів).}$$

Параметри швидкодії АЦП і ЦАП важливо враховувати завжди, особливо, коли в системі існує кілька каналів аналогових уведень і аналогових виводів. У більшій мірі це важливо для каналів уведення, оскільки час перетворення в АЦП значно більше часу перетворення в ЦАП.

Проблема полягає в тому, що канали (групи каналів) включаються послідовно й загальний час обслуговування каналів (час циклу) залежить від кількості каналів і груп каналів.

На рисунку 2.4 показана тимчасова діаграма обробки сигналів для модуля SM 331 AI 8×14 bit High Speed, який має найбільш високу швидкодію – час перетворення одного каналу становить 52 мкс, а дозвіл модуля (час його обробки) – 625 мкс.

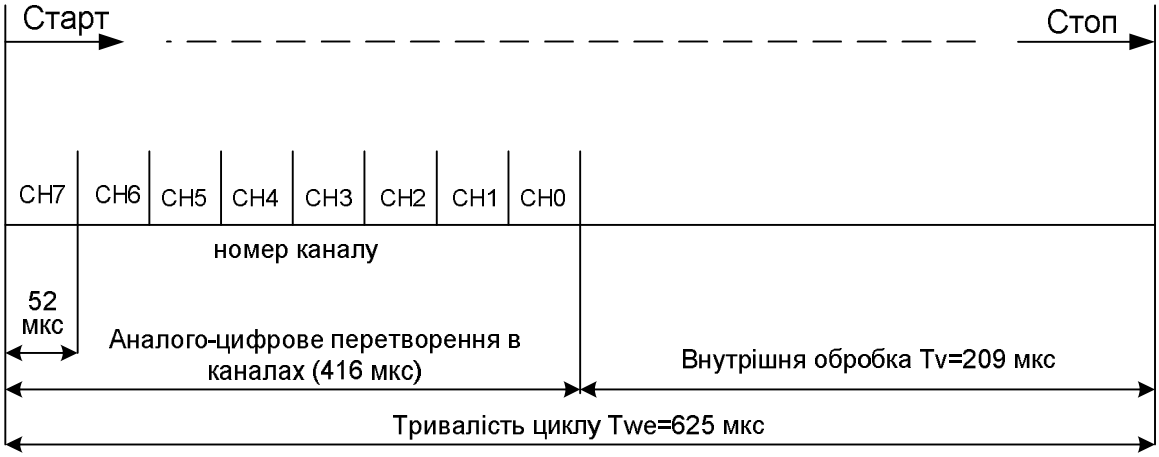

Рисунок 2.4 – Тимчасова діаграма обробки даних у модулі аналогового уведення SM 331 AI 8×14 bit High Speed

Для менш динамічних процесів слід застосовувати звичайний модуль уведення, наприклад, SM 331 AI 8×12 bit, у якого час перетворення каналу становить 100 мс, а час обслуговування модуля становить близько 1200 мс.

Із викладеного слідує, що для обслуговування модулів аналогового уведення необхідно організувати відповідний час циклу або вводити апаратне переривання при готовності модуля передати нові дані.

2.3 Розрахунки динамічних параметрів виконавчих пристроїв

Виконавчі пристрої системи керування, по суті, являють собою фільтр нижніх частот. Параметри такого фільтра слід вибирати так, щоб вони не «обрізали» бажану частотну характеристику системи керування.

Динамічні параметри виконавчих пристроїв багато в чому залежать від їхньої потужності – зі збільшенням потужності зазвичай зростають масогабаритні характеристики й моменти інерції. Крім того, динамічні характеристики залежать від конструктивних особливостей виконавчого пристрою.

Ці обставини створюють проектувальникові певні проблеми, для вирішення яких розробляються комплексні заходи. Так, наприклад, для задоволення високих вимог до динаміки в системах керування приводами подач верстатів із ЧПУ необхідно прагнути до мінімальної довжини кінематичного ланцюга, виключати пружні елементи, застосовувати двигуни з найменшими масогабаритними характеристиками, високими показниками перевантажувальної здатності, мінімальним моментом інерції якоря, а також мінімальними значеннями постійних часу.

У процесі проектування слід застосовувати ті методики розрахунків, які прийняті, як стандартні. Так, наприклад, для розрахунків електроприводів верстатів із ЧПУ слід застосовувати стандартну методику,

наведену в [4].

Виконання силових розрахунків повинне здійснюватися з обов'язковою виставою ***розрахункової схеми***, на якій необхідно показати кінематичну схему привода, схему діючих навантажень, розмірні й кінематичні характеристики.

Динамічні властивості виконавчих пристроїв характеризуються постійними часу окремих ланок. Загальне значення постійної часу виконавчого пристрою можна одержати або шляхом розрахунків, використовуючи математичний опис виконавчого пристрою, або шляхом моделювання у програмнім середовищі MATLAB Simulink.

3 РОЗРОБКА -МОДЕЛЕЙ БІЗНЕС-ПРОЦЕСІВ

3.1 Розробка моделей IDEF0

Зазвичай спочатку будується модель існуючої організації роботи AS-IS (як є). На основі моделі AS-IS досягається консенсус між різними одиницями бізнесу по тому, "хто що зробив" і що кожна одиниця бізнесу додає в процес. Модель AS-IS дозволяє з'ясувати, "що ми робимо сьогодні" перед тем, як перестрибнути на те, "що ми будемо робити завтра". Аналіз функціональної моделі дозволяє зрозуміти, де перебувають найбільш слабкі місця, у чому будуть полягати переваги нових бізнес-процесів і наскільки глибоким змінам піддасться існуюча структура організації бізнесу.

Деталізація бізнес-процесів дозволяє виявити недоліки організації навіть там, де функціональність на перший погляд видасться очевидною. Ознаками неефективної діяльності можуть бути довгі пошуки необхідної інформації, некеровані роботи, роботи, які дублюються, неефективний документообіг (потрібний документ не виявляється в потрібному місці в потрібний час), відсутність зворотних зв'язків по керуванню (на проведення роботи впливає її результат), вхідна інформація використовуються нераціонально і т.д.

Знайдені в моделі AS-IS недоліки повинні бути виправлені при створенні моделі TO-BE (як буде) – моделі нової організації бізнес-процесів. Модель TO-BE потрібна для аналізу альтернативних (кращих) шляхів виконання роботи й документування того, як буде змінено бізнес-процес у майбутньому.

Слід указати на розповсюджену помилку при створенні моделі AS-IS - це створення ідеалізованої моделі. Прикладом може служити створення моделі на основі знань керівника, а не конкретного виконавця робіт. Керівник знає, як передбачається виконання роботи згідно посадовим інструкціям і часто не знає, як насправді підлеглі виконують рутинні роботи. У результаті виходить прикрашена модель, яка несе неправильну інформацію і яку неможливо надалі використовувати для аналізу. Така модель називається SHOULD_BE (як повинно б бути).

Технологія проектування інформаційної системи має на увазі спочатку створення моделі AS-1S, її аналіз і поліпшення бізнес-процесів, тобто створення наступної моделі TO-BE, і тільки на основі моделі TO-BE будується модель даних, прототип і потім остаточний варіант інформаційної системи.

Побудова системи лише на основі моделі AS-IS приводить до автоматизації підприємства за принципом "усе залишити як є, тільки щоб комп'ютери стояли", тобто інформаційна система автоматизує недосконалі бізнес-процеси й дублює, а не заміняє існуючий документообіг. У результаті

впровадження така система приводить лише до додаткових витрат на закупівлю встаткування, створення програмного забезпечення й супровід того й іншого.

Іноді поточна AS-IS і майбутня TO-BE моделі різняться дуже сильно, так що перехід від початкового до кінцевого стану стає неочевидним. У цьому випадку необхідна третя модель, що описує процес переходу від початкового до кінцевого стану системи, оскільки такий перехід – це теж бізнес-процес.

На рисунку 3.1 показаний приклад застосування технології IDEF0 для моделювання процесу приймання замовлень у клієнтів Call-центру. Принциповою особливістю даного процесу є одинакова кількість входів та виходів процесу (скільки замовлень поступило в систему, стільки ж замовлень повинно бути оброблено).

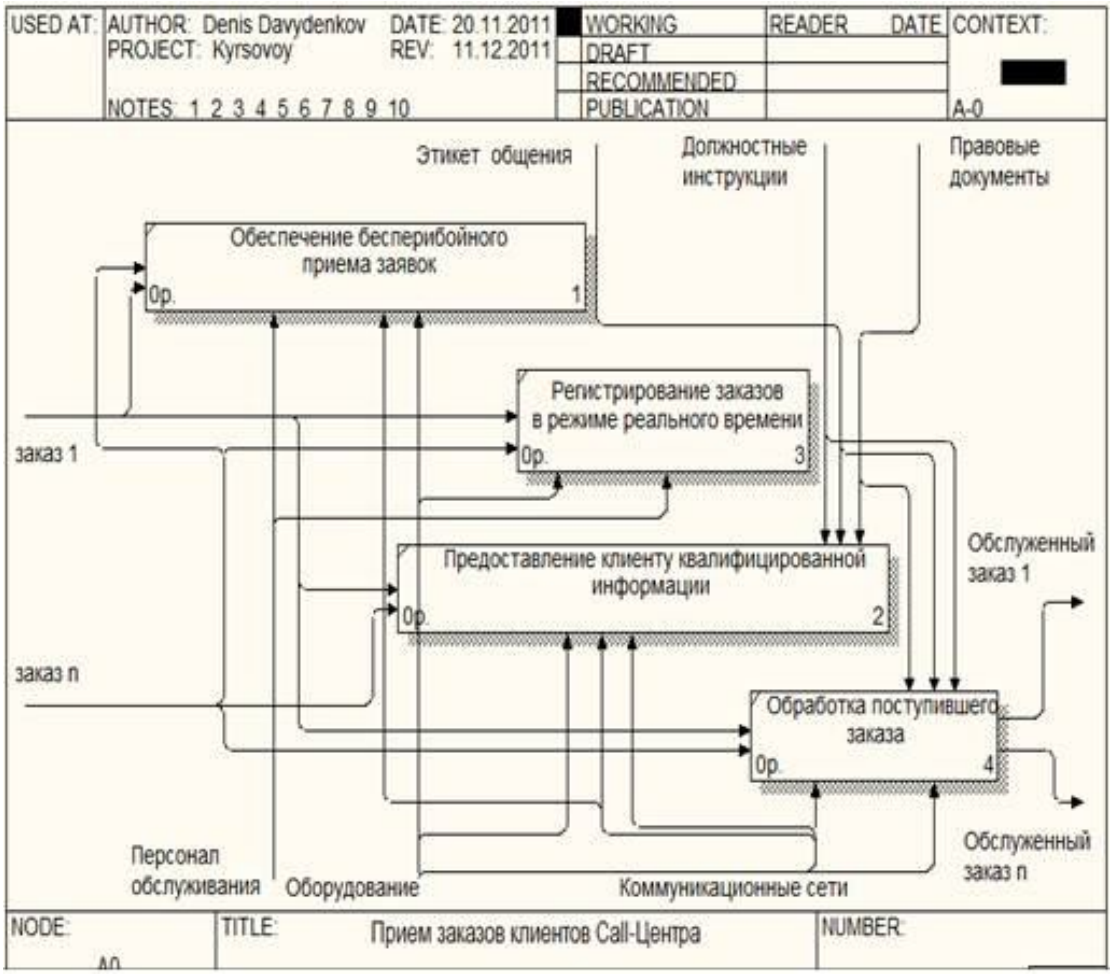

Рисунок 3.1 – Діаграма IDEF0 для моделювання процесу обслуговування клієнтів у Call-центрі

Для повноцінного функціонування Call-центру необхідно забезпечити надійне виконання кожного функціонального блоку.

3.2 Розробка інфологічної моделі предметної області

Ціль інфологічного моделювання полягає у забезпеченні найбільш природних для людини способів відбору й вистави тієї інформації, яку передбачається зберігати в базі даних. Основними конструктивними елементами інфологічних моделей є *сутності*, *зв'язки* між ними й *властивості* (*атрибути*) сутностей.

Сутність – це будь-який помітний об'єкт, тобто об'єкт, який можна відрізнити від іншого об'єкта по його *атрибутах*. Якщо об'єкти мають однакові атрибути, то вони не різняться по типу й належать одній сутності. Таким чином, атрибути необхідні, тому що вони є пойменованими характеристиками сутностей.

Зв'язок – це асоціювання двох або більш сутностей. Зв'язок дозволяє знайти в базі даних одні сутності за відомими значенням інших, а в програмі зв‘язки визначають припустимі взаємодії між об'єктами. При проектуванні баз даних створення зв'язків є найбільш складним завданням.

При побудові інфологічної моделі бази даних можна використовувати мову ER-діаграм (від англ. Entity-Relationship – сутність-зв'язок), універсальну мову моделювання UML, а також мову інфологічного моделювання. Приклад концептуальної ER-діаграми бази даних, яка *не враховує* особливості конкретної СУБД, наведено на рисунку 3.2.

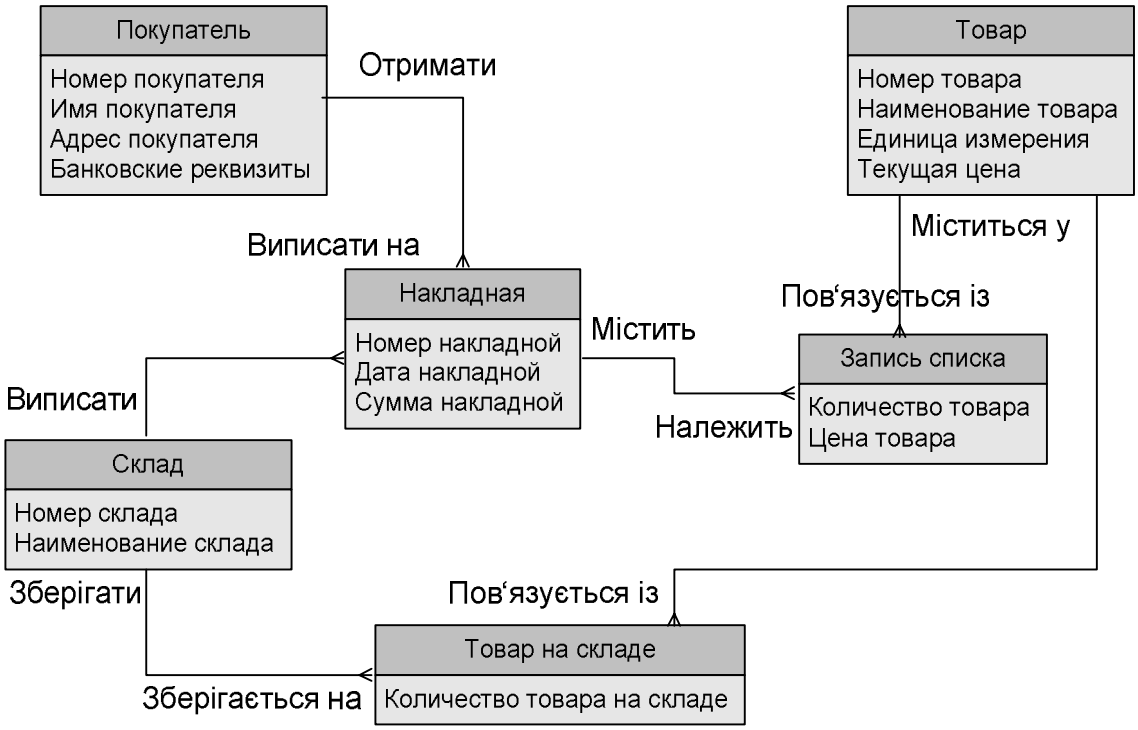

Рисунок 3.2 – Концептуальна ER-діаграма бази даних

У процесі інфологічного моделювання бази даних необхідно не тільки скласти перелік сутностей предметної області, їх атрибутів і зв'язків, але й зібрати інформацію про користувачів і особливості існуючих додатків, що використовують базу даних. Це потрібно зробити для того, щоб урахувати зовсім протилежні вимоги до бази даних з боку прикладних програмістів, що створюють бази даних, вимоги користувачів, що роблять запит на дані, а також вимоги адміністраторів баз даних, що опікуються про виключення можливих викривлень даних при введенні нової інформації або видаленні існуючої.

Слід ураховувати, що погано продуману структуру бази даних неможливо буде виправити ніякими додатками.

Особливості моделювання предметної області

Модель предметної області повинна являти собою словник термінів (об'єктів і класів), необхідних для моделювання програмного додатка.

В основу моделі повинні бути покладені діаграми класів. Клас – це місце для розміщення атрибутів (даних про об'єкти) і операцій (функцій, виконуваних об'єктами). Однак при моделюванні предметної області *у визначенні атрибутів і операцій немає необхідності* – ці процедури слід виконати пізніше. Насамперед необхідно сконцентрувати увагу на виявленні об'єктів і відносин між ними.

Побудова моделі предметної області додатка починається з ***виявлення абстракцій, таких сутностей***, які існують у реальному світі і являють собою основні концептуальні об'єкти системи.

При цьому програмне забезпечення повинне бути структуроване так, щоб у центрі виявилися об'єкти із простору задачі.

Однією з головних цілей розробки програми на основі абстракцій реального світу є створення можливості повторного використання того, або іншого програмного модуля. Ця можливість створюється саме на етапі, коли формуються класи, тобто абстракції реального світу. Від вибору класів, тобто від якості абстрагування, залежить також, якою буде спостережувана поведінка об'єкта. Реалізація цієї поведінки забезпечується внутрішнім устроєм (проведеною інкапсуляцією), однак цей устрій не повинен містити надмірностей, які ускладнюють поведінку, або не повинен бути занадто простим, що обмежує поведінку.

При моделюванні предметної області слід рухатися *зсередини назовні*. Це означає, що спочатку в системі виділяються ключові об'єкти, а потім, у процесі вивчення з якими ще об'єктами вони взаємодіють, до ключових об'єктів додаються інші з убутним ступенем значимості, поки не буде створена повна картина взаємодії об'єктів і не буде розроблений інтерфейс, що забезпечує керування цією взаємодією.

Необхідно врахувати, що при виконанні наступного етапу, тобто при виявленні прецедентів і описі поведінки системи, слід застосовувати рух ззовні усередину, поки не буде зрозуміло, за допомогою чого реалізується кожний елемент необхідної поведінки.

Отже, моделювання предметної області – це погляд на систему зсередини назовні й перше, що потрібно зробити при побудові статичної моделі системи, це знайти класи, які адекватно відбивають абстракції предметної області.

Кращими джерелами для виявлення класів є:

- високорівневий опис завдання;
- низькорівневі вимоги до системи.

Практично слід використовувати опис завдання й формулювання вимог, зроблені в першому розділі проекту, і в цих описах і формулюваннях обвести або підкреслити все іменники й іменні групи. Таким способом можна знайти майже всі важливі доменні об'єкти (класи).

У міру уточнення цього переліку повинне відбуватися наступне:

- іменники й іменні групи можуть стати об'єктами або атрибутами;
- дієслова й дієслівні групи можуть стати операціями й асоціаціями;
- родовий відмінок показує, що іменник повинен бути атрибутом, а не об'єктом.

Далі з отриманого списку класів треба вилучити непотрібні (надлишкові або несуттєві) і непридатні (занадто розпливчасті) елементи.

При побудові діаграм класів можна також розглянути можливі узагальнення (відносини виду «є», коли окремі об'єкти можна об'єднати одним іменем). Етап моделювання предметної області – це саме той момент, коли слід також прийняти рішення про агрегацію класів.

Так, наприклад, клас «Привод» містить у собі агреговані класи (об'єкти) – перетворювач, двигун і датчики, що здійснюють задану поведінку привода.

У підсумку модель предметної області, доповнена асоціаціями (статичними відносинами між парами класів), повинна адекватно описувати статичні аспекти задачі, які не залежать від часу, і в цьому нагадувати діаграму сутність-зв'язок, яка застосовується в моделюванні бази даних.

Приклад моделі предметної області для програмного забезпечення банкомата наведений на рисунку 3.3. У моделі показані класи програмного забезпечення і їх взаємозв'язки.

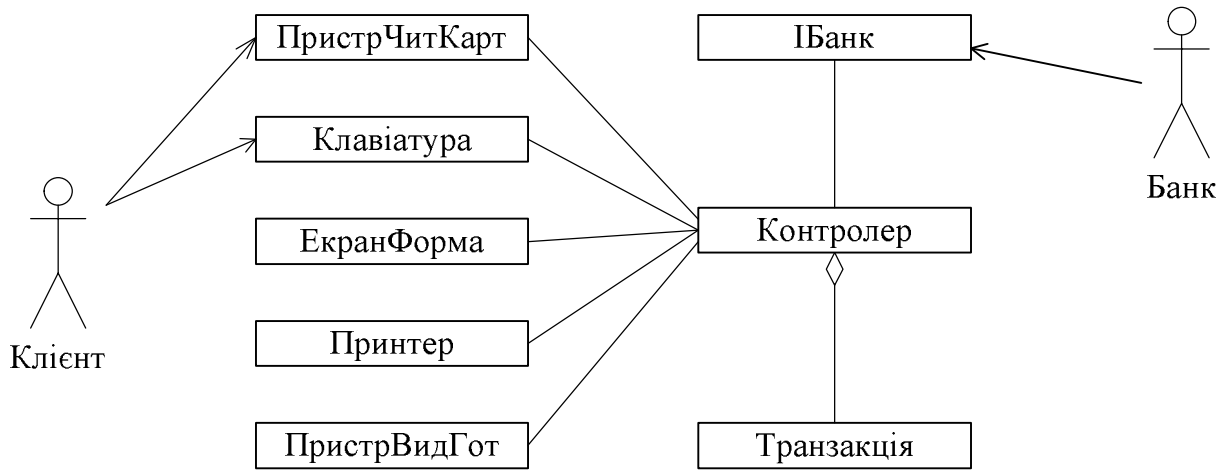

Рисунок 3.3 – Модель предметної області "Банкомат"

Для розміщення програми використовується ***абстрактний*** клас «Контролер», який не має чіткої поведінки, а тому не описується атрибутами й операціями. Чітка поведінка реалізується класом «Транзакція», що забезпечує з'єднання контролера з банком за допомогою класу-інтерфейсу «ІБанк».

Клієнт взаємодіє із програмним забезпеченням банкомата за допомогою пристрою читання пластикової карти, який обслуговується класом «ПристрЧитКарт», а також за допомогою введення інформації із клавіатури, яка обслуговується класом «Клавіатура».

Сервісні функції банкомата реалізують класи «ЕкранФорма» (діалог із клієнтом), «Принтер» (печатка квитанції) і «ПристрВидГот» (видача готівки).

Розповсюджені помилки моделювання предметної області

При моделюванні предметної області студенти допускають наступні помилки:

1. Занадто велику увагу приділяють аналізу всіх іменників і дієслів, упускаючи головні сутності (класи).

При намаганні скласти великий словник є ризик спуститися на низький рівень абстракції, що призведе до надмірного ускладнення програми.

2. Включають у класи операції, не вивчивши деталі прецедентів.

Не слід приділяти занадто багато уваги визначенню операцій на етапі моделювання предметної області. У цей момент ще мало інформації для прийняття обґрунтованих розв'язків про деталі поведінки.

3. При аналізі асоціації виду «є частиною» зазнають труднощів, що використовувати - агрегацію чи композицію.

На стадії моделювання предметної області краще говорити просто про агрегацію. Більш точний вибір слід відкласти на етап детального

проектування.

5. *Студенти дають класам малозрозумілі імена, наприклад,* cportmgr.Intf *замість* Portfoliomanager (фахівець, відповідальний за аналіз інвестицій). Чим очевидніше імена класів, тим простіше взаєморозуміння учасників процесу розробки програми. Про акроніми й інші види скорочень можна подумати на етапі реалізації.

3.3 Розробка специфікації функцій системи

Програмні додатки різняться між собою набором функцій, які вони реалізують. Набір функцій визначається конкретним завданням автоматизації.

Однак конкретність завдання на початковому етапі проектування зазвичай відсутня. Тому для формування переліку функцій програмного додатка доводиться застосовувати моделювання.

Візуальне моделювання можна представити як деякий процес порівневого спуску від найбільш загальних вистав про завдання автоматизації до уточнених процедур і операцій програмної системи. Для цього спочатку будується так звана діаграма варіантів використання (use case diagram), яка описує функціональне призначення системи або, інакше кажучи, те, що система буде робити в процесі свого функціонування.

Вершинами в діаграмі використання є *актори* й *елементи Use Case*. Актори представляють зовнішній світ, що потребує результатів роботи системи, а елементи Use Case представляють дії, виконувані системою в інтересах акторів. Один актор може використовувати кілька елементів Use Case, і навпаки, кожний елемент може бути використаним декількома акторами.

Між актором і елементом Use Case може бути ***тільки один вид відносини*** – ***асоціація***, яка відображає їхню взаємодію. Асоціація може бути позначена іменем, ролями та потужністю.

Між елементами Use Case можуть існувати два види відносин – ***розширення*** (extend) і ***включення*** (include). Ці відносини показано на рисунку 3.4.

Відношення розширення (*extend*) між елементами Use Case означає, що поведінка базового елемента *може бути розширена* поведінкою іншого елемента Use Case.

Відношення включення (*include*) між двома елементами відповідає делегації обов'язків. При цьому елемент, що включається, *реалізує обов'язкову* поведінку.

Елемент Use Case описує, *що повинна робити система*, але не

визначає, *як* вона повинна це робити. Це правило дозволяє відокремлювати інтерфейс від способу реалізації системи.

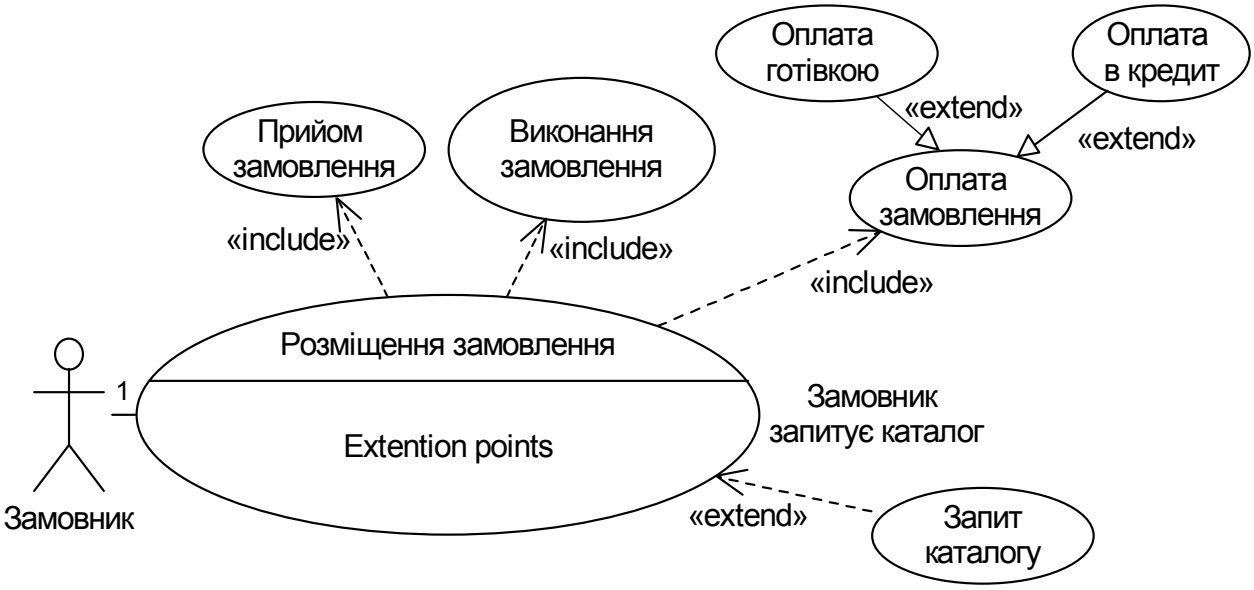

Рисунок 3.4 – Діаграма варіанта використання «Розміщення замовлення» програмної системи інтернет-магазину

Поведінка елемента Use Case описується потоком подій, у якому виділяють:

- основний потік і альтернативні потоки поведінки;
- умови для запуску й зупинки елемента Use Case;
- умови взаємодії елемента Use Case з акторами;
- дані, якими обмінюються актор і система.

Для специфікації прецедентів потрібно використовувати прототипи графічного інтерфейсу, за допомогою яких можна представити, що буде робити користувач і як на цю дію користувача буде реагувати система.

Текст прецеденту слід формулювати в ясній і короткій формі. Кожна пропозиція повинна мати структуру «іменник-дієслово» і повинні бути відразу видні актори й потенційні доменні об'єкти. У міру виявлення нових об'єктів або їх уточнення потрібно відразу обновляти модель предметної області. Важливо також уважно стежити за можливими альтернативними послідовностями дій (виключеннями) для кожного прецеденту.

Рекомендується наступна послідовність моделювання прецедентів:

1. *Створити шаблон опису прецеденту,* у якому треба передбачити розділи «Головна послідовність» і «Альтернативні послідовності». Інші розділи не потрібні, вони будуть тільки відволікати увагу.

2. *Задати собі питання: «Що повинне відбуватися?».* З відповіді на нього починається головна послідовність.

3. *Задати собі питання: «Що ще може відбуватися?».* Важливо врахувати все, що може зробити користувач і зафіксувати всі альтернативні варіанти, тобто виключення. Якщо буде упущений якийсь варіант, ситуація стане невизначеною й система зависне.

Типові помилки при моделюванні прецедентів

1. Замість словесного сценарію використання студенти пишуть функціональні вимоги.

Слід чітко відрізняти опис порядку використання від вимог до системи. Вимоги зазвичай формулюються щодо того, ***що повинна робити система***. Сценарій же описує дії, які здійснюються користувачем і системою, при цьому дії системи є її *реакцією* на дії користувача.

2. Описуються атрибути й методи, а не порядок використання.

Тексти прецедентів не повинні зайво докладно описувати деталі вистави. Слід також утриматися від посилань на поля в екранних формах. У тексті прецеденту не слід ні йменувати, не описувати методи (операції), тому що вони говорять про те, ***як система робить щось***.

3. Прецеденти записуються занадто лаконічно.

При написанні текстів прецедентів, багатослівність більш переважна, чим лаконічність. Переходячи до наступного етапу моделювання (моделювання поведінки), добре б мати частину деталей з опису прецедентів.

4. Прецеденти повністю абстрагуються від інтерфейсу користувача.

Один з фундаментальних принципів, заснований на прецедентах, полягає в тому, що при проектуванні системи розроблювачі повинні враховувати дії, які користувачі виконують за допомогою екрана й клавіатури, тобто пам'ятати про інтерфейс.

5. Не привласнюються явні імена граничним об'єктам.

Граничними називаються об'єкти, з якими взаємодіють актори. До них належать екранні форми, діалогові вікна й меню. Граничні об'єкти слід іменувати в текстах прецедентів явно.

6. Прецеденти описуються не з погляду користувача.

З погляду користувача прецедент найбільше ефективно формулюється із застосуванням *дієслів теперішнього часу в дійсній заставі*, наприклад, «*Користувач уводить пароль*». Прецеденти повинні описувати дії користувача й реакцію на них системи, а такого роду текст найкраще звучить у дійсній заставі.

7. Описуються тільки дії користувача, а реакція системи ігнорується.

Текст прецеденту повинен відображати як подію, так і відгук на цю

подію, наприклад: «Коли користувач робить те-те, система робить те-те». Прецедент повинен описувати те, що відбувається «під капотом» у відповідь на дії користувача (створення нових об'єктів, контроль даних, які вводяться, або вивід повідомлень про помилки). У тексті прецеденту повинні розглядатися обидві сторони діалогу користувача із системою. Ігнорування опису реакції системи рівносильне ігноруванню поведінки програми.

8. Опускаються описи альтернативних послідовностей дій.

Головні потоки зазвичай описувати простіше. Але це не означає, що роботу з альтернативними потоками можна відкласти до етапу детального проектування. Досвід показує, що, коли залишаються нерозкритими важливі альтернативні послідовності, виникають проблеми з доведенням проекту, тому що непередбачувані дії користувачів приводять до «зависання» додатка.

Переходити до подальших етапів процесу розробки треба після того, як досягнуті наступні цілі моделювання прецедентів:

- прецеденти описують усю необхідну функціональність системи;
- для кожного прецеденту чітко й коротко описана головна послідовність дій, а також усі альтернативні послідовності;
- виділені сценарії, загальні для декількох прецедентів.

Завдяки моделі прецедентів, розроблювач додатка може одержати уявлення про динамічний аспект об'єктної моделі.

3.4 Моделювання програмного додатка

Під вираженням «моделювання програмного додатка» розуміють розробку або створення *схеми перетворення специфікації функцій у готовий додаток*.

Для того, щоб моделювати окремі потоки керування в складі прецеденту, застосовуються діаграми взаємодій (interaction diagrams).

Діаграми взаємодій використовуються *для моделювання динамічних аспектів системи.* Вони важливі також для здійснення прямого й зворотного проектування, тобто по моделі згенерувати програмний код і, навпаки, – по програмному коду одержати модель.

До діаграм взаємодії ставляться два типи:

1. ***Діаграма послідовностей*** (Sequence diagram), що акцентує увагу на тимчасовій упорядкованості повідомлень.
2. ***Діаграма кооперації*** (Collaboration diagram), основна увага в якій приділяється структурної організації об'єктів.

Для створення ***діаграми послідовності*** необхідно, насамперед, розташувати об'єкти, що беруть участь у потоці керування, у верхній її частині уздовж горизонтальної осі. Зазвичай ініціюючий керування об'єкт

розміщають ліворуч, а інші – праворуч, за принципом підпорядкованості об'єктів.

Від кожного об'єкта вниз проводиться лінія його життя, на якій розміщаються маленькі прямокутники – фокуси керування. Фокус керування призначений для відображення активного стану об'єкта. Потім на вертикальній осі часу розміщаються горизонтальні лінії повідомлень, які об'єкти посилають і ухвалюють. Це дозволяє одержати наочну картину розвитку потоку керування в часі (рис. 3.5).

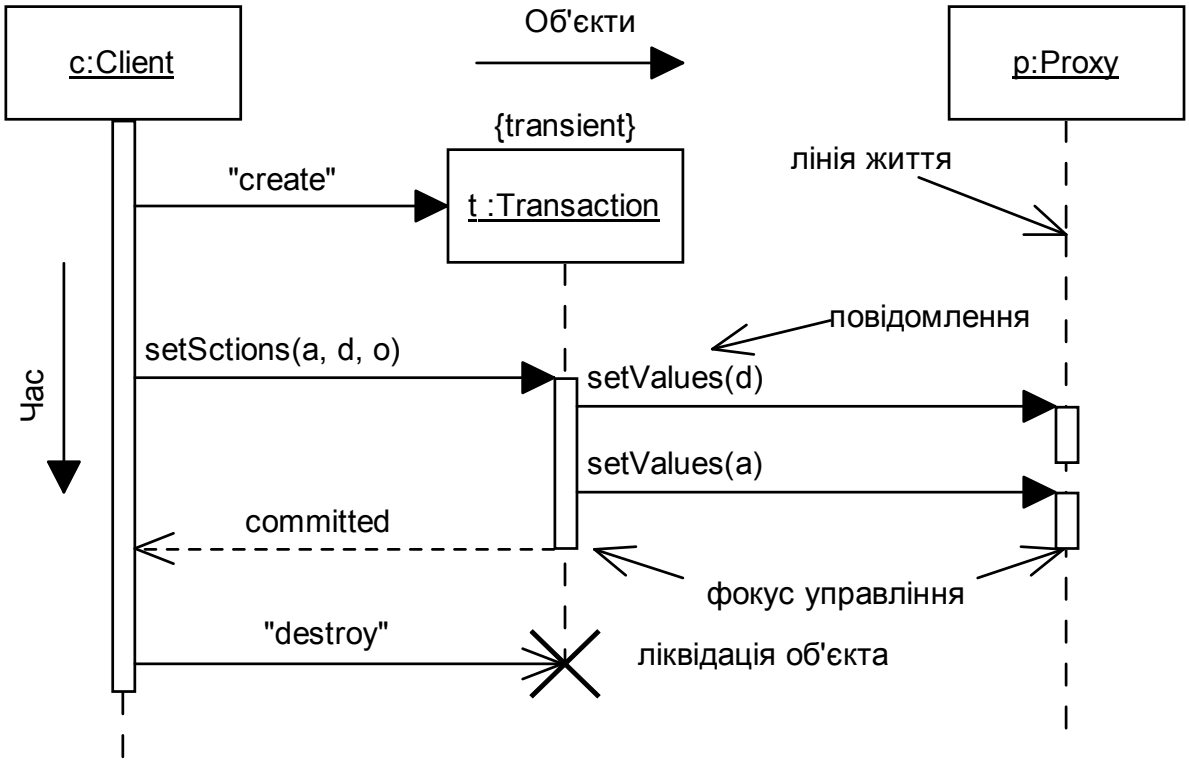

Рисунок 3.5 – Приклад оформлення діаграми послідовностей

UML дозволяє моделювати дії декількох видів:

- ***call*** (викликати) – викликає операцію, застосовувану до об'єкта;
- ***return*** (повернути) – повертає значення об'єкту;
- ***send*** (послати) – посилає об'єкту сигнал;
- ***create*** (створити) – створює новий об'єкт;
- ***destroy*** (знищити) – видаляє об'єкт.

Створюючи діаграми послідовності, слід ураховувати, що об'єктам призначаються певні зобов'язання.

Для первісного виявлення деяких об'єктів слід вивчити іменники в потоці подій (керування). Іменники допоможуть визначити, що може бути об'єктом. Якщо не зрозуміло, що описує іменник – об'єкт або атрибут, слід визначити, чи є в нього поведінка. Якщо поведінка відсутня, то це, імовірно, атрибут. Якщо для іменника поведінка визначається, то, швидше за все, це об'єкт.

Окремі об'єкти, виконавши свою роль у системі, можуть бути знищені, щоб звільнити займані ними ресурси. Для таких об'єктів лінія життя обривається в момент їх знищення. Для позначення моменту знищення об'єкта в мові UML використовується спеціальний символ у формі латинської букви "X" (рис. 3.5). Нижче цього символу пунктирна лінія не зображується, оскільки відповідного об'єкта в системі вже немає і цей об'єкт повинен бути виключений із усіх наступних взаємодій.

Зовсім не обов'язково створювати всі об'єкти в початковий момент часу. Окремі об'єкти в системі можуть створюватися в міру необхідності, суттєво заощаджуючи ресурси системи й підвищуючи її продуктивність. У цьому випадку прямокутник такого об'єкта зображується не у верхній частині діаграми послідовності, а в тій її частині, яка відповідає моменту створення об'єкта, наприклад, об'єкт **t** класу Transaction на рисунку 3.5. При цьому прямокутник об'єкта розташовується в тому місці діаграми, яке по осі часу збігається з моментом його виникнення в системі. Очевидно, що об'єкт обов'язково створюється зі своєю лінією життя й, можливо, з фокусом керування.

Фокус керування зображується у формі витягнутого вузького прямокутника, верхня сторона якого позначає початок активності, а її нижня сторона – закінчення активності. Цей прямокутник може заміняти лінію життя об'єкта, якщо на всім її протязі він є активним.

Періоди активності об'єкта можуть чергуватися з періодами його пасивності або очікування. У цьому випадку в такого об'єкта є кілька фокусів керування. Важливо розуміти, що одержати фокус керування може тільки існуючий об'єкт, у якого в цей момент є лінія життя. Якщо ж деякий об'єкт був знищений, то знову виникнути в системі він уже не може. Замість нього може бути створений інший екземпляр цього ж класу, який, строго говорячи, буде іншим об'єктом.

Ініціатором потоку керування може бути актор (зовнішній користувач), який зображується у вигляді дротового чоловічка. Актор розміщається на діаграмі послідовності найпершим об'єктом ліворуч зі своїм фокусом керування. При цьому сам актор може мати власне ім'я або залишатися анонімним.

Взаємодія об'єктів представляється повідомленнями, які є специфікацією передачі інформації й показують, що один об'єкт викликає функцію іншого. Далі, коли будуть визначені операції й методи класів, кожне повідомлення стане операцією або викликом методу.

Приклад побудови діаграми послідовності для потоку керування, який ставиться до моделювання процесу керування регуляторами струму й швидкості електропривода від керуючої програми системи ЧПУ, наведений на рисунку 3.6.

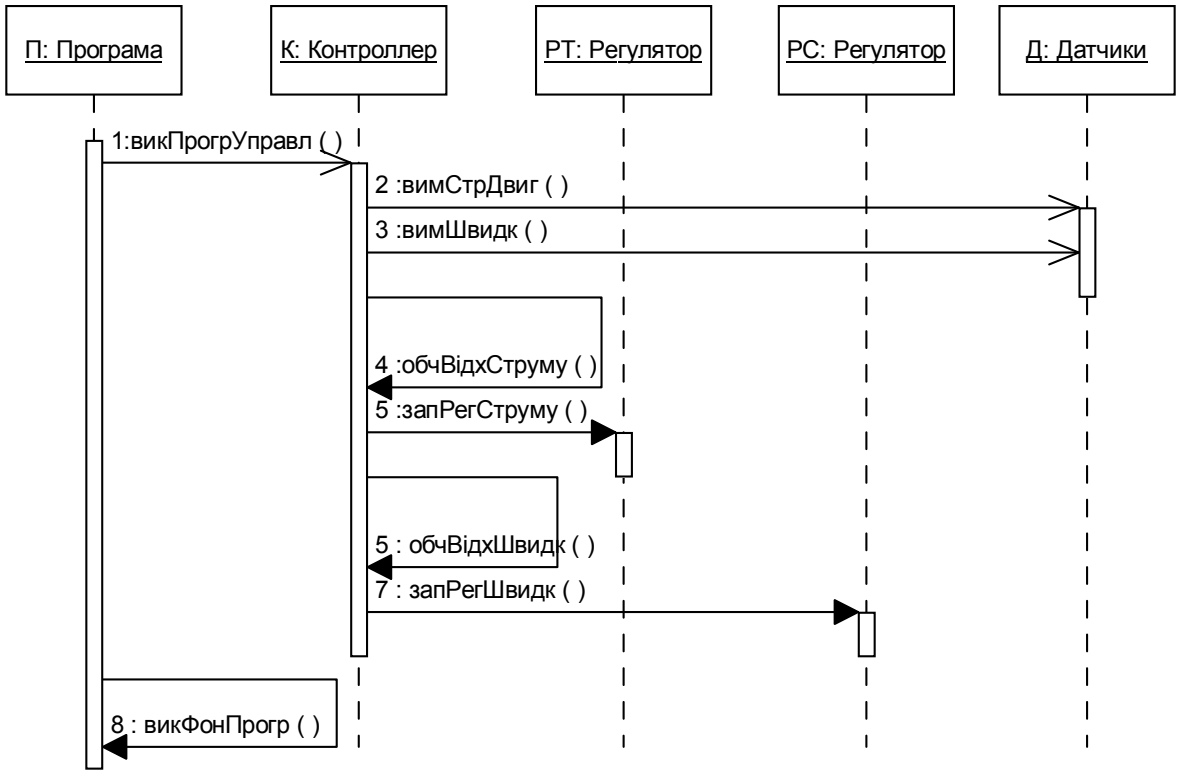

Рисунок 3.6 – Діаграма послідовності системи керування електроприводом

На цій діаграмі показана взаємодія об'єктів, що становлять предметні сутності процесу керування – програма ЧПУ (П), контролер (К), регулятор струму (РТ), регулятор швидкості (РС) і датчики (Д), що включають у себе датчики струму й швидкості.

Послідовність виконання операцій позначена порядковими номерами:

1 Керуюча програма П посилає контролеру К просте повідомлення *викПрогрУправл()* – виконати програму управління електроприводом.

2 Контролер К одержує фокус керування й направляє повідомлення *вимСтрДвиг()* – виміряти поточне значення струму двигуна.

3 Датчик струму повертає конкретне значення струму двигуна, після чого контролер К направляє об'єкту Д повідомлення *вимШвидк()* – виміряти поточне значення швидкості двигуна.

4 Датчик швидкості повертає контролеру конкретне значення швидкості й котроллер переходить на виконання рефлексивного повідомлення *обчВідхилСтруму()* – обчислити відхилення струму.

5 Відхилення струму від заданого значення записується в регістр регулятора струму.

6 Контролер виконує рефлексивне повідомлення *обчВідхШвидк()* – обчислює відхилення швидкості від заданого значення.

7 Відхилення швидкості записується в регістр регулятора швидкості.

8 Керуюча програма викликає диспетчера фонового режиму для обслуговування пульта оператора.

Після розробки діаграм послідовності, що моделюють динамічні процеси, слід перейти до розробки діаграм класів, що визначають статичний стан програмної системи. Логіка такої послідовності моделювання полягає в тому, що діаграми послідовності дозволяють уточнити об'єкти і їх поведінку. Об'єкти з однаковою поведінкою можна співвіднести із одним класом і в такий спосіб одержати уточнений перелік класів і уточнену картину їх поведінки.

Побудова діаграм класів – це найважливіший і трудомісткий етап у створенні моделі програми. Кожний клас має набір ***методів*** (operations) і ***змінних*** (attributes).

Зазвичай для опису системи створюють кілька діаграм класів. На одних показують деякі підмножини класів і відносини між класами підмножини. На інших відображають ті самі підмножини, але разом з атрибутами й операціями класів. Треті відповідають тільки пакетам класів і відносинам між ними. Для зображення повної картини системи можна розробити стільки діаграм класів, скільки потрібно.

При наявності великої кількості класів вони можуть поєднуватися в пакети (packages). Поєднувати класи можна як завгодно, однак існує декілька *найпоширеніших підходів*.

По-перше, можна групувати класи за стереотипом. У такому випадку виходить один пакет із класами-сутностями, один із прикордонними класами, один з керуючими класами і т.д. Цей підхід може бути корисним з погляду розміщення готової системи, оскільки всі прикордонні класи, які перебувають на клієнтських машинах, уже виявляються в одному пакеті.

Другий підхід полягає в об'єднанні класів по функціональності. Наприклад, у пакеті *Security* (безпека) будуть утримуватися всі класи, які відповідають за безпеку додатка.

Перевага цього методу полягає в можливості повторного використання пакетів. Якщо уважно підійти до групування класів, можна одержати практично незалежні один від іншого пакети.

Третій підхід – це комбінація двох зазначених підходів. На високому рівні можна згрупувати класи по функціональності, наприклад, створити пакет *Security*, відповідальний за безпеку, а на нижньому рівні згрупувати класи, які відповідальні за безпеку, по стереотипах.

Діаграми класів є гарним інструментом проектування. З їхньою допомогою розроблювачі можуть бачити й планувати структуру програмної системи ще до фактичного написання коду, завдяки чому на самому початку можна зрозуміти, чи добре спроектована система.

Повний опис класу включає ім'я класу, стереотип, список атрибутів і операцій (методів). Усе компоненти повинні мати специфікацію, у якій фіксуються всі властивості й коментарі.

Завдяки тому, що атрибути втримуються усередині класу, вони сховані від інших класів. Але деякі класи мають право читати й змінювати атрибути. Така можливість може бути задана ***видимістю атрибута*** (attribute visibility).

Найчастіше використовуються наступні значення видимості:

- ***public*** **(**загальний, відкритий**)**. Атрибут видно всім іншим класам. Будь-який клас може переглянути або змінити значення атрибута. У нотації UML загальному атрибуту відповідає знак "+";
- ***private*** **(закритий**, секретний**)**. Атрибут невидимий ніяким іншим класам. У нотації UML закритий атрибут позначається знаком "–";
- ***protected*** **(захищений)**. Атрибут доступний тільки самому класу і його спадкоємцям. Нотація UML для захищеного атрибута має знак "#".

У загальному випадку атрибути рекомендується робити закритими або захищеними. Це дозволяє краще контролювати сам атрибут і код, а також уникати ситуації, коли значення атрибута змінюється всіма класами.

У деяких випадках у класи включаються атрибути або методи, які ставляться до всього класу в цілому, а не до окремого екземпляра (об'єкту) цього класу. Для вказівки цього атрибут треба зробити статичним (static).

Ім'я операції визначається її типом. Зазвичай застосовуються чотири типи операцій.

Операції реалізації (implementor operations) реалізують деяку бізнес-функціональність.

Операції доступу (access operations). Атрибути зазвичай бувають закритими або захищеними. Проте, інші класи іноді повинні переглядати або змінювати їхнє значення. Такий підхід дає можливість безпечно інкапсулювати атрибути усередині класу, захистивши їх від інших класів, але все-таки дозволяє здійснювати контрольований доступ. Створення операцій *Get* і *Set* (одержання й зміни значення для кожного атрибута класу) є промисловим стандартом.

Допоміжними операціями (helper operations) називаються такі операції класу, які необхідні йому для виконання його відповідальностей, але про які інші класи не повинні нічого знати. Це закриті й захищені операції класу.

Для ідентифікації операцій необхідно виконати такі дії:

1 Вивчити повідомлення на діаграмах послідовності. Більша частина повідомлень на цих діаграмах являє собою операції реалізації. Рефлексивні повідомлення будуть допоміжними операціями.

2 Розглянути керуючі операції. Можливо, потрібно додати

конструктори й деструктори.

3 Розглянути операції доступу. *Для кожного атрибуту класу*, з яким будуть працювати інші класи, необхідно створити операції *Get* і *Set*.

Однією з особливостей добре спроектованого додатка є порівняно невелика кількість зв'язків у системі. Клас, у якого багато зв'язків, повинен знати про велике число інших класів системи. У результаті його важко буде використовувати в інших додатках. Крім того, складно буде вносити зміни в готовий програмний продукт. Внесення змін у клас може вплинути на поведінку багатьох класів.

На рисунку 3.7 наведений приклад діаграми класів системи керування приводом подачі металорізального верстата. На діаграмі представлений активний клас *УправлПрограма*, який має атрибут (властивість) *траекторіяОбробДеталі* та операції: *викПрограму()*, *аналізСтану(), прогнозЗакінчУправл()*.

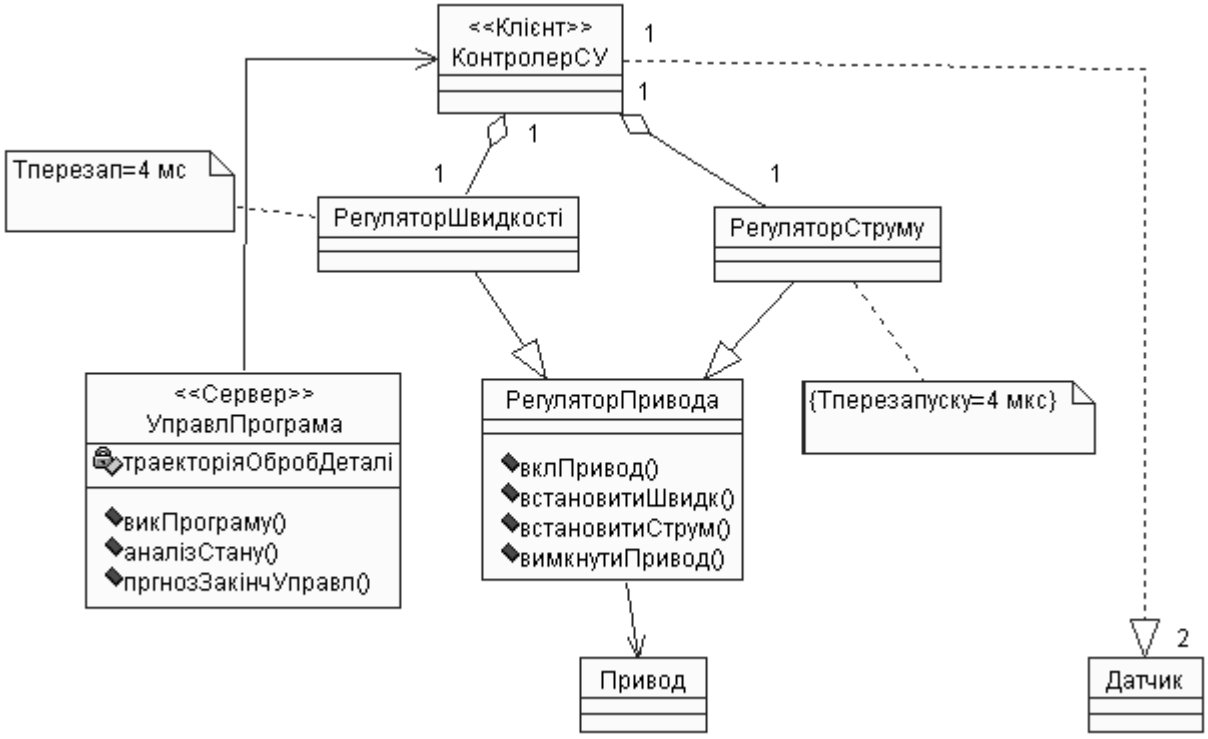

Рисунок 3.7 – Діаграма класів системи керування приводом

Між цим класом і класом *КонтролерСУ* є асоціація у вигляді клієнт-серверного відношення – екземпляри програми задають параметри руху, які повинні забезпечувати екземпляри контролера.

Клас *КонтролерСУ* включає по одному екземпляру класів *РегуляторШвидкості* та *РегуляторСтруму* та має зв‘язок реалізації із двома екземплярами класу *Датчик*.

Екземпляри *РегуляторШвидкості* і *РегуляторСтруму* включені в *КонтролерСУ* фізично – тут присутнє відношення *композиція*. Екземпляри

Датчик включені по посиланню, тобто екземпляр *КонтролерСУ* має лише покажчики (адреси або регістри) на об'єкти-датчики.

Одночасно *РегуляторШвидкості* і *РегуляторСтруму* – це підкласи абстрактного суперкласу *РегуляторПривода*, який передає їм у спадщину абстрактні операції: *вклПривод(), вимкнутиПривод(), встановитиШвидкість()* та *встановитиСтрум()*. У свою чергу, клас *РегуляторПривода* використовує конкретний клас *Привод*.

Для класу *РегуляторСтруму* задане обмеження часу на повторне включення 4 мкс, для класу *РегуляторШвидкості* – 4 мс.

На діаграмі класів об'єкти не існують ізольовано друг від друга – вони або зазнають впливу, або самі впливають на інші об'єкти. При цьому поведінка об'єкта є функцією, як його стану, так і виконуваних їм операцій.

Операції позначають обслуговування, яке об'єкт пропонує своїм клієнтам. Можливі п'ять видів операцій *клієнта над об'єктом:*

1) *модифікатор* – по цій операції змінюється стан об'єкта, наприклад, вага;
2) *селектор* – об'єкт дозволяє доступ до стану, але не змінює його, наприклад, надає інформацію;
3) *итератор* – зміст об'єкта доступний в певному порядку, наприклад, по номенклатурі;
4) *конструктор* – операцією створюється об'єкт і ініціалізується його стан, наприклад, включається об'єкт, який забезпечує заміну інструмента;
5) *деструктор* – операція руйнування об'єкта й звільнення пам'яті.

У наведеному на рисунку 3.7 прикладі операція *викПрограму()* є модифікатором, операція *аналізСтану()* – ітератором, а операція *прогнозЗакінчУправл()* – селектором.

4 ПРОЕКТУВАННЯ СИСТЕМИ УПРАВЛІННЯ ОБЛАДНАННЯМ

У цьому розділі слід розробити «каркас» проекту, тобто розв'язати основні завдання проектування системи. Перелік цих завдань зводиться до наступного:

1 Розробка структурної схеми системи керування.

2 Вибір засобів та конфігурування системи керування.

3 Розробка схем з'єднань і підключень.

Предметом проектування є цифрова система керування, яка містить у собі наступні компоненти:

- засоби програмного керування;
- засоби інтерфейсу;
- інформаційні пристрої;
- виконавчі пристрої;
- перетворювальні пристрої;
- апаратура з'єднання;
- щити й пульти керування;
- алгоритмічне й програмне забезпечення.

Розробка проектної документації проводиться в наступній послідовності.

На першому етапі об'єкт керування треба декомпозувати на окремі вузли й механізми так, щоб можна було визначити:

1) де розташовані зони концентрації інформаційних, перетворювальних і виконавчих пристроїв і які відстані між цими зонами;

2) де повинні бути встановлені шафи й пульти керування;

3) які засоби необхідно застосувати для з'єднання територіально розподілених пристроїв;

4) яка інформація повинна відображатися на панелі оператора.

Для вирішення цих завдань корисно скласти план розміщення апаратних засобів системи.

На наступному етапі розробляється спрощена структурна схема системи.

Структурна схема – це графічне зображення структури системи. Під структурою розуміється сукупність частин, на які може бути розділена система за певною ознакою, а також шляхи передачі впливів між ними.

Створюючи структурну схему, слід відображати такі структурні

елементи (приводяться для орієнтування):

- Апаратура керування верхнього рівня.
- Програмувальний контролер.
- Панель оператора.
- Пульти ручного керування.
- Перетворювачі енергії.
- Основні датчики.
- Виконавчі пристрої.
- Шини й лінії зв'язку.

У теперішній час для контролю й керування широко застосовуються багатофункціональні агрегатні системи. Такі системи спрощують виконання монтажних робіт та поліпшують умови експлуатації систем управління.

Практично всі сучасні системи керування технологічного рівня забезпечують можливість інтеграції в системи верхніх рівнів керування за допомогою їх підключення до промислової мережі Industrial Ethernet.

При виборі апаратури керування важливо врахувати, яку інформаційну, апаратну й програмну підтримку забезпечує виробник цієї апаратури. Слід віддати перевагу тим виробникам, які вдосконалюють якість своєї продукції, надають широкий спектр послуг, застосовують гнучкі механізми знижок на ціну продукції, користуються довірою на ринку.

Приклад структурної схеми наведений на рисунку 4.1.

Слід урахувати, що засоби керування повинні мати можливість вибору автоматичного або ручного режиму роботи.

Автоматичний режим є основним (робочим) режимом. Ручний режим необхідний для проведення ремонтно-налагоджувальних робіт, випробовування після завершення налагоджувальних робіт, ліквідації аварійних і позаштатних ситуацій, а також для роботи без локальної системи керування у випадку її відмови. Відмінною рисою цього режиму є те, що всі сигнали керування формуються оператором за допомогою пульта й впливають на перетворювачі, які управляють виконавчими пристроями.

У зв'язку із цим при виборі перетворювача слід ураховувати, що для нормальної роботи виконавчого пристрою в режимі ручного керування в перетворювачі повинні бути інтегровані всі функції керування й захисту виконавчих пристроїв.

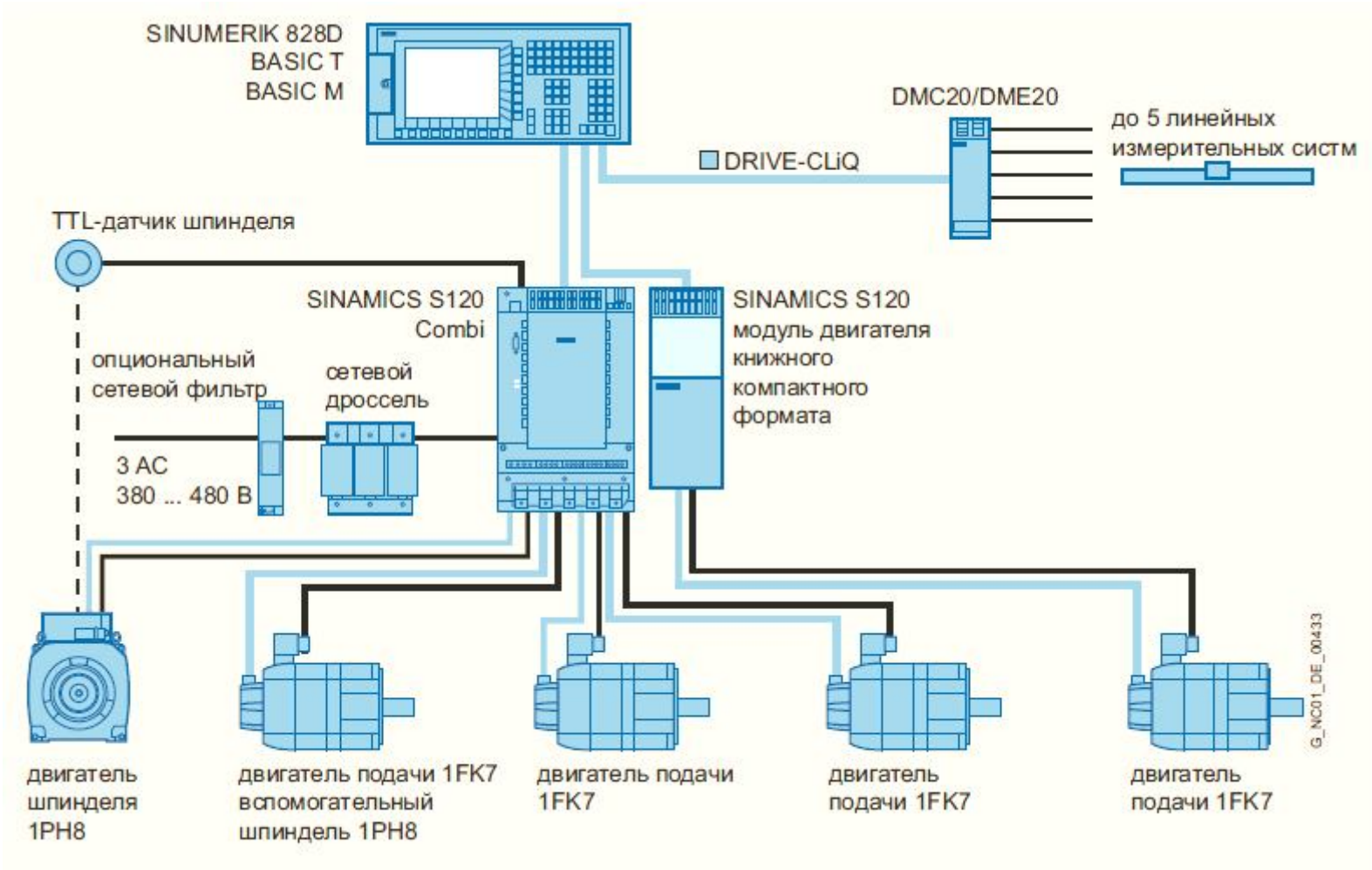

Рисунок 4.1 – Приклад структорної схеми приводної системи SINAMICS верстата із ЧПУ

При виборі перетворювача для керування електродвигуном необхідно переконатися, що він забезпечує наступні (основні) функції захисту:

- від короткого замикання;
- від перевантаження;
- від підвищення й зниження напруги в ланцюзі постійного струму;
- від обриву фази живлячої напруги;
- від перегріву радіаторів силових елементів;
- від перегріву двигуна;
- від замикань на землю силових ланцюгів.

Спрацьовування будь-якого вузла захисту повинне приводити до відключення електропривода. Інформація про спрацьовування захисту повинна передаватися на панель керування.

Усі периферійні засоби повинні бути розраховані на підключення до портів уведення-виводу і, у той же час, мати достатню гнучкість та підтримувати нормальне функціонування у випадку заміни програмувального контролера на іншій, більш досконалий, або побудований по іншій архітектурі.

Після вибору основних структурних елементів можна перейти до конфігурування системи керування.

Конфігурування системи автоматизації необхідно для того, щоб виключити помилки в створенні комунікаційного середовища. Для розв'язку завдання конфігурування застосовуються спеціальні механізми й програмні засоби, які забезпечують ретельний аналіз усіх аспектів функціонування майбутньої системи.

Якщо проектована система керування устаткуванням будується на базі апаратури фірми Siemens, то для її конфігурування необхідно застосовувати програмні додатки Configuring Networks (Netpro) і Hardware Configuration (HW Config) програмного середовища STEP 7.

Фірма Siemens розробила також інструмент швидкого проектування систем ЧПУ верстатів – програму NCD-Configurator. Ця програма призначена для конфігурування систем типу SINUMERIK.

При використанні апаратури інших виробників конфігурування можна виконати за допомогою програмної системи CoDeSys або SCADA-системи TRACE MODE.

Результати конфігурування представляються в проекті у вигляді файлу конфігурації, а також скріншотів у двох представленнях – конфігурація контролера і конфігурація розподіленої периферії системи (функціональна архітектура системи).

Після розробки структурної або функціональної схеми слід розробити

більш детальне представлення проекту – скласти принципіальні схеми, схеми з'єднань (монтажні) і схеми підключень (зовнішніх з'єднань).

Принципіальні схеми (електричні, гідравлічні й пневматичні) служать для визначення повного складу приладів, апаратів і пристроїв, а також зв'язків між ними. Принципіальні схеми є основою для розробки монтажних схем.

Схеми з'єднань (монтажні) призначені для виконання монтажу щитів і пультів, а також для з'єднання частин певної електроустановки. Прикладом такої схеми може служити схема з'єднань перетворювача з електродвигуном і пусковою апаратурою.

Схеми підключення (зовнішніх з'єднань) служать для графічного відображення з'єднань апаратури керування як між собою (для цього використовуються шини, кабелі й дроти), так і із зовнішніми пристроями – вхідними й вихідними. При цьому передбачається, що це електричне устаткування територіально «розкидане».

При виконанні схем підключення необхідно враховувати вимоги, наведені в ГОСТ 2.701-84 і ГОСТ 2.702-75 "Правила выполнения электрических схем".

Схеми *підключень* пристроїв уведення сигналів будуються в такий спосіб. У верхній частині аркушу розташовуються підключення до лінії напруги живлення (точки найвищого потенціалу). Нижче розташовуються пристрої, що підключаються до модуля уведення. У нижній частині аркушу розташовується умовне зображення модуля. Пристрої й клеми модуля забезпечуються сполучними лініями.

Схеми підключень пристроїв виводу сигналів керування або індикації будуються інакше. Модуль виводу, виходи якого мають найбільший потенціал, зображується у верхній частині аркуша, а зовнішні пристрої, що підключуються до нього і мають нижчий потенціал, – у нижній частині аркуша.

5 РЕКОМЕНДАЦІЇ З МОДЕЛЮВАННЯ СИСТЕМИ

У системах керування технологічного рівня зазвичай моделюють роботу виконавчих пристроїв.

При моделюванні привода постійного струму, для якого розроблений досить повний математичний опис, доцільно застосовувати функціональні моделі пакету MATLAB Simulink.

Для моделювання приводів змінного струму, що відрізняються громіздкими математичними перетвореннями й переходами від однієї системи координат до іншої, більш ефективне застосування віртуальних блоків, наявних у бібліотеках додатка SimPowerSystems пакета MATLAB/Simulink.

Віртуальні блоки дозволяють моделювати перетворювачі, включені в мережу однофазної й трифазної напруги. Типи віртуальних блоків наведені на рисунку 5.1.

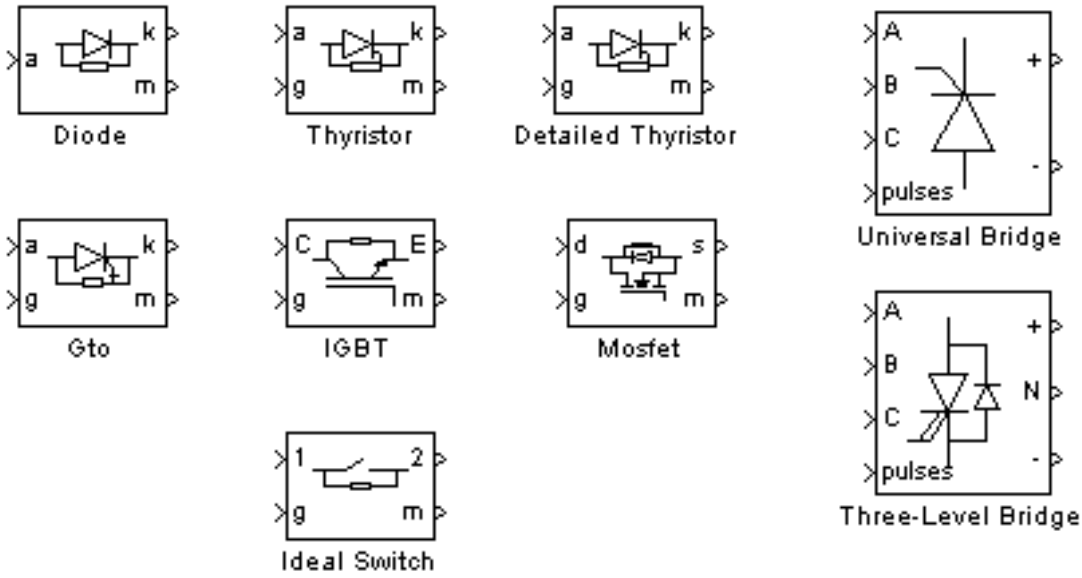

Рисунок 5.1 – Віртуальні блоки напівпровідникових перетворювачів в SimPowerSystems

Слід урахувати, що тиристорні мостові схеми моделюються з використанням блоку Universal Bridge.

Повна віртуальна модель системи векторного керування асинхронним короткозамкненим двигуном з регулятором моменту (power_acdrive.mdl) наведена на рисунку 5.2.

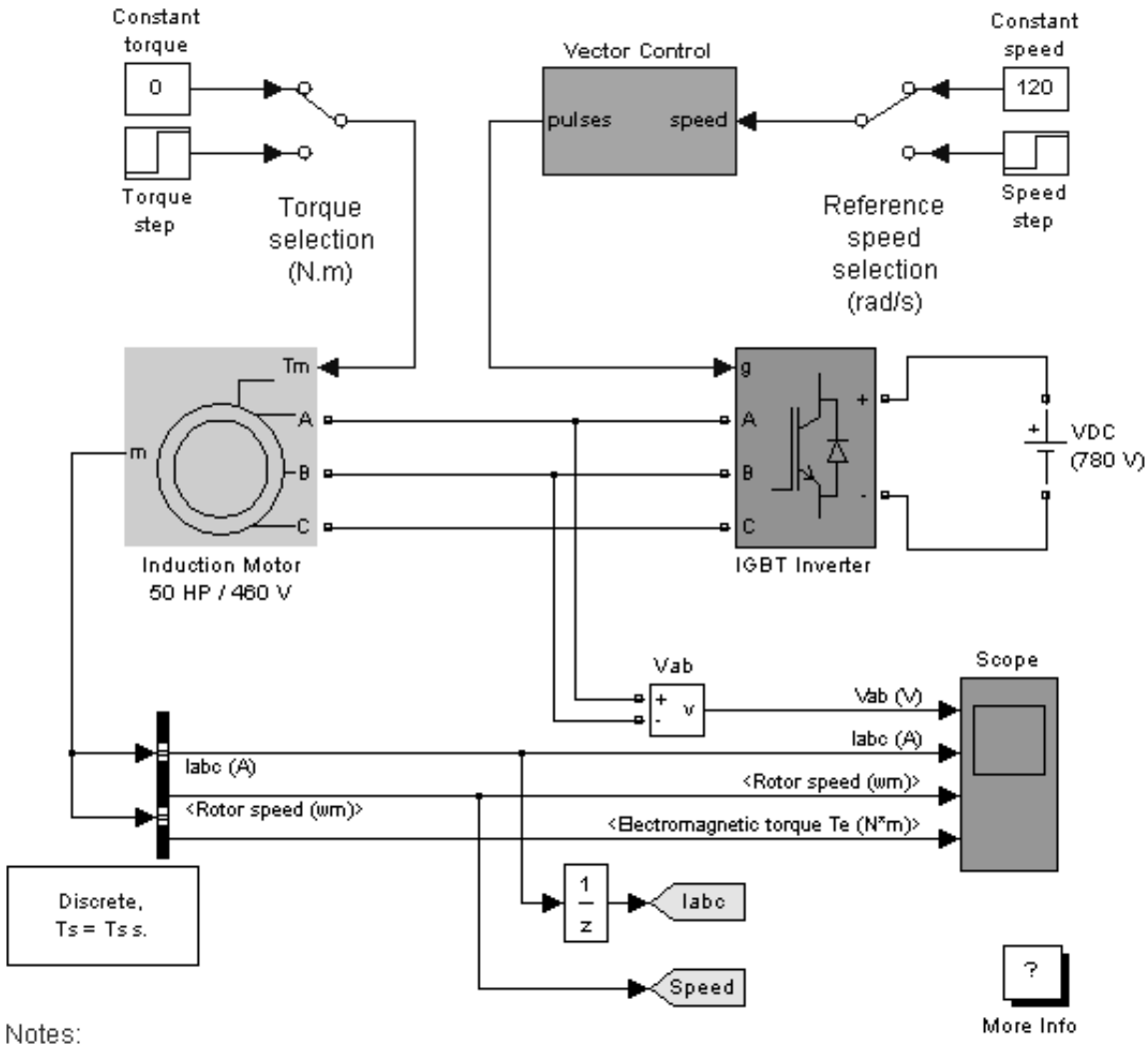

Рисунок 5.2 – Структурна схема моделі системи частотного керування двигуном змінного струму

Модель дозволяє досліджувати струм статора, момент і швидкість обертання в перехідних процесах при зміні початкових умов або параметрів настроювання регуляторів.

Модель містить у собі три основні компоненти:

1 Віртуальний асинхронний двигун (Induction Motor), завдання параметрів якого проводиться у вікні настроювання.

2 Трифазний автономний інвертор на IGBT-транзисторах, параметри якого задаються у вікні його настроювання.

3 Система векторного керування (Vector Control).

У свою чергу, модель системи векторного керування, наведена на рис. 5.3 (її потрібно розкрити подвійним клацанням миші), складається з наступних компонентів:

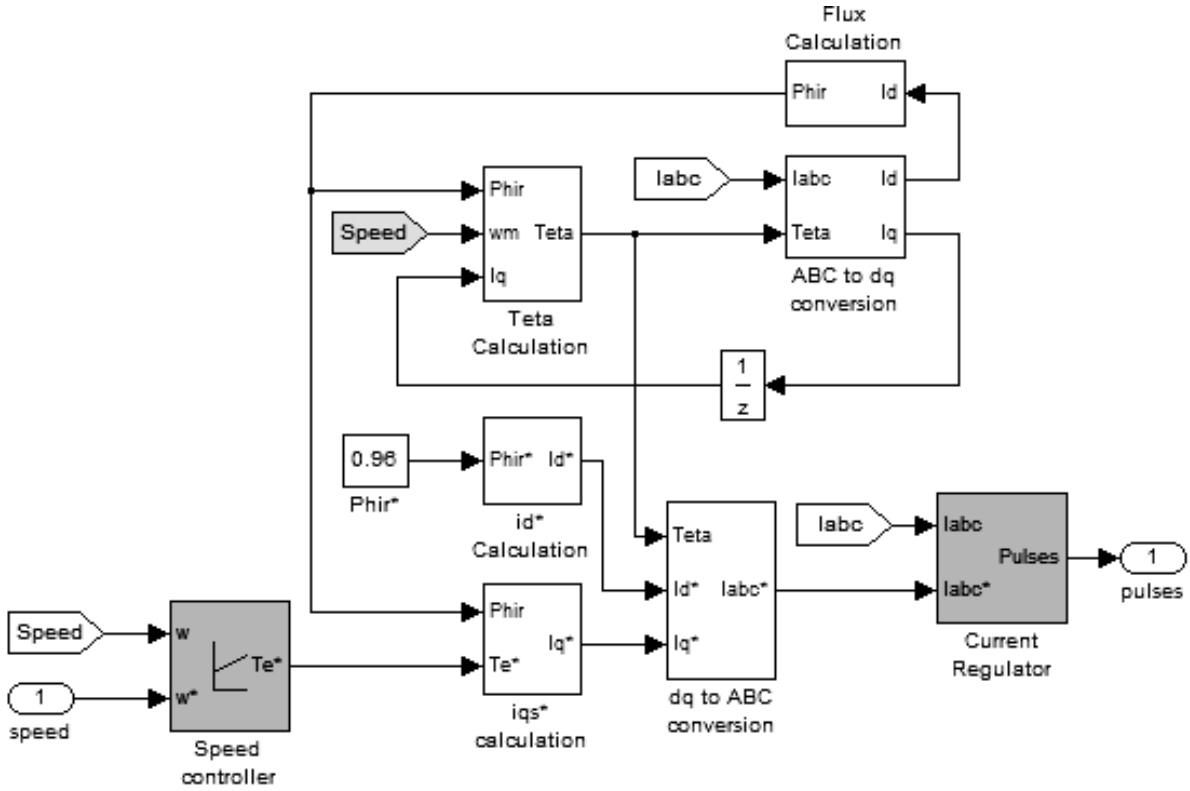

Рисунок 5.3 – Структурна схема моделі системи векторного керування Vector Control

1 Гістерезисний трифазний регулятор струму (Current Regulator), на вхід якого подаються сигнали завдання фазних струмів і сигнали зворотного зв'язку про фактичні значення струмів, а у вікні настроювання можна вказати ширину «струмового коридору».

2 Цифровий ПІ-регулятор швидкості (Speed controller).

3 Блоки перетворення координат (ABC to dq conversion і dq to ABC conversion), які здійснюють перетворення трифазної нерухливої системи координат у двофазну обертову систему й навпаки.

4 Блок визначення вихідної частоти інвертора (Teta Calculation), який забезпечує виконання умови $\overline{\psi}_R = \psi_{Rx}$ й $\psi_{Ry} = 0$ (орієнтування осі x по потокозчепленню ротора).

5 Блок обчислення потоку (Flux Calculator).

6 Блок завдання потоку (Fhir*).

7 Цифровий регулятор струму по осі x (id* Calculation).

8 Блок обчислення струму по осі y (iqs* Calculation), який ділить сигнал з виходу регулятора швидкості (Speed controler) на обчислений потік Phir з виходу блоку Flux Calculator.

Процес моделювання вимагає настроювання блоків моделі. Настроюванню підлягають наступні блоки: двигуна (Induction Motor), регулятора швидкості (Speed controller), регулятора струму (Current Regulator), а також блоки завдання керуючих та збурюючих впливів.

При необхідності дослідження *оптимальних* значень коефіцієнтів регулятора в нелінійних системах керування слід застосовувати пакет прикладних програм Nonlinear Control Design (NCD) Blockset. Цей пакет реалізує *метод динамічної оптимізації Зіглера-Ніколса*.

Набір блоків пакета автоматично набудовує параметри моделюємих систем, ґрунтуючись на заданих користувачем обмеженнях тимчасових характеристик. Враховуючи те, що в цифрових системах керування розрахункові процедури затримують сигнали керування в часі, введення в систему елементів затримки Transport Delay цілком логічно.

У роботі з пакетом використовується метод *click and drag* («клацни й тягни»), який прийнятий в Simulink.

За допомогою пакета (NCD) Blockset можна виконати:

- інтерактивну оптимізацію регуляторів;
- моделювання об'єктів із запізнюванням;
- моделювання схем придушення перешкод;
- настроювання регуляторів в умовах невизначеності параметрів.

Основним блоком пакета є блок оптимізації NCD Outport. Приклад його підключення для оптимізації параметрів ПІД-регулятора показаний на рисунку 5.4.

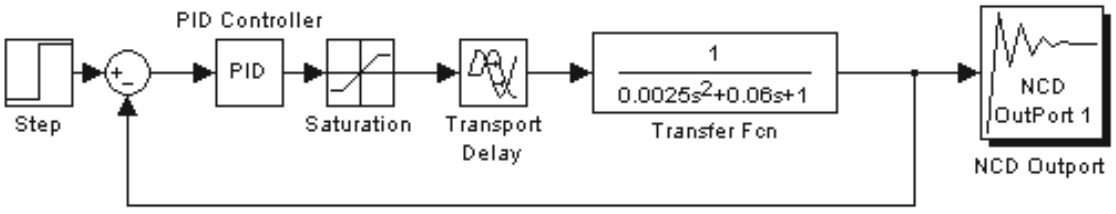

Рисунок 5.4 – Приклад підключення блоку оптимізації NCD Outport

Початкові значення параметрів, що підлягають оптимізації, задаються в режимі командного рядка MATLAB, інші параметри вводяться у вікнах настроювання блоку NCD Outport.

6 РЕКОМЕНДАЦІЇ З РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРОЕКТУ

У сучасних умовах відбувається постійне підвищення складності завдань. Ця складність позначається на можливості застосовувати доступні (відомі) схеми алгоритмів. Щоб зменшити складність, потрібно декомпозувати програму. Декомпозиція дозволяє представити програму у вигляді підпрограм, модулів, компонентів або блоків.

Перевагою такого розподілення програми є можливість переносу й повторного використання готових компонентів програми в інших розробках, що дозволяє прискорити процеси розробки програм. Це особливо важливо для систем керування устаткуванням, де апаратні засоби та виконавчі пристрої, з погляду керування, практично однакові для багатьох процесів. Крім того, програмні компоненти можуть бути створені з використанням тієї мови програмування, яка буде для них найбільш виразною.

В інформаційних задачах створення й обробки документації алгоритми програм не є постійними у зв'язку з мінливими умовами організації виробництва. У таких задачах доводиться враховувати специфічні та перемінливі умови підприємства, а тому для створення програмних додатків доцільно застосовувати об'єктно-орієнтовану методологію розробки програм. Ця методологія дозволяє порівняно легко модернізувати окремі компоненти програми, пристосовуючи їх до нових умов виробництва.

У системах керування технологічного рівня для програмування логічних контролерів застосовуються стандартні мови програмування й стандартні підходи в організації програм. Ці підходи базуються на принципі модульної організації програм, парадигмою якого (по Страуструпу) є: *«Розбий програму так, щоб сховати дані в модулях»*.

У програмних комплексах STEP 7, CoDeSys, TRACE MODE і інших складна програма керування структурується шляхом її розбивки на функціональні блоки. Процес створення блоків закінчується тоді, коли здійснювати подальшу розбивку програми недоцільно або неможливо. При цьому в кожному блоці можна застосувати одну із стандартних мов програмування.

Якщо завдання програми зводиться до керування автоматичним циклом, у якому задіяно ряд виконавчих пристроїв, то програмування слід виконувати мовою S7-HiGraph.

Мова програмування S7-HiGraph заснована на використанні графів станів. Процес розділяється на індивідуальні графи стану з певною функціональною областю дії. Така вистава зручна не тільки для

програмістів, але й для інженерів-механіків, а також інженерів по експлуатації.

Програма структурується в такий спосіб:

1. Завдання автоматизації розділяються на функціональні матеріальні одиниці. Цими одиницями в системі, наприклад, металорізального верстата можуть бути "Затискний пристрій", "Двигун привода головного руху", "Пристрій подачі" і т.п.

2. Поведінка кожної функціональної одиниці описується за допомогою графа станів. Дії, які робить автомат у цих станах, можуть бути зроблені при вході в стан, під час стану й при виході зі стану.

3. Для переходу від одного стану до іншого аналізуються умови, задані при програмуванні. При досягненні встановлених умов автомат здійснює транзакцію – перехід у новий стан.

4. Для програмування умов і дій використовується мова програмування STEP 7 STL.

5. Графи станів вкладаються в групові графи й можуть зв'язуватися один з одним за допомогою повідомлень. Групові графи стану можуть використовуватися як координатори (диспетчери).

6. Для групового графа створюється функція ініціалізації (FC) і блок даних (DB), що забезпечують підготовку автомата при включенні системи. Блок даних містить також дані для індивідуальних графів стани.

Таким чином, процес програмування складається з наступних етапів:

1. Створення графів станів.
2. Оголошення змінних.
3. Програмування станів.
4. Програмування транзакцій.
5. Програмування постійних інструкцій.
6. Створення групового графа.
7. Установка послідовності виконання графів.
8. Призначення фактичних параметрів.
9. Компіляція проекту і його збереження.

На рисунку 6.1 показаний граф станів, створений у редакторі програми S7-HiGraph для керування гідроциліндром привода вертикальної подачі інструмента. Тут у початковому стані 0 проводиться ініціалізація й визначається, у який стан повинна перейти функціональна одиниця після включення живлення – або продовжити операцію, перервану відключенням живлення, або встановити початковий стан графа. Із цією метою аналізується значення змінної INIT_SD. Далі перехід з одного стану в інший визначається політикою групового графа.

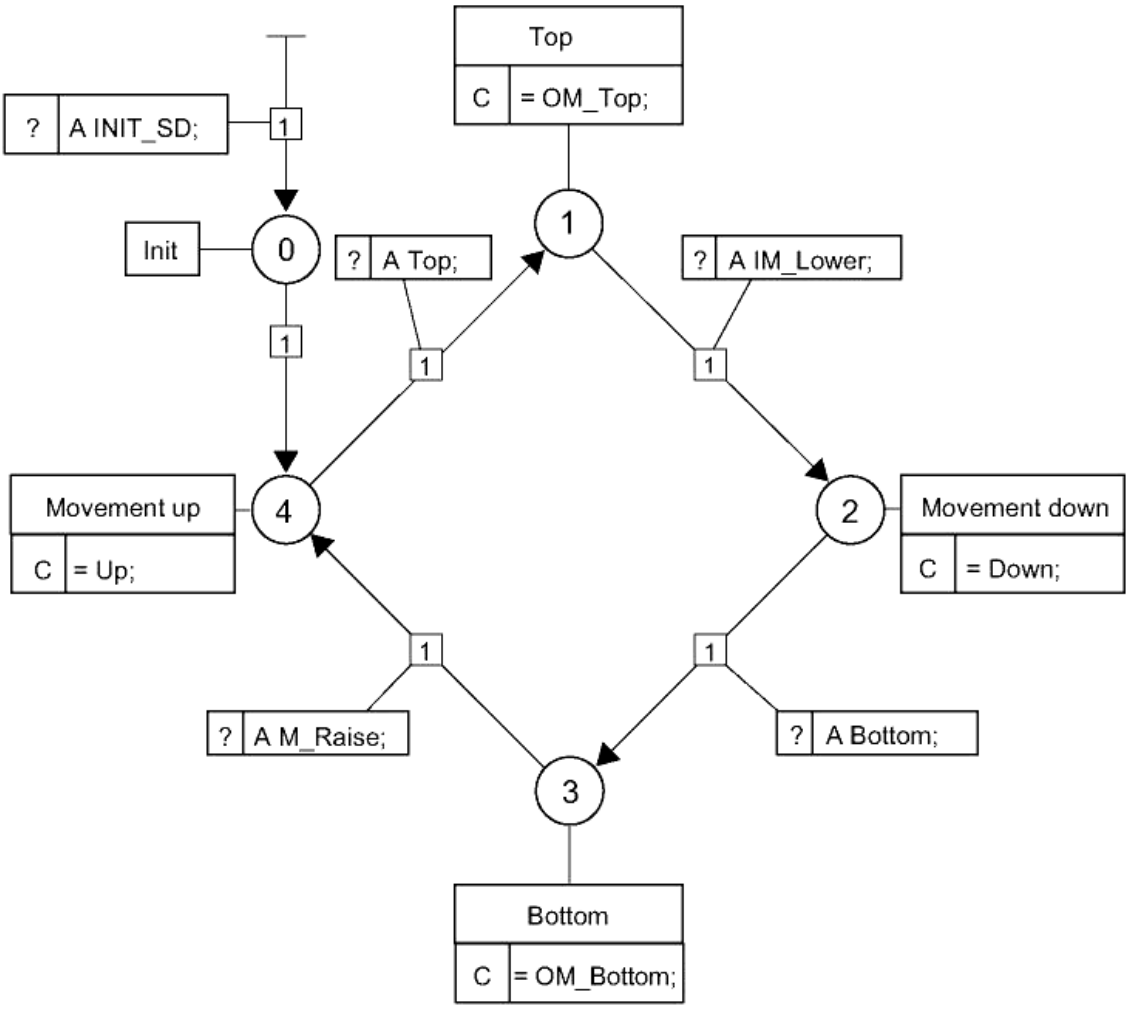

Рисунок 6.1 – Граф станів для програми S7-HiGraph керування гідроциліндром

Підключення станів друг до друга здійснюється командою Transition. У кожному стані програмуються три види дій: дія при вході в стан, циклічні дії в самому стані й дія при виході зі стану, тобто при виконанні наступної транзакції. Пріоритет транзакції задається й відображається в маленькому квадраті, від якого відходить стрілка, що відображає транзакцію.

При додаванні нового стану графа редактор автоматично створює для нього набір змінних (інтерфейс стану), що спрощує процес програмування.

Створена програма може бути налагоджена в цьому ж редакторі.

7 ОФОРМЛЕННЯ ДОКУМЕНТАЦІЇ ПРОЕКТУ

7.1 Вимоги до оформлення текстових документів

Текстові документи слід оформляти з дотриманням вимог стандарту України ДСТУ 3008-95.

У прийнятій на кафедрі АПП системі позначення документів пояснювальна записка курсового проекту в 14-м триместрі студента з номером залікової книжки, наприклад, 4257 буде мати позначення:

КП02.004257.001ПЗ

На текстових документах основний напис виконується *на аркуші 4* (зміст) за ГОСТ 2.104 – 68 форма 2 (рис 7.1).

Рисунок 7.1 – Форма 2 основного напису текстового документа

Загальні дані про склад проекту виконують за формою, наведеною на рисунку 7.2.

№ п/п	Позначення	Найменування	Кіл.	Примеч.
	Текстові документи			
1	КП08.004257.001.П3	Пояснювальна записка	1	35с.
	Графічні документи			
2	КП08.004257.001Е3	Модуль керування	2	
3	✓	✓		

Рисунок 7.2 – Форма відомості проекту

Відомість проекту забезпечується основним написом за формою 1 (рис. 7.3). Вище від основного напису будується таблиця відомості з наступними розмірами стовпців (зліва направо): 15; 60; 70; 15; 25.

Рядок заголовка виконується висотою 15 мм, інші рядки – 10 мм.

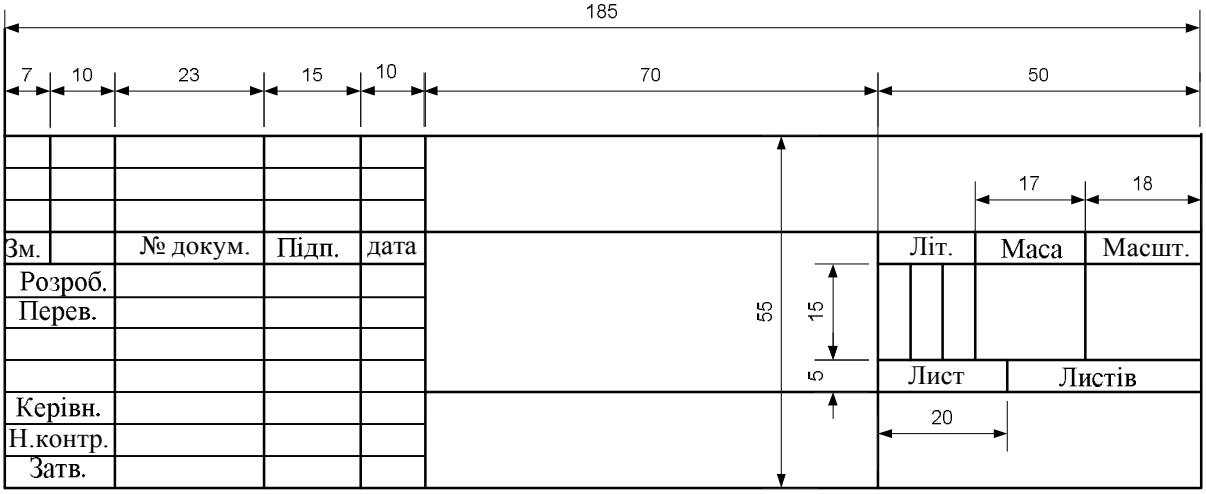

Рисунок 7.3 – Форма 1 основного напису

Пояснювальну записку умовно розділяють на вступну частину, основну частину й додатки.

Вступна частина містить наступні структурні елементи:

- титульний аркуш;
- завдання;
- реферат;
- зміст (лист 4).

Основна частина містить наступні структурні елементи:

- вступ;
- суть звіту;
- висновки;
- перелік посилань.

Додатки розміщають після основної частини звіту.

Реферат призначений для ознайомлення зі звітом. Він повинен бути коротким, інформативним і містити відомості, що дозволяють приймати рішення щодо доцільності прочитання всього звіту.

Реферат повинен містити:

- відомості про обсяг звіту, кількості ілюстрацій, таблиць, додатків, кількості джерел у переліку посилань;
- текст реферату;
- перелік ключових слів.

Текст реферату повинен відображати інформацію, представлену у звіті, у такій послідовності:

- об'єкт дослідження або розробки;
- ціль (мета) роботи;
- результати розробок і їх новизна;
- рекомендації з використання результатів роботи;
- значимість роботи й висновки.

Реферат необхідно виконувати обсягом не більш 500 слів ***на одній сторінці формату А4***. Ключові слова, істотні для розкриття суті звіту, поміщають після тексту реферату. Перелік ключових слів включає від 5 до 15 слів (словосполучень), надрукованих прописними буквами в називному відмінку в рядок через коми.

Вступ повинен містити:

- оцінку існуючого стану процесу;
- актуальність даної роботи й підстава для її проведення;
- ціль роботи і перелік завдань;
- об‘кт і предмет дослідження;
- досягнені результати і їх новизна.

Перелік посилань на джерела інформації приводять у порядку, у якім вони вперше згадуються в тексті. Бібліографічні описи посилань у переліку повинні відповідати діючому стандарту по бібліотечній і видавничій справі. Зразки оформлення посилань приводяться в додатку Г.

Додатки містять матеріал, який:

- є необхідним для повноти звіту, але включення його в основну частину звіту може змінити впорядковане й логічне уявлення про роботу;
- не може бути послідовно розміщений в основній частині звіту через великий обсяг або способи відтворення;
- може бути виключений для широкого кола читачів, але є необхідним для фахівців у даній області.

Текст звіту слід друкувати шрифтом Times New Roman розмір 14, дотримуючи наступні розмірів полів: ***верхнє, ліве й нижнє – не менш 20 мм, праве – не менш 10 мм.***

При виконанні звіту необхідно дотримувати рівномірної щільності, контрастності й чіткості зображення по всьому звіту.

Структурні елементи "РЕФЕРАТ", "ЗМІСТ", "УВЕДЕННЯ", "ВИВОДИ", "ПЕРЕЛІК ПОСИЛАНЬ" ***не нумерують***, а їх найменування служать заголовками структурних елементів.

Розділи й підрозділи повинні мати заголовки. Заголовки структурних елементів звіту й заголовки розділів слід розташовувати в середині рядка й друкувати прописними буквами без крапки наприкінці, не підкреслюючи. *Переноси слів у заголовку розділу не допускаються*.

Заголовки підрозділів, пунктів і підпунктів звіту слід починати з абзацного відступу й ***друкувати малими літерами***, крім першої прописний, не підкреслюючи, без крапки наприкінці.

Абзацний відступ повинен бути однаковим по всьому тексту звіту й рівним ***п'ятьом знакам***.

Відстань між заголовком і наступним або попереднім текстом повинна бути не менш трьох інтервалів.

Відстань між рядками заголовка, а також між двома заголовками приймається такою ж, як у тексті.

Не допускається розміщати найменування розділу, підрозділу, а також пункту й підпункту в нижній частині сторінки, якщо, після нього розташований тільки один рядок тексту.

Сторінки звіту слід нумерувати арабськими цифрами, дотримуючи наскрізну нумерації по всьому тексту звіту. Номер сторінки проставляють у правому верхньому або нижньому куті сторінки без крапки наприкінці.

Титульний аркуш включають у загальну нумерацію сторінок звіту. Номер сторінки на титульному аркуші не проставляють.

Розділи підрозділи, пункти, підпункти звіту слід нумерувати арабськими цифрами.

Підрозділи повинні мати порядкову нумерацію в межах кожного розділу. Номер підрозділу складається з номера розділу й порядкового номера підрозділу, розділених крапкою. Після номера підрозділу крапку не ставлять.

Ілюстрації (креслення, рисунки, графіки, схеми, діаграми, фотознімки) слід розташовувати у звіті ***безпосередньо після тексту***, у якім вони згадуються вперше, або на наступній сторінці. На всі ілюстрації повинні бути дані посилання у звіті.

Ілюстрації слід нумерувати арабськими цифрами порядковою нумерацією в межах розділу. Номер ілюстрації складається з номера розділу й порядкового номера ілюстрації, розділених крапкою, наприклад, рисунок 3.2 – другий рисунок третього розділу.

Ілюстрація позначається словом "Рисунок", яке разом з номером і назвою ілюстрації поміщають ***після пояснюючих даних***.

Таблиці застосовують для оформлення цифрового матеріалу або класифікацій. Горизонтальні й вертикальні лінії, які розмежовують рядки таблиці, а також лінії, що обмежують таблицю ліворуч, праворуч і знизу, можна не проводити, якщо їх відсутність не утрудняє користування таблицею.

Таблицю слід розташовувати безпосередньо після тексту, у якім вона згадується вперше, або на наступній сторінці. На всі таблиці повинні бути посилання в тексті звіту.

Таблиці слід нумерувати арабськими цифрами порядковою нумерацією в межах розділу, за винятком таблиць, що приводяться в додатках. Номер таблиці складається з номера розділу й порядкового номера таблиці, розділених крапкою, наприклад, «Таблиця 2.1» – перша таблиця другого розділу. Таблиця *може мати назву*, яку друкують малими літерами (крім першої прописної) і поміщають над таблицею з абзацним відступом. Назва повинна бути короткою і відбивати зміст таблиці.

Формули й рівняння розташовують безпосередньо після тексту, у якім вони згадуються, посередині сторінки. Вище й нижче кожної формули або рівняння повинне бути залишене не менш одного вільного рядка.

Формули й рівняння у звіті (за винятком формул і рівнянь, наведених у додатках) слід нумерувати порядковою нумерацією в межах розділу. Номер формули або рівняння складається з номера розділу й порядкового номера формули або рівняння, розділених крапкою. Номер формули або рівняння вказують на рівні формули або рівняння в дужках у крайньому правому положенні на рядку.

Формули виконуються в редакторі Equation у *математичному* стилі.

Додатки слід оформляти як продовження звіту на його наступних сторінках. Кожний додаток повинен починатися з нової сторінки. Додаток повинен мати заголовок, надрукований угорі малими літерами з першої прописної симетрично щодо тексту сторінки. Над заголовком у центрі рядка малими літерами з першої прописний повинне бути надруковане слово "Додаток" і далі прописна буква, що позначає додаток.

Додатки слід позначати послідовно прописними буквами (для українського алфавіту виключаються букви Є, З, І, Ї, І, О, Ч, Ь), наприклад, додаток А, додаток Б і т. д.

Один додаток позначається як додаток А.

7.2 Вимоги до оформлення графічної частини

Графічна частина проекту може включати креслення й плакати. Графічна частина оформляється на аркушах формату А4 або на А3. Кількість аркушів визначається завданням на проектування, однак зазвичай достатньо 4-5 аркушів.

Структурні схеми систем керування повинні мати засоби інтеграції у відповідності зі стандартом OPC. На структурних схемах повинне бути показане, які саме технічні засоби і їх модифікації застосовані в системі автоматизації.

На функціональних схемах технологічне устаткування повинне показуватися спрощено, однак давати ясне уявлення про принцип роботи й взаємодії. Технологічні апарати, трубопроводи, датчики, прилади й засоби

автоматизації показуються умовними зображеннями відповідно з вимогами стандартів і повинні мати відповідні написи й позначення.

Принципіальні електричні схеми виконують за правилами, установленим діючими стандартами:

- ГОСТ 2.708-81 – правила виконання електричних схем цифрової обчислювальної техніки;
- ГОСТ 2.759-82 – правила виконання електричних схем аналогової техніки.

На принципіальній схемі зображують усі електричні елементи, необхідні для здійснення й контролю у виробі заданих електричних процесів, а також електричні елементи (рознімання, затискачі й т.п.), якими закінчуються вхідні й вихідні ланцюги.

Для принципіальної схеми повинен бути складений перелік елементів, який може бути оформлений окремим документом.

Зв'язок переліку елементів зі схемою здійснюється через позиційні позначення, що складаються із літерного позначення й порядкового номера елемента. Запис елементів у таблицю проводиться в порядку латинського алфавіту, починаючи з літерного позначення A и закінчуючи Z.

Лінії зв'язку (горизонтальні й вертикальні відрізки) повинні мати мінімальне число зламів, показуються повністю, однак якщо це утрудняє читання схем, допускається їх обривати, закінчуючи стрілкою з оцінкою, куди підключаються, або номером ланцюга, наприклад: «+12В», «402».

На схемах з'єднання й підключення повинні бути показані ланцюги живлення, керування, виміру (контролю) і сигналізації. Елементи схем (модулі, пристрої, шини) повинні мати позиційні позначення й позначення виводів.

Плакати повинні бути постачені заголовками. Основний напис на плакатах не ставиться, однак у нижньому правому куті листа повинен бути напис «До курсового проекту студ. _____».

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Герман-Галкин С. Г. Компьютерное моделирование полупроводниковых систем в MATLAB 6.0: Учеб. пособ. / С.Г. Герман-Галкин. – СПб.: КОРОНА принт, 2001. – 320 с.
2. Дьяконов В. MATLAB. Анализ, идентификация и моделирование систем. Специальный справочник / В. Дьяконов, В. Круглов. – СПб.: Питер, 2002. – 448 с.
3. Голенищев Э. П. Информационное обеспечение систем управления. Серия «Учебники и учебные пособия». / Э.П. Голенищев, И.В. Клименко. - Ростов н/Д: «Феникс», 2003. – 352 с.
4. Лебедев А.М. Следящие электроприводы станков с ЧПУ / А.М. Лебедев, Р.Т. Орлова, А.В. Пальцев – М.: Энергоатомиздат, 1988. – 223 с.
5. Крёнке Д. Теория и практика построения баз данных. 8-е изд. / Д. Крёнке. – СПб.: Питер, 2003. – 800 с.
6. Семенов А. Б. Проектирование и расчет структурированных кабельных систем и их компонентов. /А.Б. Семенов - М.: ДМК Пресс, 2003. – 416 с.

ЕЛЕКТРОННІ РЕСУРСИ

7. Ганс Бергер. Автоматизация с помощью программ STEP 7 LAD и FBD. Программируемые контроллеры SIMATIC S7-300/400. [Електронний ресурс] / SIEMENS, 2001. – Режим доступу: http://books.tr200.ru/v.php?&id=56695&p=0
8. Датчики. Руководство по выбору. [Електронний ресурс] / Schneider Electric/ Режим доступу: http://www.schneider-electric.ru
9. Олифер Н. Базовые технологии локальных сетей. [Електронний ресурс] / Н. Олифер, В. Олифер, Центр Информационных Технологий. - Режим доступу: http://www.citforum.ru/nets/protocols2/index.shtml
10. Преобразователи частоты Altivar 71 (0,37 - 45 кВт / 200 - 240 В). [Електронний ресурс] / Schneider Electric / Руководство пользователя. 2005. - Режим доступу: http://www.altivar.ucoz.ua/load
11. SIEMENS. Каталог CA 01-2007RUS. [Електронний ресурс] / SIEMENS, 2007. – Режим доступу: http://www.automation-drives.ru/support/ca01/

Додаток А

Приблизна тематика курсових проектів

1 Проект модернізації системи керування токарського верстата із ЧПУ типу 16Д20Ф3.

2 Проект модернізації системи керування натягом смуги для прокатного стану 700.

3 Проект модернізації локальної обчислювальної мережі для сервісного центру НКМЗ.

4 Проект автоматизованої виконавчої системи для термічного виробництва СКМЗ.

5 Проект інформаційної системи безпеки для заводу «Славтяжмаш»

6 Розробка автоматизованої системи керування приводом скіпового підйомника.

7 Розробка автоматизованої системи керування приводами подачі каретки й роликів бесцентрового токарського верстата.

8 Розробка системи автоматичного заправлення дроту для електроерозійного агрегату.

9 Проект автоматизації термічної печі для загартування й відпустки металовиробів в умовах Дружковського заводу металовиробів.

10 Розробка підсистеми керування приводом повороту платформи крокуючого екскаватора ЭШ 11/70.

11 Розробка інформаційної системи обліку енергетичних ресурсів заводу «ЭМСС».

12 Проект автоматизованої системи діагностики вузлів редукторів для мостового крана.

Додаток Б
Зразок бланка завдання

Донбаська державна машинобудівна академія
Кафедра автоматизації виробничих процесів

ЗАВДАННЯ

на розробку курсового проекту по дисципліні
«Цифрові системи керування й обробки інформації»

Студент гр. _____
_____
(прізвище, ім'я, по батькові)
Тема курсового проекту:_____
_____
_____
Вихідні дані:_____
_____
Зміст пояснювальної записки
_____
_____
_____
_____
_____
Зміст графічної частини
_____
_____
_____
_____
_____

Завдання видав:
_____ _____
(прізвище, ініціали)
Дата видачі завдання _____

Строк здачі проекту _____

Додаток В
Зразок титульного аркуша розрахунково-пояснювальної записки

Міністерство освіти й науки України
Донбаська державна машинобудівна академія
Кафедра автоматизації виробничих процесів

КУРСОВИЙ ПРОЕКТ

по дисципліні
«Цифрові системи керування й обробки інформації»

На тему:_____
_____
_____
_____

Виконав студент гр. _____ _____
(прізвище й ініціали)

Керівник _____
(прізвище й ініціали)

Краматорськ 2018

Додаток Г
Зразки оформлення переліку посилань до наукових праць, курсових і дипломних проектів ГОСТ 7.1-2003

1. Аверин А. К. Приспособления для металлорежущих станков : справочник / А. К. Аверин. - 7-е изд., перераб. – М. : Машиностроение, 1991. – 303 с.	Книга із 1 автором
2. Болотин Х. Л. Станочные приспособления : учеб. пособие / Х. Л. Болотин, С. П. Костромин. – 5-е изд., перераб. и доп. – М. : Высш. шк., 1992. – 344 с.	Книга з 2 авторами
3. Зверев М. П. Технологическая оснащенность в машиностроении / М. П. Зверев, Э. В. Рыжов, А. В. Аверченков. – Минск : Наука и техника, 1992. – 443 с.	Книга 3-х авторів
4. Изменяющаяся подача / Г. М. Шейнин [и др.]. – М. : Наука, 1992. – 40 с.	Книга 4-х авторів
5. Савельев И. В. Курс общей физики. В 3 т. Т. 1. Механика. Молекулярная физика : учеб. пособие / И. В. Савельев. – 21-е изд., перераб. – М. : Наука, 1992. – 432 с.	Опис тому із багатотомн. вид.
6. Инструкция по крашению изделий из натурального меха :утв. Упр. хим. чистки и крашения М-ва быт. обслуж. РСФСР 12.11.83. - М., 1994. - 16 с.	Інструкція
8. Михайлов А. И. Методика расчета оптимального режима резания / А. И. Михайлов // Труды / Горьков. политехн. ин-т. – Горький, 1992. – Т. 128. – С. 75-77.	Стаття із збірника праць
9. Кислородотерапия в пульмонологии : тез. докл. респ. конф., Тула, 11 - 13 июня 1992 г. / Рос. гос. мед. ун-т и др. ; под общ. ред. А. Г. Чучалина. – Тула : ТППО, 1992. - 57 с.	Тезиси доповідей
10. Абрамов А. С. Вопросы инженерной геодинамики / А. С. Абрамов // Технология машиностроения. – Тула, 1992. – Вып. 1. – С. 1-6.	Стаття із збірника
11. Дементьев А. А. Эффективность научных исследований / А. А. Дементьев // Изв. вузов. Машиностроение. – 1991. – № 6. – С. 4-9.	Стаття із журналу
12. ГОСТ Р 51771 –2001. Аппаратура радиоэлектронная бытовая. Входные и выходные параметры и типы соединений. Технические требования. – Введ. 2002-01-01. – М. : Госстандарт России : Изд-во стандартов, 2001. – IV, 27 с. : ил.	ГОСТ
13. А. с. 107970 СССР, МКИ В25 J 15/00. Устройство для захвата неориентированных деталей типа валов / В. С. Ваулин, В. Г. Кемайкин (СССР). – № 3360585/25 -08 ; заявл. 23.11.81 ; опубл. 30.03.83, Бюл. № 12. - 2 с. : ил.	Авторське свідоцтво

14. Пат. 2187888 Российская Федерация, МПК H 04 B 1/38, H 04 J 13/00. Приемопередающее устройство / Чуева В. И. ; заявитель и патентообладатель Воронеж. науч.-исслед. ин-т связи. - № 2000131736/09 ; заявл. 18.12.00 ; опубл. 20.08.02, Бюл. № 23 (II ч.). - 3 с. : ил.	Патент
15. Винтовой холодильный компрессор ВХ 1400-7-3 : каталог / Центр. ин-т НТИ и техн.-экон. исслед. по хим. и нефт. машиностроению. – М., 1993. – 2 с.	Каталог
16. Вишняков И. В. Модели и методы оценки коммерческих банков в условиях неопределенности :дис. канд. экон. наук : 08.00.13 : защищена 12.02.02 : утв. 24.06.02 / Вишняков Илья Владимирович. - М., 2002. – 234 с. - Библиогр.: с. 220-230.	Дисертація
17. Борисов С. Н. Методы машинной монографии и их приложения : автореф. дис. д-ра техн. наук : 06.17.01 / С. Н. Борисов ; ТулГУ. – М., 2001. – 32 с.	Автореферат дисертації
23. Sosodia M. N. Microwave circuits and passive devices / M. N. Sosodia, D. S. Raghuvanshi. – New York : Wiley, 1991. – 240 p.	Книга на іноземній мові
24. Parker SuSan T. What's new in metallcuttin research / Parker SuSan T. // Amer. Mach. – 1992. – Vol. 129, N 7. – P. 75-77.	Стаття із іноземного журналу

ЕЛЕКТРОННІ РЕСУРСИ

1. Художественная энциклопедия зарубежного классического искусства [Электронный ресурс]. - Электрон. текстовые, граф., зв. дан. и прикладная прогр. (545 Мб). -М. : Большая Рос. энцикл., 1996. - 1 электрон. опт. диск (CD-ROM) : зв., цв. ; 12 см. + рук. пользователя (1л.) + открытка (1 л.). - (Интерактивный мир)

2. Всемирная история в лицах [Электронный ресурс] / РАН, Рос. акад. образования. - Электрон. текстовые данные. - М. : НТЦ «Прогресс», 1996. - 12 электрон. опт. дисков (CD-ROM)

3. Коганов А. В. Время и энтропия: [Электронный ресурс] / А. В. Коганов // Изучение феномена времени : Материалы круглого стола / НИИСИ РАН. – Электрон. дан. (1 файл). - http://www.chronos.linia.ru/reports/koganov_tezisy.html

4. Андреева Е. А. Возникновение и развитие епархиальных женских училищ в России [Электронный ресурс] : Автореф. дис. канд. пед. наук / Е. А. Андреева. – М., 2001. - Режим доступа : http:// www.oim.ru

Додаток Д

Зразок оформлення реферату

РЕФЕРАТ

Розрахунково-пояснювальна записка містить 39 с., 15 рис., 5 таблиць, 3 додатка, 19 джерел. Дослідницька частина містить 13 с.

Об'єкт проектування – інтегрована система керування термічною вертикальною піччю В-1 заводу НКМЗ.

Ціль роботи – підвищення ефективності термічного виробництва.

У проекті виконаний критичний аналіз існуючої системи інформаційного забезпечення ділянки термічного виробництва, сформульовані технічні вимоги до розроблюваної системи керування тепловим режимом печі, а також завдання проекту.

Для підвищення ефективності керування термічним виробництвом із застосуванням мови UML розроблені моделі програмного забезпечення. Для реалізації програми керування термічним режимом печі застосована мова PHP. Виконані розробки дозволяють інтегрувати окремі ділянки термічного виробництва в єдину інформаційну систему із централізованим керуванням.

Для реалізації проекту здійснені розрахунки локальної мережі й зроблений вибір мережного устаткування. Проведене моделювання мережі дозволяє зробити вивід про її працездатність.

ПІЧ ТЕРМІЧНА ВЕРТИКАЛЬНА, АВТОМАТИЗАЦІЯ ТЕРМІЧНОЇ ПЕЧІ, ВИКОНАВЧА СИСТЕМА ВИРОБНИЦТВА, ДІАГРАМА UML, СИСТЕМА ТЕПЛОВОГО РЕЖИМУ ПЕЧІ.

ЗМІСТ