МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ДО ВИКОНАННЯ ПРАКТИЧНИХ РОБІТ

ОСНОВИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

2023 |

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ФАХОВИЙ КОЛЕДЖ ПРОМИСЛОВОЇ АВТОМАТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
ОДЕСЬКОГО НАЦІОНАЛЬНОГО ТЕХНОЛОГІЧНОГО УНІВЕРСИТЕТУ»

ЗАТВЕРДЖУЮ
Заступник директора з
навчально-методичної роботи
_____ Вікторія ОКСАНІЧЕНКО
___.___.2022 року

Методичні вказівки до виконання практичних робіт

з дисципліни Основи програмної інженерії
(Назва дисципліни)

галузі знань 12 «Інформаційні технології»
(шифр і назва галузі знань)

спеціальності 121 «Інженерія програмного забезпечення»
122 «Комп'ютерні науки»
(шифр і назва спеціальності)

освітні програми Інженерія програмного забезпечення
Комп'ютерні науки
(назва освітньо-професійної програми)

Одеса 2022

Методичні вказівки для виконання практичних робіт з дисципліни «Основи програмної інженерії» для підготовки фахових молодших бакалаврів за спеціальностями 121 «Інженерія програмного забезпечення» та 122 «Комп'ютерні науки».

Укладач викладач вищої кваліфікаційної категорії ФКПАІТ ОНТУ Тетяна КОСТИРЕНКО

Методичні вказівки для виконання практичних робіт затверджені на засіданні циклової комісії комп'ютерних наук та інженерії програмного забезпечення ФКПАІТ ОНТУ

Протокол від ___._____.20___ року, № ____

Голова циклової комісії "Комп'ютерних наук та інженерії програмного забезпечення"

_____Тетяна КОСТИРЕНКО

(підпис) (власне ім'я та прізвище)

___._____.20___ року

Методичні вказівки до виконання практичних робіт з дисципліни «Основи програмної інженерії» розроблено згідно навчальної програми з предмету.

Метою вивчення навчальної дисципліни є оволодіння студентами теоретичних знань та набуття практичних навиків з дисципліни основи програмної інженерії, необхідного для спеціальної підготовки та майбутньої професійної діяльності.

Основними завданнями вивчення дисципліни є формування у студентів базових уявлень про основи програмної інженерії та основи моделювання програмного забезпечення; інтелектуальний розвиток особистості, розвиток у студентів логічного мислення і просторової уяви, алгоритмічної, інформаційної та графічної культури, пам'яті, уваги, інтуїції; формування у студентів компетентностей за фахом.

Згідно з вимогами освітньо-професійної програми студенти повинні знати:

- Загальні поняття інженерії програмного забезпечення;
- Поняття інформаційної системи, інформації, компоненти інформаційної системи;
- Поняття життєвого циклу програмного забезпечення, моделі життєвого циклу та його структуру;
- Основні показники якісного програмного забезпечення;
- Принципи на яких базуються методології проектування інформаційних систем;
- Основні поняття методологій SADT, IDEF, DFD.
- Основи побудови ER діаграм різними графічними нотаціями;

Згідно з вимогами освітньо-професійної програми студенти повинні вміти:

- Аналізувати інформаційні системи та класифікувати їх за різними критеріями;
- Будувати SADT-моделі та SADT-діаграми.
- Будувати DFD діаграми;
- Моделювати бази даних за допомогою ER-діаграм за методами Р. Баркера та П. Чена;
- Аналізувати предметну область за допомогою методології моделювання IDEF-3.

Зміст

Практична робота №1. Розробка вимог до ІС

Мета: Скласти і проаналізувати вимоги до інформаційної системи, отримати практичні навики організації діаграми ідентифікації точок зору і діаграми ієрархії точок зору.

Теоретичні відомості:

Проблеми, які доводиться вирішувати фахівцям в процесі створення програмного забезпечення, дуже складні. Природа цих проблем не завжди ясна, особливо якщо розробляється програмна система інноваційна. Зокрема, важко чітко описати ті дії, які повинна виконувати система. Опис функціональних можливостей і обмежень, що накладаються на систему, називається вимогами до цієї системи, а сам процес формування, аналізу, документування та перевірки цих функціональних можливостей і обмежень - **розробкою вимог**.

Вимоги поділяються на призначені для користувача і системні.

Призначені для користувача вимоги - це опис природною мовою (плюс пояснюють діаграми) функцій, які виконуються системою, і обмежень, що накладаються на неї.

Системні вимоги - це опис особливостей системи (архітектура системи, вимоги до параметрів обладнання і т.д.), необхідних для ефективної реалізації вимог користувача.

Розробка вимог

Розробка вимог - це процес, що включає заходи, необхідні для створення і затвердження документа, що містить специфікацію системних вимог. Розрізняють чотири основні етапи процесу розробки вимог:

- Аналіз технічної здійсненності створення системи,
- Формування і аналіз вимог,
- Специфицирование вимог і створення відповідної документації,
- Атестація цих вимог.

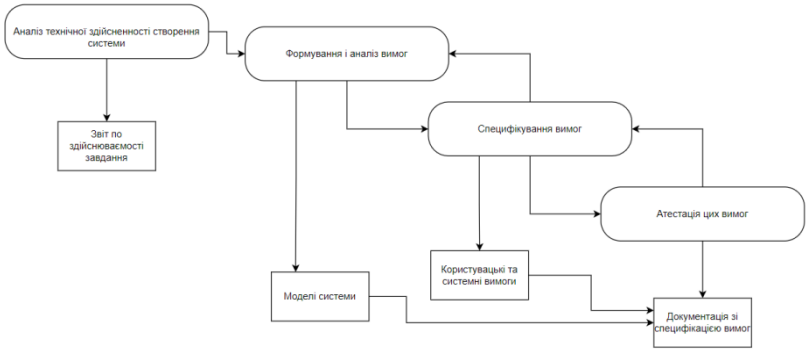


Рисунок 1 – процес розробки вимог

Для нових програмних систем процес розробки вимог повинен починатися з аналізу здійсненності. Початком такого аналізу є загальний опис системи і її призначення, а результатом аналізу - звіт, в якому має бути чітка рекомендація,

продовжувати або ні процес розробки вимог проектованої системи. Іншими словами, аналіз здійсненності повинен освітити наступні питання.

1. Чи відповідає система загальним і бизнес-целям організації-замовника і організації-розробника?

2. Чи можна реалізувати систему, використовуючи існуючі на даний момент технології і не виходячи за межі заданої вартості?

3. Чи можна об'єднати систему з іншими системами, які вже експлуатуються?

Наступним етапом процесу розробки вимог являється формування (визначення) і аналіз вимог. На цьому етапі команда розробників ПО працює із замовником і кінцевими користувачами системи для з'ясування сфери застосування, опису системних сервісів, визначення режимів роботи системи і її характеристик виконання, апаратних обмежень і так далі.

Узагальнена модель процесу формування і аналізу вимог показана на мал. 2. Кожна організація використовує власний варіант цієї моделі, залежний від "місцевих" чинників, : досвіду роботи колективу розробників, типу системи, що розробляється, використовуваних стандартів і так далі.

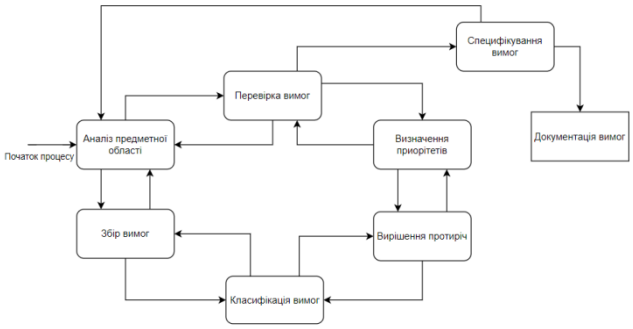


Рисунок 2 - Процес формування і аналізу вимог

Процес формування і аналізу вимог проходить через ряд етапів.

1. Аналіз предметної області. Аналітики повинні вивчити предметну область, де експлуатуватиметься система.

2. Збір вимог. Це процес взаємодії з особами, що формують вимоги. Під час цього процесу триває аналіз предметної області.

3. Класифікація вимог. На цьому етапі безформний набір вимог перетвориться в логічно пов'язані групи вимог.

4. Вирішення протиріч. Без сумніву, вимоги численних осіб, зайнятих в процесі формування вимог, будуть суперечливими. На цьому етапі визначаються і дозволяються протиріччя такого роду.

5. Призначення пріоритетів. У будь-якому наборі вимог одні з них будуть важливіші, ніж інші. На цьому етапі спільно з особами, що формують вимоги, визначаються найбільш важливі вимоги.

6. Перевірка вимог. На цьому етапі визначається їх повнота, послідовність і несуперечність.

Як показано на мал. 2, процес формування і аналізу вимог циклічний, із зворотним зв'язком від одного етапу до іншого. Цикл починається з аналізу предметної області і закінчується перевіркою вимог. Розуміння вимог предметної області збільшується в кожному циклі процесу формування вимог.

Розглянемо три основні підходи до формування вимог: метод, заснований на безлічі опорних точок зору, сценарії і етнографічний метод.

Опорні точки зору

Підхід з використанням різних опорних точок зору до розробки вимог визнає різні (опорні) точки зору на проблему і використовує їх в якості основи побудови та організації як процесу формування вимог, так і безпосередньо самих вимог.

Різні методи пропонують різні трактування виразу "точка зору". Точки зору можна трактувати наступним чином.

Як джерело інформації про системні дані. У цьому випадку на основі опорних точок зору будується модель створення і використання даних в системі. У процесі формування вимог відбираються всі такі точки зору (і на їх основі визначаються дані), які будуть створені або використані при роботі системи, а також способи обробки цих даних.

Як структура уявлень. В цьому випадку точки зору розглядаються як особлива частина моделі системи. Наприклад, на основі різних точок зору можуть розроблятися моделі "сутність-зв'язок", моделі кінцевого автомата і т.д.

Як одержувачі системних сервісів. В цьому випадку точки зору є зовнішніми (щодо системи) одержувачами системних сервісів. Точки зору допомагають визначити дані, необхідні для виконання системних сервісів або їх управління.

Найбільш ефективним підходом до аналізу таких систем є використання зовнішніх опорних точок зору. На основі цього підходу розроблено метод **VORD** (Viewpoint-Oriented Requirements Definition - визначення вимог на основі точок зору) для формування та аналізу вимог. Основні етапи методу VORD показані на рис. 3:

- Ідентифікація точок зору, які отримують системні сервіси, і ідентифікація сервісів, відповідних кожній точці зору.
- Структурування точок зору - створення ієрархії згрупованих точок зору. Загальносистемні сервіси надаються більш високих рівнів ієрархії і успадковуються точками зору нижчого рівня.
- Документування опорних точок зору, яке полягає в точному описі ідентифікованих точок зору і сервісів.
- Відображення системи точок зору, яка показує системні об'єкти, визначені на основі інформації, що містяться в опорних точках зору.

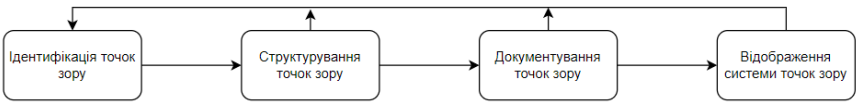


Рисунок 3 – Метод VORD

Приклад. Розглянемо використання методу VORD на перших трьох кроках аналізу вимог для системи підтримки замовлення і обліку товарів у бакалійній крамниці. У бакалійній крамниці для кожного товару фіксується місце зберігання (певна полиця), кількість товару і його постачальник. Система підтримки замовлення і обліку товарів повинна забезпечувати додавання інформації про новий товар, зміна або видалення інформації про наявний товар, зберігання (додавання, зміна і видалення) інформації про постачальників, що включає в себе назва фірми, її адреса і телефон. За допомогою системи складаються замовлення постачальникам. Кожне замовлення може містити кілька позицій, у кожній позиції вказуються найменування товару і його кількість у замовленні. Система на вимогу користувача формує і видає на друк наступну довідкову інформацію:

- список всіх товарів;
- список товарів, що є в наявності;
- список товарів, кількість яких необхідно поповнити;
- список товарів, що поставляються даним постачальником.

Першим кроком у формуванні вимог є **ідентифікація опорних точок зору**. У всіх методах формування вимог, заснованих на використанні точок зору, початкова ідентифікація є найбільш важким завданням. Один з підходів до ідентифікації точок зору - метод "**мозкової атаки**", коли визначаються потенційні системні сервіси і організації, що взаємодіють із системою. Організується зустріч осіб, що задіяні у формуванні вимог, які пропонують свої точки зору. Ці точки зору представляються у вигляді діаграми, що складається з низки областей у вигляді кола, що відображають можливі точки зору (рис. 4). Під час "мозкової атаки" необхідно ідентифікувати потенційні опорні точки зору, системні сервіси, вхідні дані, нефункціональні вимоги, управлячі події і виняткові ситуації.

Наступною стадією процесу формування вимог буде ідентифікація опорних точок зору (на рис. 4 показані у вигляді темних кругових областей) і сервісів (показані у вигляді затінених областей). Сервіси повинні відповідати опорним точкам зору. Але можуть бути сервіси, які не поставлені їм у відповідність. Це означає, що на початковому етапі "мозкової атаки" деякі опорні точки зору не були ідентифіковані.

Рисунок 4 - Діаграма ідентифікації точок зору

У таблиці 1 показаний розподіл сервісів для деяких ідентифікованих на рис. 4 точок зору. Той самий сервіс може бути співвіднесений з декількома точками зору.

Таблиця 1 - Сервіси, співвіднесені з точками зору

Клієнт	**Покупець**	**Постійний покупець**	**Товар**	**Постачальник**	**Продавець**	**Адміністратор**
Перевірка наявності товару	Занесення в список постійних клієнтів	Одержання знижки	Прийом товару	Занесення в базу даних (назва, адреса, телефон і т.д.)	Продаж товару	Доступ до бази даних
Покупка товару		Одержання інформацію про нові надходження	Занесення в базу даних (дані про постачальника, кількість, місце зберігання.)		Друк чека	Перевірка статистики
Одержання чека			Визначення ціни		Доступ до каталогу	Перевизначення ціни
Замовлення товару			Перевизначення ціни		Перевірка наявності товару	Оформлення замовлення постачальникові
Занесення покупця і суми покупки в базу даних			«товар, що купується» або «товар, що не купується»		Оформлення замовлення покупцеві	Друк замовлення

Інформація, що витягнута з точок зору, використовується для заповнення форм шаблонів точок зору і організації точок зору в ієрархію спадкування. Це дозволяє побачити загальні точки зору і повторно використовувати інформацію в ієрархії спадкування. Сервіси, дані і керуюча інформація успадковуються підмножиною точок зору. На рис. 5 показана частина ієрархії точок зору для системи підтримки замовлення і обліку товарів.

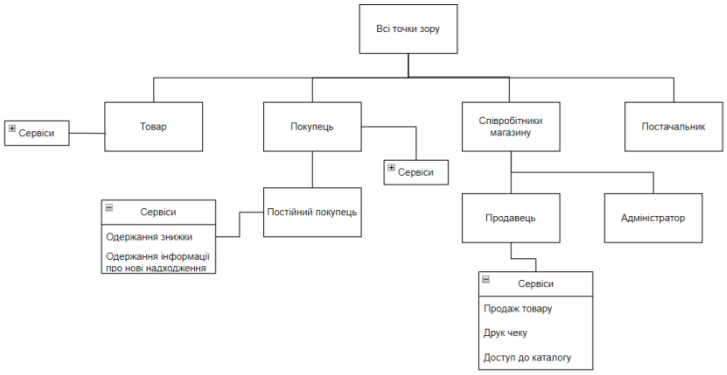


Рисунок 5 – Фрагмент ієрархії точок зору

Атестація вимог

Атестація повинна продемонструвати, що вимоги дійсно визначають ту систему, що хоче мати замовник. Перевірка вимог важлива, тому що помилки в специфікації вимог можуть привести до переробки системи і більших витрат, якщо будуть виявлені під час процесу розробки системи або після введення її в експлуатацію. Вартість внесення в систему змін, необхідних для усунення помилок у вимогах, набагато вище, ніж виправлення помилок проектування або кодування. Причина в тому, що зміна вимог зазвичай спричиняє значні зміни в системі, після внесення яких вона повинна пройти повторне тестування.

Під час процесу атестації повинні бути виконані різні типи перевірок вимог.

- Перевірка правильності вимог. Користувач може вважати, що система необхідна для виконання деяких певних функцій. Проте подальші роздуми і аналіз можуть призвести до необхідності введення додаткових або нових функцій. Системи призначені для різних користувачів з різними потребами, і тому набір вимог буде являти собою деякий компроміс між вимогами користувачів системи.

- Перевірка на несуперечність. Специфікація вимог не повинна містити протиріч. Це означає, що у вимогах не повинні бути суперечних один одному обмежень або різних описів однієї і тої ж системної функції.

- Перевірка на повноту. Специфікація вимог повинна містити вимоги, які визначають всі системні функції і обмеження, що накладаються на систему.

- Перевірка на виконуємість. На основі знання існуючих технологій вимоги повинні бути перевірені на можливість їхнього реального виконання. Тут також перевіряються можливості фінансування і графік розробки системи.

Існує ряд методів атестації вимог, які можна використовувати спільно або кожний окремо (дивитись лекцію).

Користувальницькі і системні вимоги

На підставі отриманих моделей будуються користувальницькі вимоги, тобто як було сказано на початку, опис природною мовою функцій, що виконуються системою, і обмежень, що накладаються на неї.

Користувальницькі вимоги повинні описувати зовнішнє поводження системи, основні функції і сервіси надавані системою, її нефункціональні властивості. Необхідно виділити опорні точки зору і згрупувати вимоги відповідно до них.

Користувальницькі вимоги можна оформити як простим перерахуванням, так і використовуючи нотацію варіантів використання.

Далі складаються системні вимоги. Вони включать у себе:

- *Вимоги до архітектури системи*. Наприклад, число і розміщення сховищ і серверів додатків.

- *Вимоги до параметрів устаткування*. Наприклад, частота процесорів серверів і клієнтів, обсяг сховищ, розмір оперативної і відео пам'яті, пропускна здатність каналу і т.д.

- *Вимоги до параметрів системи*. Наприклад, час відгуку на дію користувача, максимальний розмір переданого файлу, максимальна швидкість передачі даних, максимальне число одночасно працюючих користувачів і т.д.

- *Вимоги до програмного інтерфейсу*.

- *Вимоги до структури системи*. Наприклад, Масштабованість, розподіленість, модульність, відкритість.

масштабованість – можливість поширення системи на велику кількість машин, що не приводить до втрати працездатності і ефективності, при цьому здатність системи нарощувати свою потужність повинна визначатися тільки потужністю відповідного апаратного забезпечення.

розподіленість - система повинна підтримувати розподілене зберігання даних.

модульність - система повинна складатися з окремих модулів, інтегрованих між собою.

відкритість - наявність відкритих інтерфейсів для можливої доробки і інтеграції з іншими системами.

- *Вимоги щодо взаємодії і інтеграції з іншими системами*. Наприклад, використання загальної бази даних, можливість одержання даних з баз даних певних систем і т.д.

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.

- Побудувати опорні точки зору на підставі методу VORD для формування і аналізу вимог. Результатом повинні з'явитися дві діаграми: діаграма ідентифікації точок зору і діаграма ієрархії точок зору.

- На підставі отриманих діаграм ідентифікації точок зору та діаграми ієрархії точок зору сформувати вимоги користувача і системні вимоги.

- Провести атестацію вимог, указати які типи перевірок вибрали.

- Скласти звіт, що включає всі отримані рівні моделі, опис функціональних блоків, потоків даних, сховищ і зовнішніх об'єктів.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи

- Відповідь на контрольні питання

- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що таке вимоги до ІС?
2. Які є типи вимог?
3. Які є методи виявлення вимог?
4. Що таке опорна точка зору?
5. Для чого використовують метод VORD?
6. Що включають в себе системні вимоги?

Практична робота №2: Розробка ТЗ

Мета: Отримати практичні навики оформлення технічного завдання на розробку інформаційної системи.

Теоретичні відомості:

Технічне завдання являє собою документ, у якому сформульовані основні цілі розробки, вимоги до програмного продукту, визначені строки і етапи розробки і регламентований процес приймально-здавальних випробувань. У розробці технічного завдання беруть участь як представники замовника, так і представники виконавця. В основі цього документа лежать вихідні вимоги замовника, аналіз передових досягнень техніки, предпроєктних досліджень, наукового прогнозування і т. і.

Порядок розробки технічного завдання

Розробка технічного завдання виконується в наступній послідовності:

1. встановлюють набір виконуваних функцій, а також перелік і характеристики вихідних даних;
2. визначають перелік результатів, їх характеристики і способи подання;
3. уточнюють середовище функціонування програмного забезпечення:
4. конкретну комплектацію і параметри технічних засобів;
5. версію використовуваної операційної системи і, можливо, версії і параметри іншого встановленого програмного забезпечення, з яким має взаємодіяти у майбутньому програмний продукт;
6. у випадках, коли програмне забезпечення збирає і зберігає деяку інформацію або включається у керування яким-небудь технічним процесом, необхідно також чітко регламентувати дії програми у випадку збоїв устаткування і енергопостачання.

Технічне завдання оформляють відповідно до ГОСТ 19.106-78 на аркушах формату А4 і А3 за ГОСТ 2.301-68, як правило, без заповнення полів листа. Номера аркушів (сторінок) проставляють у верхній частині листа над текстом.

Технічне завдання повинне містити наступні розділи:

• вступ;

• найменування і область застосування;

• підстави для розробки;

• призначення розробки;

• технічні вимоги до програми або програмного виробу;

• техніко-економічні показники;

• стадії й етапи розробки;

• порядок контролю і приймання;

• додатки.

Залежно від особливостей програми або програмного виробу допускається уточнювати зміст розділів, уводити нові розділи або поєднувати окремі з них. При необхідності допускається в технічне завдання включати додатки.

Зміст розділів

2.1. Вступ повинен включати коротку характеристику області застосування програми або програмного продукту, а також об'єкта (наприклад, системи), у якому передбачається їх використовувати. Основне призначення вступу - продемонструвати актуальність даної розробки й показати, яке місце ця розробка займає в ряді подібних.

2.2. У розділі «Найменування і область застосування» вказують найменування, коротку характеристику області застосування програми або програмного виробу і об'єкта, у якому використовують програму або програмний виріб.

2.3. У розділі «Підстава для розробки» повинні бути зазначені:

• документ (документи), на підставі яких ведеться розробка. Таким документом може служити план, наказ, договір т.і.

• організація, що затвердила цей документ, і дата його затвердження;

• найменування і (або) умовна позначка теми розробки.

2.4. У розділі «Призначення розробки» повинне бути зазначене функціональне й експлуатаційне призначення програми або програмного виробу.

2.5. Розділ «Технічні вимоги до програми або програмного виробу» повинен містити наступні підрозділи:

• вимоги до функціональних характеристик;

• вимоги до надійності;

• умови експлуатації;

• вимоги до складу й параметрів технічних засобів;

• вимоги до інформаційної і програмної сумісності;

• вимоги до маркування та пакування;

• вимоги до транспортування і зберігання;

• спеціальні вимоги.

2.5.1. У підрозділі «Вимоги до функціональних характеристик» повинні бути зазначені вимоги до складу виконуваних функцій, організації вхідних і вихідних даних, часовим характеристикам т.і.

2.5.2. У підрозділі «Вимоги до надійності» повинні бути зазначені вимоги до забезпечення надійного функціонування (забезпечення стійкого функціонування, контроль вхідної й вихідної інформації, час відновлення після відмови т.і.).

2.5.3. У підрозділі «Умови експлуатації» повинні бути зазначені умови експлуатації (температура навколишнього повітря, відносна вологість і т.п. для обраних типів носіїв даних), при яких повинні забезпечуватися задані характеристики, а також вид обслуговування, необхідна кількість і кваліфікація персоналу.

2.5.4. У підрозділі «Вимоги до складу і параметрів технічних засобів» указують необхідний склад технічних засобів та їх технічних характеристик.

2.5.5. У підрозділі «Вимоги до інформаційної й програмної сумісності» повинні бути зазначені вимоги до інформаційних структур на вході і виході, методам вирішення, вихідним кодам, мовам програмування. При необхідності повинна забезпечуватися захист інформації і програм.

2.5.6. У підрозділі «Вимоги до маркування і пакування» у загальному випадку вказують вимоги до маркування програмного виробу, варіанти і способи пакування.

2.5.7. У підрозділі «Вимоги до транспортування і зберігання» повинні бути зазначені для програмного виробу умови транспортування, місця зберігання, умови зберігання, умови складування, строки зберігання в різних умовах.

2.5.8. У розділі «Техніко-економічні показники» повинні бути зазначені: орієнтовна економічна ефективність, передбачувана річна потреба, економічні переваги розробки в порівнянні із кращими вітчизняними і закордонними зразками або аналогами.

2.6. У розділі «Стадії і етапи розробки» встановлюють необхідні стадії розробки, етапи і зміст робіт (перелік програмних документів, які повинні бути розроблені, погоджені й затверджені), а також, як правило, строки розробки і визначають виконавців.

2.7. У розділі «Порядок контролю і приймання» повинні бути зазначені види випробувань і загальні вимоги до приймання роботи.

2.8. У додатках до технічного завдання при необхідності приводять:

• перелік науково-дослідних і інших робіт, що обґрунтовують розробку;

• схеми алгоритмів, таблиці, опису, обґрунтування, розрахунки й інші документи, які можуть бути використані при розробці;

• інші джерела розробки.

У випадках, якщо які-небудь вимоги, передбачені технічним завданням, замовник не пред'являє, треба у відповідному місці вказати «Вимоги не пред'являються».

Приклад 1.1. Розробити технічне завдання на програмний продукт, призначений для сортування одномірного масиву.

Розроблювальна програма повинна. забезпечувати можливість виконання наступних функцій:

- уведення розміру масиву і самого масиву;
- зберігання масиву в пам'яті;
- вибір методу сортування;
- виведення текстового опису методу сортування;
- виведення результату сортування.

Приклад розробки технічного завдання на програмний продукт

1. Вступ

Це технічне завдання поширюється на розробку програми сортування одномірного масиву методами бульбашки, прямого вибору, Шелла і швидкого сортування, призначеної для використання студентами при вивченні курсу розробки програмного забезпечення.

2. Підстава для розробки

2.1. Програма розробляється на основі навчального плану циклової комісії комп’ютерних наук та інженерії програмного забезпечення.

2.2. Найменування роботи:

«Програма сортування одномірного масиву».

2.3. Виконавець: компанія Student.

2.4. Співвиконавці: немає.

3. Призначення

Програма призначена для використання студентами при вивченні теми «Обробка одномірних масивів».

4. Вимоги до програми або програмного виробу

4.1. Вимоги до функціональних характеристик

4.1.1. Програма повинна забезпечувати можливість виконання наступних функцій:

- уведення розміру масиву і самого масиву;
- зберігання масиву в пам'яті;
- вибір методу сортування;
- виведення текстового опису методу сортування;
- виведення результату сортування.

4.1.2. Вихідні дані:

- розмір масиву, заданий цілим числом;
- масив.

4.1.3. Організація вхідних і вихідних даних

Вхідні дані надходять з клавіатури.

Вихідні дані відображаються на екрані і при необхідності друкуються.

4.2. Вимоги до надійності

Передбачити контроль введення даних.

Передбачити блокування некоректних дій користувача при роботі із системою.

4.3. Вимоги до складу і параметрів технічних засобів.

Система повинна працювати на IBM-сумісних персональних комп'ютерах.

Мінімальна конфігурація:

• тип процесора. Pentium і вище;

• обсяг оперативного запам'ятовувального пристрою 32 Мб і більше;

• обсяг вільного місця на жорсткому диску 40 Мб.

Конфігурація, що рекомендується:

• тип процесора. Pentium II 400;

• обсяг оперативного запам'ятовувального пристрою 128 Мб;

• обсяг вільного місця на жорсткому диску 60 Мб.

4.4. Вимоги до програмної сумісності.

Програма повинна працювати під управлінням сімейства операційних систем Win 32 (Windows 95/98/2000/ME/XP і т.п.).

5. Вимоги до програмної документації

5.1. Розроблювані програмні модулі повинні бути самодокументовані, тобто тексти програм повинні містити всі необхідні коментарі.

5.2. Розроблювана програма повинна включати довідкову інформацію про роботу програми, опис методів сортування і підказки студентам.

5.3. До складу супровідної документації повинні входити:

5.3.1. Пояснювальна записка на п'яти аркушах, що містить опис розробки.

5.3.2. Посібник користувача.

Приклад 1.2. Розробити технічне завдання на розробку «Модуля автоматизованої системи оперативно-диспетчерського керування теплопостачанням корпусів ФКПАІТ ОНТУ»

1. Вступ

Робота виконується в рамках проекту «Автоматизована система оперативно-диспетчерського керування електротеплопостачанням корпусів ФКПАІТ ОНТУ».

2. Підстава для розробки

2.1. Підставою для даної роботи служить договір № 1234 від 10 вересня 2022р.

2.2. Найменування роботи:

«Модуль автоматизованої системи оперативно-диспетчерського керування теплопостачанням корпусів ФКПАІТ ОНТУ».

2.3. Виконавець: Студент IV курсу спеціальності 121 «Інженерія програмного забезпечення» Іваненко Іван

2.4. Співвиконавці: немає.

3. Призначення розробки

Створення модуля для контролю і оперативної корекції стану основних параметрів електротеплопостачання корпусів ФКПАІТ ОНТУ.

4. Технічні вимоги

4.1. Вимоги до функціональних характеристик.

4.1.1. Склад виконуваних функцій.

Розроблювальне ПЗ повинне забезпечувати:

• збір і аналіз інформації про витрату тепла, гарячої і холодної води по даним теплолічильників SA-94 на всіх теплових виходах;

• збір і аналіз інформації із пристроїв керування системами повітряного опалення і кондиціювання типу РТ1 і РТ2 ;

• попередній аналіз інформації на предмет знаходження параметрів у припустимих межах і сигналізування при виході параметрів за межі допуску;

• видачу рекомендацій з подальшої роботи;

• відображення поточного стану по наборі параметрів - циклічно постійно (режим роботи цілодобовий), при збереженні періодичності контролю інших параметрів;

• візуалізацію інформації з витрати теплоносія:

- поточну, аналогічно показанням лічильників;

- з нагромадженням за минулу добу, тиждень, місяць - у вигляді погодинного графіка для інформації за добу і тиждень;

- добова витрата - для інформації за місяць.

Для пристроїв керування приточною вентиляцією поточна інформація повинна містити номер приточної системи і всі параметри, що видаються на власний індикатор.

По окремому запиті здійснюються внутрішні настроювання.

Наприкінці звітного періоду система повинна архівувати дані.

4.1.2. Організація вхідних і вихідних даних.

Вихідні дані в систему надходять у вигляді значень з датчиків, встановлених у приміщеннях Коледжу. Ці значення відображаються на комп'ютері диспетчера. Після аналізу інформації, що надійшла, оператор диспетчерського пункту встановлює необхідні параметри для пристроїв, що регулюють опалення і вентиляцію в приміщеннях. Можлива також автоматична установка деяких параметрів для пристроїв регулювання.

Основний режим використання системи - щоденна робота.

4.2. Вимоги до надійності.

Для забезпечення надійності необхідно перевіряти коректність одержуваних даних з датчиків.

4.3. Умови експлуатації і вимоги до складу і параметрів технічних засобів.

Для роботи системи повинен бути виділений відповідальний оператор.

Вимоги до складу і параметрів технічних засобів уточнюються на етапі ескізного проектування системи.

4.4. Вимоги до інформаційної і програмної сумісності.

Програма повинна працювати на платформах Windows 98/ NT/2000/XP/VISTA.

4.5. Вимоги до транспортування і зберігання.

Програма поставляється на лазерному носії інформації. Програмна документація поставляється в електронному і друкованому вигляді.

4.6. Спеціальні вимоги:

• програмне забезпечення повинне мати дружній інтерфейс, розрахований на кваліфікованого (у плані комп'ютерної грамотності) користувача;

• через об'ємність проєкту задачі передбачається вирішувати поетапно, при цьому модулі ПЗ, створені в різний час, повинні припускати можливість нарощування системи і бути сумісні один з одним, тому документація на прийняте експлуатаційне ПЗ повинна містити повну інформацію, необхідну для роботи програмістів з ним;

• мова програмування - на вибір виконавця, повинен забезпечувати можливість інтеграції програмного забезпечення з деякими видами периферійного устаткування (наприклад, лічильник SA-94 і т.п.).

5. Вимоги до програмної документації.

Програмна документація повинна бути підготована згідно вимог Єдиної Системи Програмної Документації (ЕСПД) і включати такі документи:

- керівництво користувача;
- керівництво адміністратора;
- опис застосування.

6. Техніко-економічні показники

Ефективність системи визначається зручністю використання системи для контролю і керування основними параметрами теплозабезпечення приміщень ФКПАІТ ОНТУ, а також економічною вигодою, отриманої від впровадження апаратно-програмного комплексу.

7. Порядок контролю й приймання

Після передачі Виконавцем окремого функціонального модуля програми Замовникові останній має право тестувати модуль протягом 7 днів. Після тестування Замовник повинен прийняти роботу по цьому етапу або в письмовому виді викласти причину відмови прийняття. У випадку обґрунтованої відмови Виконавець зобов'язується доробити модуль.

8. Календарний план робіт

№ етапу	Назва етапу	Строки етапу	Чим закінчується етап
1.	Вивчення предметної області. Проектування системи. Розробка пропозицій по	01.02.2022-28.02.2022	Пропозиції по роботі системи. Акт здачі-приймання

	реалізації системи		
2.	Розробка програмного модуля по збору і аналізу інформації з лічильників і пристроїв керування. Впровадження системи для одного з корпусів	01.03.2022-31.08.2022	Програмний комплекс, що вирішує поставлені задачі для пілотного корпуса №1. Акт здачі-приймання
3.	Тестування й налагодження модуля. Впровадження системи в всіх корпусах ФКПАІТ ОНТУ	01.09.2022-30.12.2022	Готова система контролю тепло-забезпечення ФКПАІТ ОНТУ, встановлена у диспетчерському пункті. Програмна документація. Акт здачі-приймання робіт.

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Обрати предметну область за варіантом скориставшись Додатком №1 або обрати власну предметну область оговоривши її з викладачем і зробивши її короткий опис.
- Розробити технічне завдання на розробку ІС.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що таке технічне завдання? Для чого його складають?
2. Як оформлюється ТЗ?
3. З яких розділів складається ТЗ?

Практична робота №3: Побудова ER-діаграми за нотацією Баркера

Мета: Отримати практичні навики побудови моделі «сутність-зв'язок» з використанням нотації Баркера.

Теоретичні відомості:

Нотація Баркера та її модифікації (наприклад, Crow's Foot) для ERD, були розроблені для усунення недоліків ERD у нотації Чена, зокрема таких:

- ERD в нотації Чена не дають наочного зображення усієї моделі даних, оскільки для визначення атрибутів існує окремий вид діаграм; якщо поєднати ERD разом з діаграмами атрибутів, то в результаті отримується занадто велика і погано читабельна схема;

- Взаємозв'язки на ERD в нотації Чена зображають окремими вузлами, які за розмірами є спів вимірними з вузлами-сутностями, що не дає змогу сконцентрувати увагу власне на найважливіших елементах ERD-сутностей.

У нотації Бракера використовують лише один тип діаграм – ERD, які містять як взаємозв'язки між сутностями, так і атрибути сутностей.

Сутність (Entity) – це реальний або уявний об'єкт, який має історичне значення для аналізованої предметної області, інформація про який полягає збереженню. Кожна сутність зображується прямокутником, у верхній частині якого вказується унікальне ім'я (рис. 1). Сутність має один або декілька атрибутів, які зображуються списком імен цих атрибутів у середині прямокутника сутності. Ім'я кожного атрибута є унікальним в межах сутності і є іменником або іменниковою фразою, що описує характеристику, яку подає атрибут.

Атрибут – це будь-яка характеристика сутності, яка є істотною для аналізованої предметної області і призначена для кваліфікації, ідентифікації, класифікації, кількісної характеристики або відображення стану сутності. Атрибут відображає тип характеристик або властивостей, асоційованих із множиною реальних або абстрактних об'єктів (людей, місць, подій, станів, ідей, предметів тощо). Екземпляр атрибутів визначається типом характеристики і її значенням – значенням атрибута. У ERD атрибути асоціюються з конкретними сутностями. Отже, екземпляр сутності повинен мати єдине визначене значення для асоційованого атрибута.

Серед атрибутів виділяють один або декілька атрибутів – потенційний ключ, що однозначно ідентифікує кожний екземпляр сутності. Серед усіх потенційних ключів один позначається як первинний ключ, а інші – як альтернативні ключі. Переважно в сутностях існує один потенційний ключ, який автоматично є первинним.

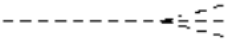
Атрибут може бути або обов'язковим, або необов'язковим. Обов'язковість означає, що атрибут не може отримувати невизначених значень (null-значень). Атрибут може бути або описовим, або входити до складу первинного ключа. На

рис. 1 зображено призначення ключових, обов’язкових та необов’язкових атрибутів сутності.

Взаємозв’язок (Relationship) – це іменована асоціація між двома сутностями, істотна для аналізованої предметної області. За наявності взаємозв’язку, як правило, кожен екземпляр однієї сутності, яка називається банківською сутністю, асоціюється з довільною (зокрема нульовою) кількістю екземплярів іншої сутності, яка називається сутністю-нащадком. А кожен екземпляр сутності-нащадка асоціюється точно з одним екземпляром батьківської сутності. Отже екземпляр сутності-нащадка може існувати тільки у разі існування відповідного екземпляра батьківської сутності.

Сутності позначаються прямокутниками, всередині яких наводиться список атрибутів; ключові атрибути – символом «#» (решітка); зв'язки – лініями з іменами, місце з'єднання зв'язку і сутності визначає кардинальність зв'язку: 0,1; 1,1; 0, N; 1, N (таблиця 1).

Таблиця 1 – Позначення зв’язків діаграм «сутність-зв’язок» у нотації Баркера

Позначення	Кардинальність
	0,1
	1,1
	0,N
	1,N

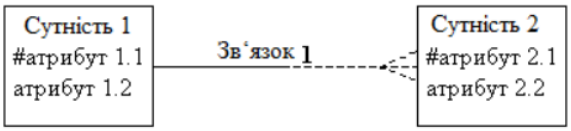


Рисунок 1 - Зображення зв'язків сутностей в нотації Баркера

Для позначення відношення категоризації вводиться елемент «дуга» (рисунок 2).

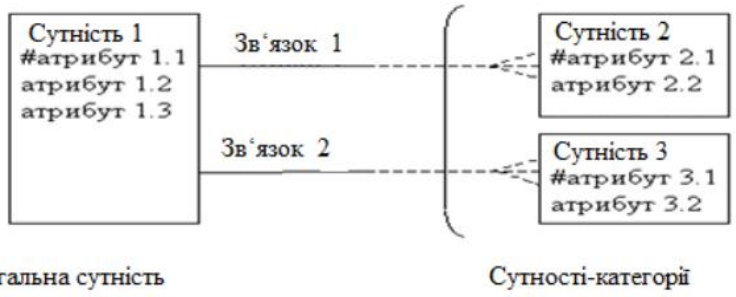


Рисунок 2 – Зображення зв'язків категоризації сутностей в нотації Баркера

Кожній сутності присвоюється унікальне ім'я і номер, що розділяються косою рисою "/" і розміщуються над блоком.

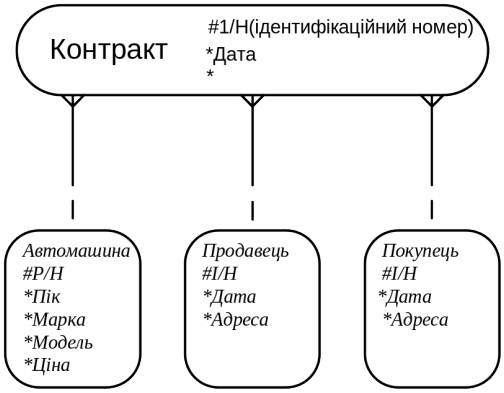


Рисунок 3 – оформлення сутностей

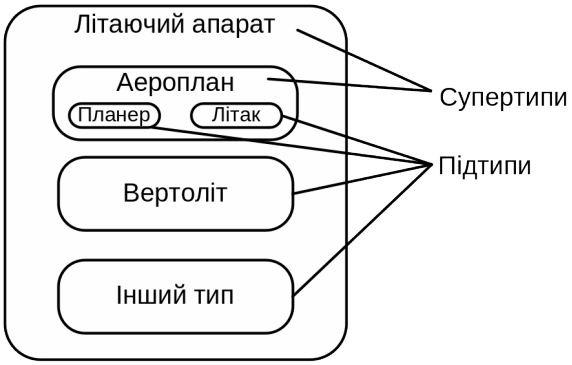


Рисунок 4 – Супертипи та підтипи

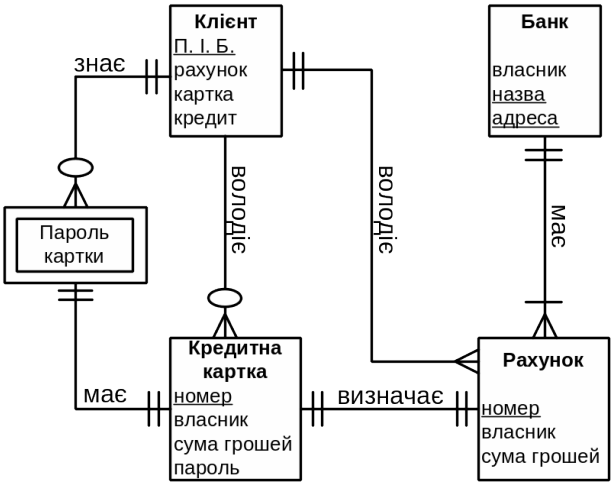


Рисунок 5 – приклад діаграми «сутність-зв’язок»

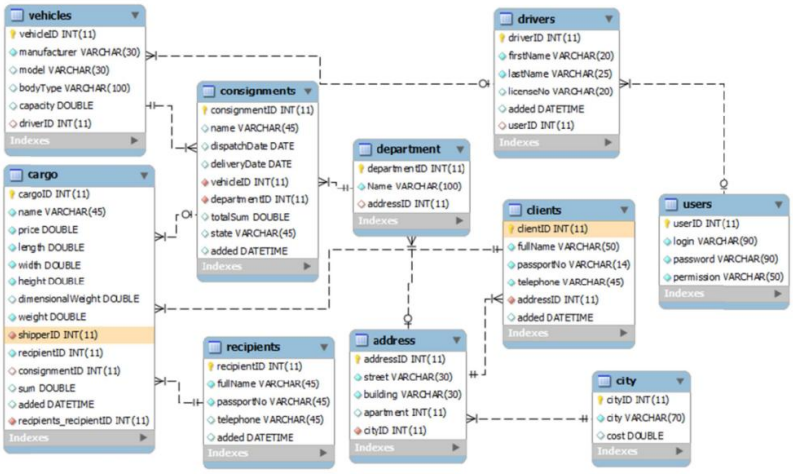


Рисунок 6 - приклад діаграми «сутність-зв’язок»

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області розглянутої в ПР 1 – 2 побудувати ER модель нотацією Баркера.
- Додати опис сутностей.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що таке модель «сутність зв’язок»
2. Опишіть графічну нотацію Баркера для побудови ERD.

Практична робота №4 Побудова ER-діаграми за нотацією П. Чена

Мета: Отримати практичні навики побудови діаграми «сутність-зв’язок» за нотацією П. Чена.

Теоретичні відомості:

Діаграми «сущность-зв'язок» (ERD) призначені для розробки моделей даних і забезпечують стандартний спосіб визначення даних і відносин між ними. За допомогою ERD-діаграми здійснюється деталізація сховища даних, моделюваної та проектованої шляхом ідентифікації та документування об’єктів (сутностей), систем для важливих предметних областей, властивостей цих об’єктів (атрибутів) та їх відносин з іншими об’єктами (зв’язками).

Нотація ERD-діаграм була запропонована Ченом. Нотація Чена надає багатий набір засобів моделювання даних, включаючи власні ERD-діаграми, а також діаграми атрибутів і діаграми категоризації. Ці діаграмні технології використовуються для моделювання та проектування як реляційних, так і ієрархічних та мережевих баз даних.

Сущність являє собою безліч реальних або абстрактних об'єктів (людей, подій, предметів, складів, ідей і т.п.), що володіють загальними характеристиками (атрибутами). Любий об'єкт системи може бути представлений тільки однією сутністю, яка повинна бути однозначно ідентифікована. При цьому імені сутності

слід відобразити тип або клас об’єкта, а не його конкретний екземпляр (наприклад, Місто, а не Одеса).

Відношення представляє собою зв'язок між двома і більш сутностями. Іменування відносин здійснюється за допомогою граматичного обороту глагола (Имеет, Определяет, Может владеть и т.п.).

Таблиця 1 – Позначення елементів діаграм «сутність-зв’язок» у нотації Чена

Елемент діаграми	Означення
Ім‘я	незалежна сутність
Ім‘я	залежна сутність
Ім‘я	батьківська сутність в ієрархічному зв'язку
Ім‘я	зв'язок
Ім‘я	ідентифікуючий зв'язок
Ім‘я	атрибут
Ім‘я	первинний ключ
Ім‘я	зовнішній ключ (поняття зовнішнього ключа вводиться в реляційній моделі даних)
Ім‘я	багатозначний атрибут
Ім‘я	одержуваний (успадкований) атрибут в ієрархічних зв'язках

Рисунок 1 – Зображення зв’язків сутностей в нотації Чена

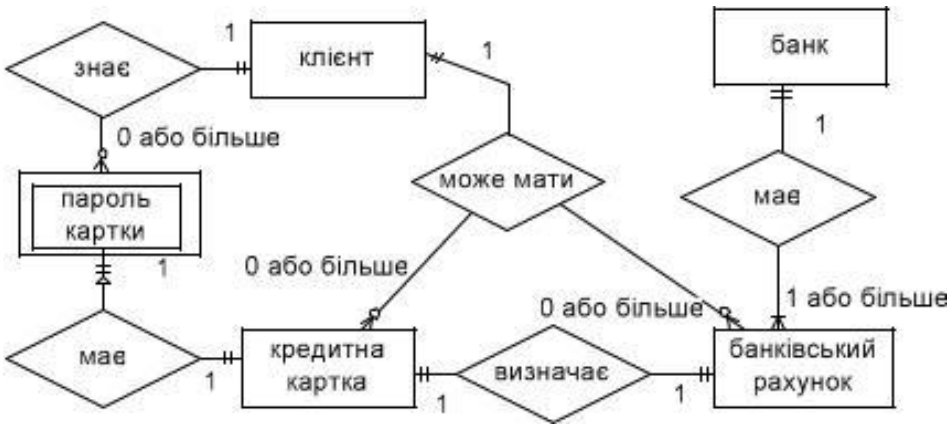


Рисунок 2 – приклад діаграми «сутність-зв’язок» нотацією Чена

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області розглянутої в ПР 1 – 2 побудувати ER модель нотацією Чена.
- Додати опис сутностей.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що таке модель «сутність-зв’язок»
2. Опишіть графічну нотацію П. Чена для побудови ERD.

Практична робота №5 Побудова SADT-блоків з інтерфейсними дугами

Мета: Отримати практичні навички виділення SADT-блоків з інтерфейс ними дугами.

Теоретичні відомості:

SADT (Технологія структурного аналізу і проектування) – це графічні позначення для опису систем.

Деякі області ефективного застосування SADT:

- Програмне забезпечення телефонних мереж
- Системна підтримка і діагностика
- Довгострокове і стратегічне планування
- Автоматизоване виробництво і проектування
- Конфігурація комп'ютерних систем
- Навчання персоналу
- Вбудоване програмне забезпечення оборонних систем
- Управління фінансами і матеріально-технічним постачанням

SADT – це методологія, розроблена спеціально для того, щоб полегшити опис і розуміння штучних систем, які потрапляють в розряд середньої складності.

SADT-модель – це опис системи. В SADT – моделях використовуються як природня так і графічна мови. Для передачі інформації про конкретну систему джерелом природньої мови служить людина, що описує систему, а джерелом графічної мови – сама методологія SADT.

Кожна SADT – діаграма містить блоки і дуги. Блоки зображують функції моделюємої системи. Дуги зв'язують блоки разом і відображають взаємодії і взаємозв'язки між ними.

Функціональні блоки на діаграмах зображуються прямокутниками. Блок представляє функцію або активну частину системи, наприклад, визначити ступінь виконання завдання, вибрати інструменти, підготовити робоче місце.

На відміну від інших графічних методів структурного аналізу в SADT кожна сторона блоку має особливе, цілком визначене призначення. Ліва сторона блоку призначена для входів, верхня – для управління, права – для виходів, нижня – для механізмів. Таке позначення відображає певні системні принципи: входи перетворюються у виходи, управління обмежує або указує умови виконання перетворень, механізми показують хто, що і як виконує функцію.

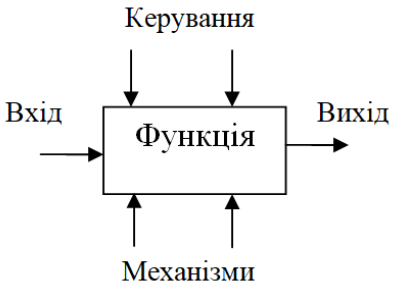


Рисунок 1 – нотація відображення функції

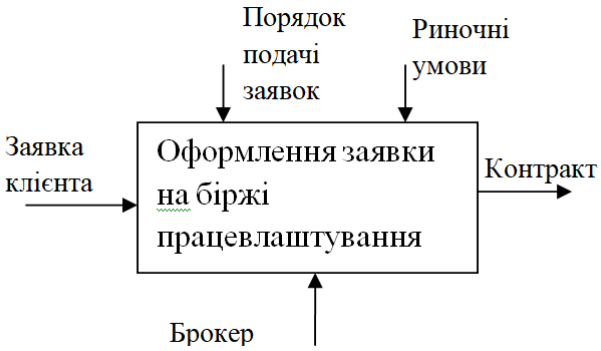


Рисунок 2 – приклад блоку

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області розглянутої в попередніх роботах побудувати SADT блоки (8-10 блоків).
- Додати до блоків інтерфейсі доги.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що таке методологія SADT?
2. З чого складаються SADT-моделі?
3. Що таке функціональні блоки та дуги? Яка нотація використовується?

Практична робота №6 Побудова ієрархій SADT-моделей

Мета: Отримати практичні навики побудови SADT-моделі та формування ієрархії діаграм

Теоретичні відомості:

SADT-модель – це опис системи. В SADT – моделях використовуються як природня так і графічна мови. Для передачі інформації про конкретну систему джерелом природньої мови служить людина, що описує систему, а джерелом графічної мови – сама методологія SADT.

З погляду SADT модель може бути зусереджена або на функціях системи, або на її об'єктах. SADT – моделі, орієнтовані на функції, прийнято називати функціональними моделями, а орієнтовані на об'єкти системи – моделями даних. Функціональна модель представляє з наобхідним ступенем деталізації систему функцій, які у свою чергу відображають свої взаємозв'язки через об'єкти системи.

При визначені моделі необхідно визначити позицію, з якої спостирігатеметься система, і створюватиметься її модель. Якість опису системи різко знижується, якщо вона не сфокусована ні на чому, SADT вимагає, щоб модель розглядалась весь час з однієї і тієїж позиції. Ця позиція називається *точкою зору* даної моделі.

Точку зору краще всього уявляти собі як місце (позиую) людини або об'єкту, в яке потрібно стати, щоб побачити систему в дії. З цієї фіксованої точки зору можна створити злагоджений опис системи так, щоб модель не дрейфувала навкруги, і в ній не змішувалися б незв'язані описи.

SADT – модель об'єднує і організовує діаграми в ієрархічні структури, в яких діаграми верхніх рівнів менш деталізовані ніж діаграми нижніх рвнів. Іншими словами, SADT – модель можна представити у вигляді деревовидної структури діаграм, де верхня діаграма є самою загальною, а самі нижні найбільш деталізовані.

Діаграма є основним робочим елементом при створені моделі. Розробник діаграм і моделей звичайно називається аналітиком, або, в термінології SADT автором. Діаграми мають власні синтаксичні правила відмінні від синтаксичних правил моделей.

Кожна SADT – діаграма містить блоки і дуги. *Блоки* зображують функції моделюємої системи. *Дуги* зв'язують блоки разом і відображають взаємодії і взаємозв'язки між ними.

Функціональні блоки на діаграмах зображуються прямокутниками. Блок представляє функцію або активну частину системи, наприклад, визначити ступінь виконання завдання, вибрати інструменти, підготовити робоче місце.

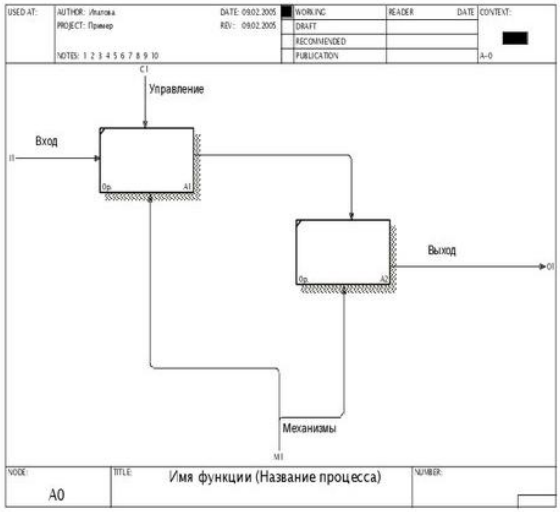
Методологія SADT є сукупністю методів, правил і процедур, призначених для побудови функціональної моделі об'єкту будь-якої предметної області. Функціональна модель SADT відображає функціональну структуру об'єкту, тобто

виділені нею дії і зв’язки між цими діями. Основні елементи цієї методології грунтуються на наступних концепціях:

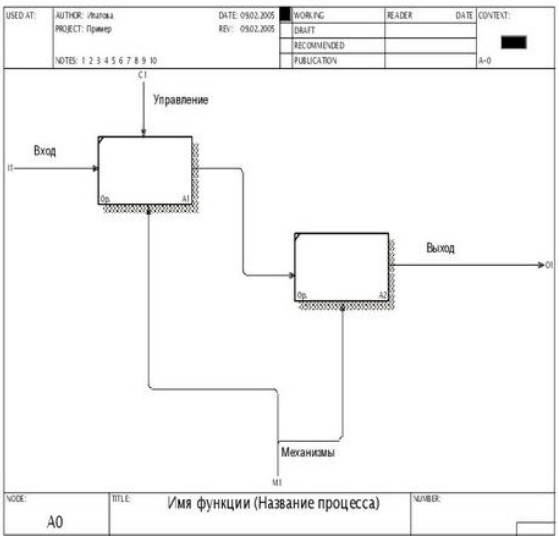
1. Графічне представлення блокового моделювання. Графіка блоків і дуг SADT – діаграми відображає функцію у вигляді блоку, а інтерфейси входу/виходу представляються дугами, які відповідно входять в блок і виходять з нього. Взаємодія блоків один з одним описується за допомогою інтерфейсних дуг виражаючих «обмеження», які у свою чергу визначають, коли і ким функції виконуються і управляються.
2. Концепція строгості і точності. Виконання правил SADT вимагає достатньої строгості і точності, не накладаючи при тому великих обмежень на дії аналітика. Правила SADT включають:
   - Обмеження кількості блоків на кожному рівні (3-6 блоків)
   - Зв’язаність діаграм (номери блоків)
   - Унікальність міток і найменувань (відсутність імен, що повторюються)
   - Синтаксичні правила для графіки блоків і дуг
   - Розділ входів і управління (правило визначення ролі даних)

Взаємовплив блоків може виражатися або в пересиланні Виходу до інший функції для подальшого перетворення, або в виробленні керуючою інформації, розпорядчої, що саме повинна робити інша функція. Існують п'ять типів взаємозв'язків між блоками для опису їх відносин:

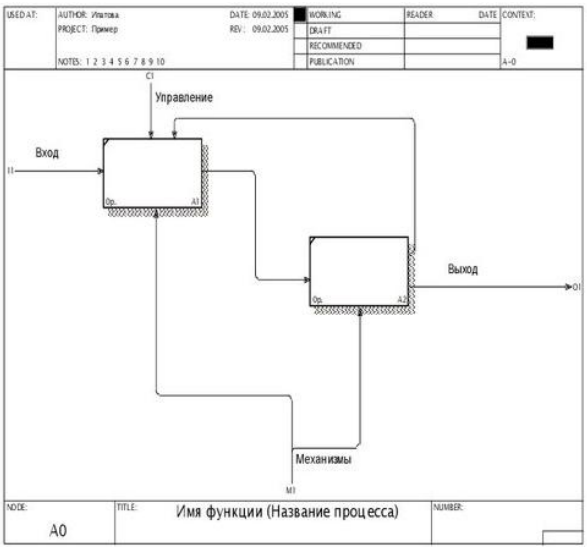
- управління,
- Вхід,
- Зворотній зв`язок по Управлінню,
- Зворотній зв`язок по Входу,
- Вихід-Механізм.



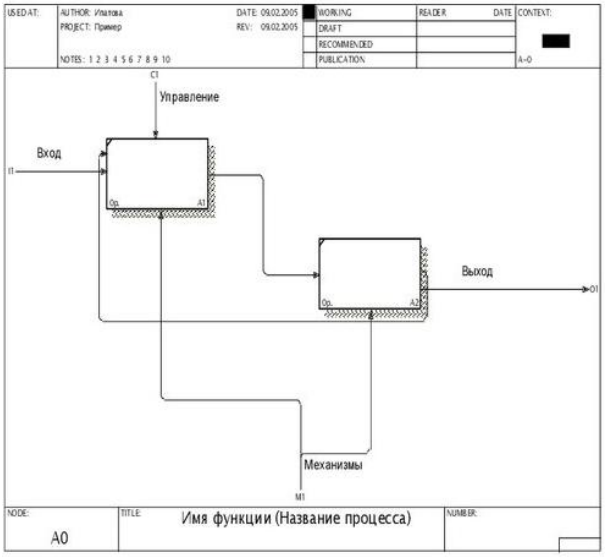
Відношення управління



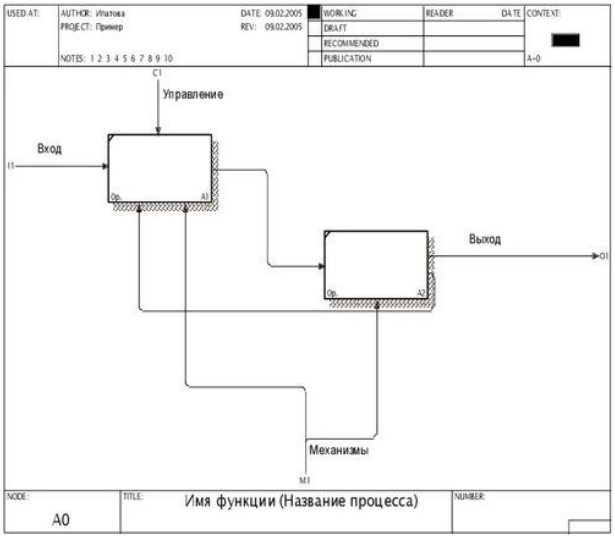
Відношення Вхід



Зворотній зв`язок по Управлінню

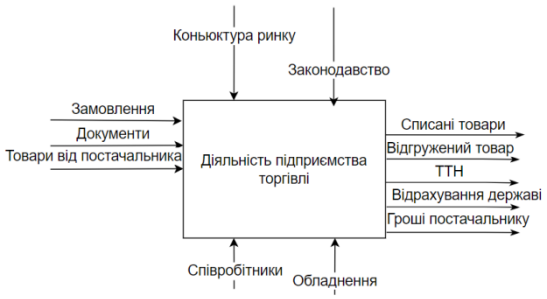


Зворотній зв`язок по Входу

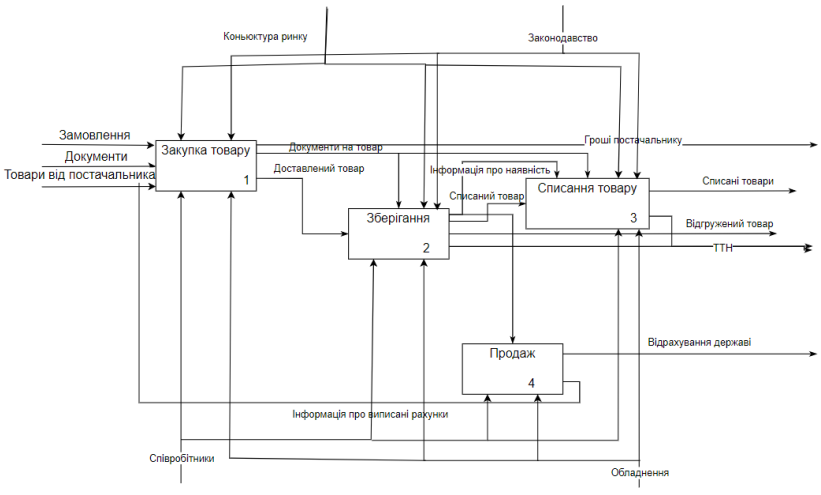


Вихід-Механізм

Приклад виконання роботи



А-0 Діяльність підприємства торгівлі



А-0 Діяльність підприємства торгівлі

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області розглянутої в попередніх роботах побудувати SADT-модель, що міститиме ієрархію діаграм.
- При побудові діаграм використовувати різні типи зв’язків.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. З чого складаються SADT- моделі?
2. Зв’язки яких типів можуть зустрічатись на діаграмах SADT?
3. Що таке точка зору?

Практична робота №7.1 Побудова початкової контекстної діаграми та матриці списку подій

Мета: Отримати практичні навики виконання етапу аналізу, побудови початкової контекстної діаграми нотацією Йордана-Де Марко та ELM.

Теоретичні відомості:

Увесь проект розділяється на 4 фази: аналіз, глобальне проектування (проектування архітектури системи), детальне проектування й реалізація (програмування).

На фазі аналізу будується модель середовища (Environmental Model). Побудова моделі середовища включає:

• аналіз поведінки системи (визначення призначення ИС, побудова початкової контекстної діаграми потоків даних (DFD) і формування матриці списку подій (ELM), побудова контекстних діаграм);

• аналіз даних (визначення складу потоків даних і побудова діаграм структур даних (DSD), конструювання глобальної моделі даних у вигляді Er- Діаграми).

Перед побудовою контекстної DFD необхідно проаналізувати зовнішні події (зовнішні об'єкти), що виявляють вплив на функціонування бібліотеки. Ці об'єкти взаємодіють із ИС шляхом інформаційного обміну з нею.

Приклад: У якості предметної області використовується опис роботи відеобібліотеки, яка отримує запити на фільми від клієнтів і стрічки, що повертаються клієнтами. Запити розглядаються адміністрацією відеобібліотеки з використанням інформації про клієнтів, фільми і стрічки. При цьому перевіряється й обновлюється список орендованих стрічок, а також перевіряються записи про членство в бібліотеці. Адміністрація контролює також повернення стрічок, використовуючи інформацію про фільми, стрічки і список орендованих стрічок, котрий обновляється. Опрацювання запитів на фільми і повернення стрічок включає такі дії: якщо клієнт не є членом бібліотеки, він не має права на оренду. Якщо необхідний фільм є в наявності, адміністрація інформує клієнта про орендну плату. Проте, якщо клієнт прострочив термін повернення наявних у нього стрічок, йому не дозволяється брати нові фільми. Коли стрічка повертається, адміністрація розраховує орендну плату плюс пеню за невчасне повернення.

Відеобібліотека отримує нові стрічки від своїх постачальників. Коли нові стрічки надходять у бібліотеку, необхідна інформація про неї фіксується. Інформація про членство в бібліотеці міститься окремо від записів про оренду стрічок.

Адміністрація бібліотеки регулярно робить звіти за визначений період часу про членів бібліотеки, постачальників стрічок, видачу визначених стрічок і стрічки, придбані бібліотекою.

З опису предметної області випливає, що в процесі роботи бібліотеки беруть участь наступні групи людей: клієнти, постачальники і керівництво. Ці групи є зовнішніми об'єктами. Вони не тільки взаємодіють із системою, але також визначають її межі і зображаються на початковій контекстній DFD як термінатори (зовнішні сутності).

Початкова контекстна діаграма зображена на рисунку 1.

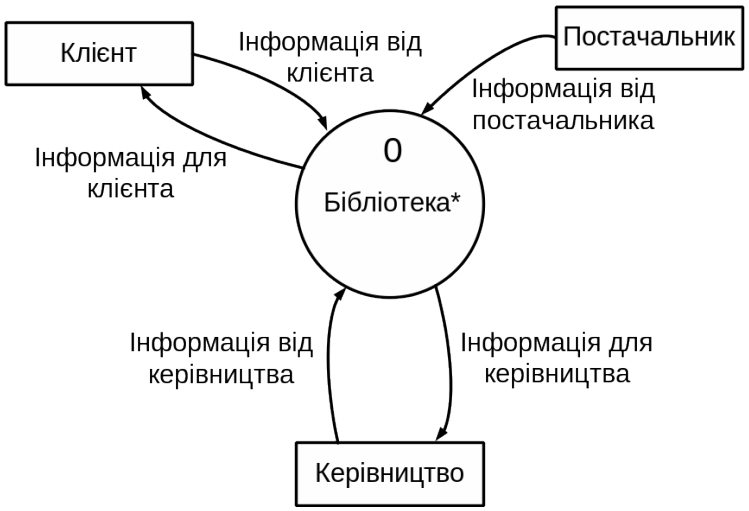


Рисунок 1 – початкова контекстна діаграма

Список подій будується у вигляді матриці (ELM) і описує різноманітні дії зовнішніх сутностей і реакцію ІС на них. Ці дії являють собою зовнішні події, що впливають на бібліотеку Розрізняють наступні типи подій:

Абревіатура	**Тип**
NC	Нормальне керування
ND	Нормальні дані
NCD	Нормальне керування/дані
TC	Тимчасове керування
TD	Тимчасові дані
TCD	Тимчасове керування/дані

Всі дії позначаються як нормальні дані. Ці дані є подіями, які ІС сприймає безпосередньо, наприклад, зміна адреси клієнта, що повинно бути відразу зареєстровано. Вони з'являються в DFD у якості вмісту потоків даних.

Матриця списку подій має такий вид:

№	Опис	Тип	Реакція
1	Клієнт бажає стати членом бібліотеки	ND	Реєстрація клієнта в якості члена бібліотеки
2	Клієнт повідомляє про зміну адреси	ND	Реєстрація зміненої адреси клієнта
3	Клієнт запитує оренду фільму	ND	Розгляд запиту
4	Клієнт повертає фільм	ND	Реєстрація повернення
5	Керівництво надає повноваження новому постачальнику	ND	Реєстрація постачальника
6	Постачальник повідомляє про зміну адреси	ND	Реєстрація зміненої адреси постачальника
7	Постачальник спрямовує фільм у бібліотеку	ND	Отримання нового фільму
8	Керівництво запитує новий фільм	ND	Формування необхідного запиту для керівництва

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Зробити опис предметної області.
- Для предметної області розглянутої в попередніх роботах побудувати початкову контекстну діаграму та матрицю списку подій.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що включає в себе модель середовища?
2. Як побудувати початкову контекстну діаграму?
3. Опишіть графічну нотацію Йордана-Де Марко?

Практична робота №7.2 Побудова контекстної діаграми, діаграми структур даних та ERD

Мета: Отримати практичні навики виконання етапу аналізу, побудови повної контекстної діаграми нотацією Йордана-Де Марко та діаграми структур даних.

Теоретичні відомості:

На приведеній DFD (рис. 1) накопичувач даних «бібліотека» є глобальним або абстрактним представленням сховища даних.

Аналіз функціонального аспекту поведінки системи дає уявлення про обмін і перетворення даних у системі. Взаємозв'язок між «абстрактними» потоками даних і «конкретними» потоками даних на діаграмі нульового рівня виражається в діаграмах структур даних (рисунок 3).

На фазі аналізу будується глобальна модель даних, яка подається у вигляді діаграми «сутність-зв'язок» .

Між різноманітними типами діаграм існують наступні взаємозв'язки:

ELM-DFD: події - вхідні потоки, реакції - вихідні потоки

DFD-DSD: потоки даних - структури даних верхнього рівня

DFD-ERD: накопичувані даних - ER- діаграми

DSD-ERD: структури даних нижнього рівня – атрибути сутностей.

Приклад: У якості предметної області використовується опис роботи відеобібліотеки, яка отримує запити на фільми від клієнтів і стрічки, що повертаються клієнтами. Запити розглядаються адміністрацією відеобібліотеки з використанням інформації про клієнтів, фільми і стрічки. При цьому перевіряється й обновлюється список орендованих стрічок, а також перевіряються записи про членство в бібліотеці. Адміністрація контролює також повернення стрічок, використовуючи інформацію про фільми, стрічки і список орендованих стрічок, котрий обновляється. Опрацювання запитів на фільми і повернення стрічок включає такі дії: якщо клієнт не є членом бібліотеки, він не має права на оренду. Якщо необхідний фільм є в наявності, адміністрація інформує клієнта про орендну плату. Проте, якщо клієнт прострочив термін повернення наявних у нього стрічок, йому не дозволяється брати нові фільми. Коли стрічка повертається, адміністрація розраховує орендну плату плюс пеню за невчасне повернення.

Відеобібліотека отримує нові стрічки від своїх постачальників. Коли нові стрічки надходять у бібліотеку, необхідна інформація про неї фіксується. Інформація про членство в бібліотеці міститься окремо від записів про оренду стрічок.

Адміністрація бібліотеки регулярно робить звіти за визначений період часу про членів бібліотеки, постачальників стрічок, видачу визначених стрічок і стрічки, придбані бібліотекою.

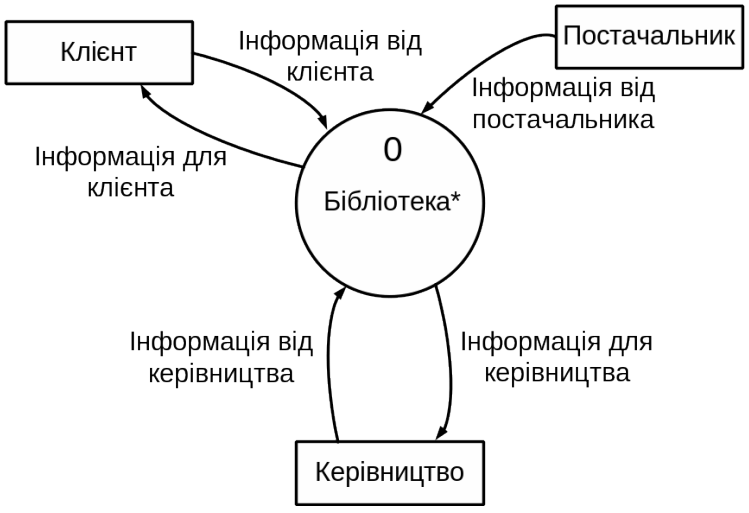


Рисунок 1 – початкова контекстна діаграма

Матриця списку подій мас такий вид:

№	Опис	Тип	Реакція
1	Клієнт бажає стати членом бібліотеки	ND	Реєстрація клієнта в якості члена бібліотеки
2	Клієнт повідомляє про зміну адреси	ND	Реєстрація зміненої адреси клієнта
3	Клієнт запитує оренду фільму	ND	Розгляд запиту
4	Клієнт повертає фільм	ND	Реєстрація повернення
5	Керівництво надає повноваження новому постачальнику	ND	Реєстрація постачальника
6	Постачальник повідомляє про зміну адреси	ND	Реєстрація зміненої адреси постачальника
7	Постачальник спрямовує фільм у бібліотеку	ND	Отримання нового фільму
8	Керівництво запитує новий фільм	ND	Формування необхідного запиту для керівництва

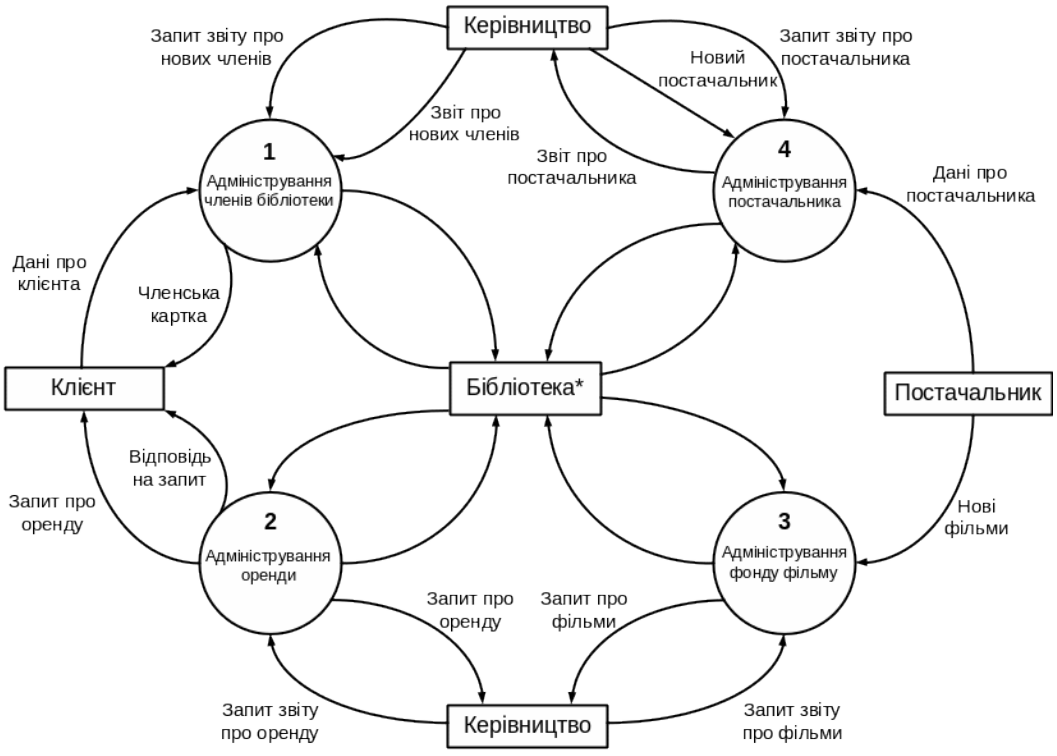


Рисунок 2 – контекстна діаграма першого рівня

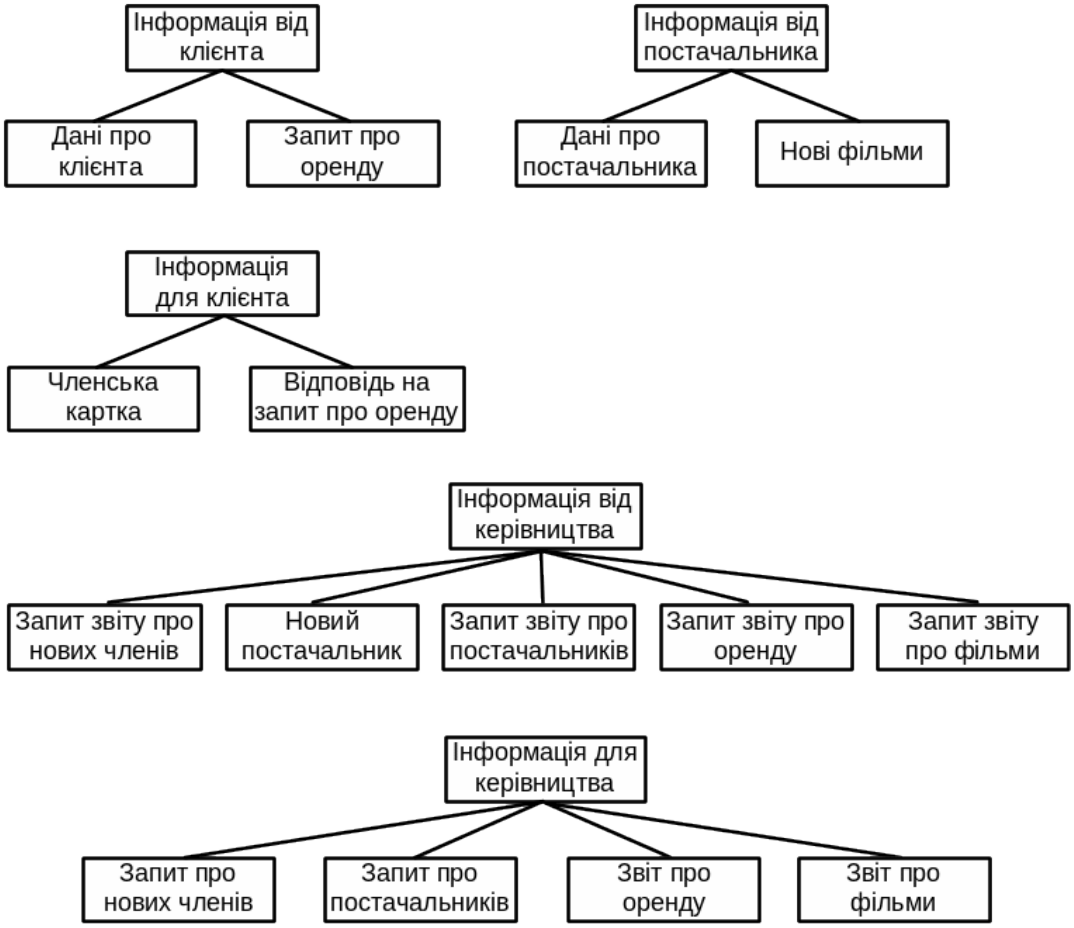


Рисунок 3 – Діаграма структур даних

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області з ПР№7.1 побудувати контекстні діаграми першого і другого рівня.
- Привести опис сховища даних.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Що включає в себе модель середовища?
2. Як побудувати декомпозиція контекстних діаграм?
3. Як пов’язані між собою різні діаграми моделі середовища?

Практична робота №8 Побудова DFD. Нотація Гейна Сарсона

Мета: Отримати практичні навики побудови діаграм потоків даних нотацією Гейна Сарсона

Теоретичні відомості:

Методологія структурного аналізу і проектування ІС визначає основні напрямки для оцінки і вибору проекту ІС, що розробляється, основні кроки, що повинні бути виконані, їхня послідовність, правила розподілу і призначення операцій і засобів.

Методологія Гейна-Сарсона ґрунтується на ідеї висхідної ієрархічної організації. Метою даної методології є перетворення загальних, нечітких знань про вимоги до системи в точні (наскільки це можливо) визначення. Дана методологія фокусує увагу на потоках даних, її головне призначення — створення в графіці документів за функціональними вимогами. Методологія підтримується традиційними висхідними засобами проектування специфікацій і забезпечує один з кращих засобів зв'язку між аналітиками, розробниками і користувачами системи. Відповідно до методології модель системи визначається як ієрархія діаграм потоків даних, що описують асинхронний процес перетворення інформації від її введення в систему до видачі користувачу. Діаграми верхніх рівнів ієрархії визначають основні процеси або підсистеми ІС із зовнішніми входами і виходами. Вони деталізуються за допомогою діаграм нижнього рівня. Така декомпозиція продовжується, створюючи багаторівневу ієрархію діаграм доти, поки не буде досягнуто такий рівень декомпозиції, при якому процеси стають елементарними і деталізувати їх далі неможливо.

Джерела інформації (зовнішні сутності) породжують інформаційні потоки (потоки даних), що переносять інформацію до підсистем або процесів. Ті у свою чергу перетворюють інформацію і породжують нові потоки, які переносять інформацію до інших процесів або підсистем, накопичувачам даних або зовнішнім сутностям - споживачам інформації. Таким чином, основними компонентами діаграм потоків даних є: зовнішні сутності; системи/підсистеми; процеси; накопичувачі даних; потоки даних.

Зовнішня сутність представляє собою матеріальний предмет або фізичну особу, що представляє собою джерело або приймач інформації, наприклад, замовники, персонал, постачальники, клієнти, склад. Визначення деякого об'єкта або системи в якості зовнішньої сутності вказує на те, що вона перебуває за межами кордонів аналізованої ІС. У процесі аналізу деякі зовнішні сутності можуть бути перенесені усередину діаграми аналізованої ІС, якщо це необхідно, або, навпаки, частина процесів ІС може бути винесена за межі діаграми і подана як зовнішня сутність.

Зовнішня сутність позначається квадратом (рисунок 1), розташованим як би «над» діаграмою із тінню, для того, щоб можна було виділити цей символ серед інших позначень:

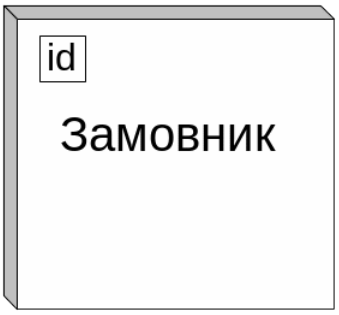


Рисунок 1 – Зовнішня сутність

При побудові моделі складної ІС вона може бути подана в самому загальному виді на так називаній контекстній діаграмі у вигляді однієї системи як єдиного цілого, або може бути декомпозійована на ряд підсистем.

Підсистема (або система) на контекстній діаграмі зображається в такий спосіб (рисунок 2).

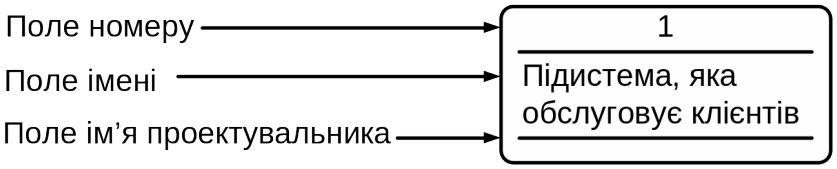


Рисунок 2 – підсистема

Номер підсистеми служить для їх ідентифікації. У поле імені вводиться найменування підсистеми у вигляді речення з підметом і відповідними визначеннями і доповненнями.

Процес являє собою перетворення вхідних потоків даних на вихідні відповідно до визначеного алгоритму. Фізично процес "може бути реалізований різноманітними засобами: це може бути підрозділ організації (відділ), що виконує опрацювання вхідних документів і випуск звітів, програма, апаратно реалізований логічний пристрій і т.д.

Процес на діаграмі потоків даних зображається, як показано на рисунку 3.

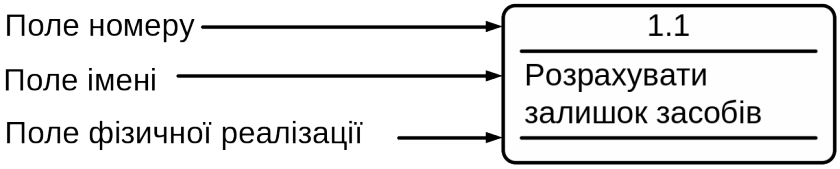


Рисунок 3 – процес

Номер процесу служить для його ідентифікації. У поле імені водиться найменування процесу у вигляді речення з активним недвозначним дієсловом у невизначеній формі (обчислити, розрахувати, перевірити, визначити, створити, одержати), за яким слідують іменники в знахідному відмінку, наприклад:

- «Ввести відомості про клієнтів»;
- «Видати інформацію про поточні витрати»;
- «Перевірити кредитоспроможність клієнта».

Використання таких дієслів, як «опрацювати», «модернізувати» або «відредагувати» означає, як правило, недостатньо глибоке розуміння даного процесу і потребує подальшого аналізу.

Інформація в полі фізичної реалізації показує, який підрозділ організації, програма або апаратний пристрій виконує даний процес.

Накопичувач даних являє собою абстрактний пристрій для збереження інформації, який можна в будь-який момент помістити в накопичувач і через якийсь час витягти, причому засоби поміщення і витягу можуть бути будь-якими.

Накопичувач даних може бути реалізований фізично у вигляді ящика в картотеці, таблиці в оперативній пам'яті, файл на магнітному носії і т. д. Накопичувач даних на діаграмі потоків даних зображається, як показано на рисунку 4.

Рисунок 4 – Накопичувач даних

Накопичувач даних ідентифікується буквою «D» і довільним числом. Ім'я накопичувача вибирається з погляду найбільшої інформативності для проектувальника.

Накопичувач даних у загальному випадку є прообразом майбутньої бази даних і опис даних які в ньому зберігаються, повинен бути пов'язаний з інформаційною моделлю.

Потік даних визначає інформацію, передану через деяке з'єднання від джерела до приймача. Реальний потік даних може бути інформацією, переданою по кабелю між двома пристроями, магнітними стрічками або дискетами і т. д.

Потік даних на діаграмі зображається лінією, що закінчується стрілкою, котра показує напрямок потоку (рисунок 5). Кожний потік даних має ім'я, яке відображає його зміст.

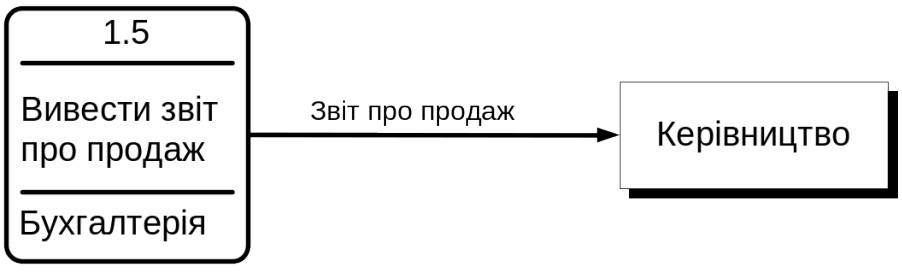


Рисунок 5 – потік даних

Першим кроком при побудові ієрархії DFD є побудова контекстних діаграм. Звичайно при проектуванні відносно простих ІС будується єдина контекстна діаграма з зіркоподібною топологією, у центрі якої знаходиться так називаний головний процес, сполучений із приймачами і джерелами інформації, за допомогою яких із системою взаємодіють користувачі й інші зовнішні системи.

Якщо ж для складної системи обмежитися єдиною контекстною діаграмою, то вона буде містити занадто велику кількість джерел і приймачів інформації, які

важко розташувати на листі паперу нормального формату, і крім того, єдиний головний процес не розкриває структури розподіленої системи. Ознаками складності (у змісті контексту) можуть бути:

- наявність великої кількості зовнішніх сутностей (десять і більше);
- розподілена природа системи;
- багатофункціональність системи з вже сформованим або виявленим групуванням функцій в окремі підсистеми.

Для складних ІС будується ієрархія контекстних діаграм. При цьому контекстна діаграма верхнього рівня містить не єдиний головний процес, а набір підсистем, сполучених потоками даних. Контекстні діаграми такого рівня деталізують контекст і структуру підсистем.

Ієрархія контекстних діаграм визначає взаємодію основних функціональних підсистем проектованої ІС як між собою, так і з зовнішніми вхідними і вихідними потоками даних і зовнішніми об'єктами (джерелами і приймачами інформації), із якими взаємодіє ІС.

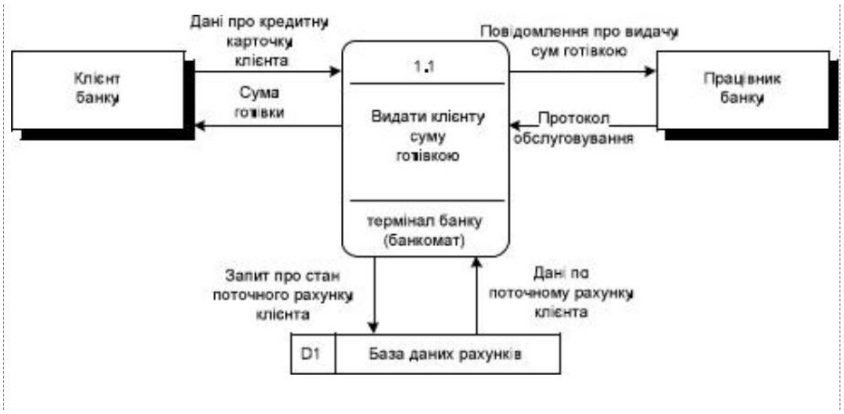


Рисунок 6 – Приклад DFD для процесу отримання деякої суми готівки з кредитної карти

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області розробити ієрархію діаграм потоків даних використовуючи нотацію Гейна Сарсона.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)

- Список використаної літератури

Контрольні питання:

1.Призначення діаграм потоків даних.

2.Основні елементи діаграм потоків даних.

3.Особливості побудови діаграм потоків даних.

4.Нотації діаграм потоків даних.

Практична робота №9 Побудова діаграми в нотації IDEF3

Мета: Отримати практичні навики моделювання бізнес процесів.

Теоретичні відомості:

IDEF3 - це метод, що має основною метою дати можливість аналітикам описати ситуацію, коли процеси виконуються в певній послідовності, а також описати об'єкти, що беруть участь спільно в одному процесі.

Техніка опису набору даних IDEF3 є частиною структурного аналізу. На відміну від деяких методик описів процесів IDEF3 не обмежує аналітика надмірно твердими рамками синтаксису, що може привести до створення неповних або суперечливих моделей.

IDEF3 може бути також використаний як метод створення процесів.

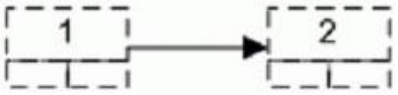
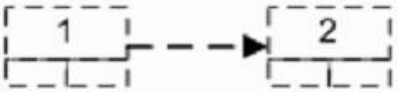
Кожна робота в IDEF3 описує який-небудь сценарій бізнес- процесу й може бути складовою іншої роботи. Оскільки сценарій описує мету й рамки моделі, важливо, щоб роботи йменувалися віддієслівним іменником, що позначають процес дії, або фразою, що містить такий іменник.

Точка зору на модель повинна бути документована. Звичайно це точка зору людину, відповідального за роботу в цілому. Також необхідно документувати мету моделі - ті питання, на які покликано відповісти модель.

Діаграма є основною одиницею опису в IDEF3. Важливо правильно побудувати діаграми, оскільки вони призначені для читання іншими людьми (а не тільки автором).

Одиниці роботи - Unit of Work (UOW) - також називані роботами (activity), є центральними компонентами моделі. В IDEF3 роботи зображуються прямокутниками із прямими кутами й мають ім'я, що позначає процес дії і номер (ідентифікатор); інше ім'я відображає основний вихід (результат) роботи (наприклад, "Виготовлення виробу"). Часте ім'я міняється в процесі моделювання, оскільки модель може уточнюватися й редагуватися. Ідентифікатор роботи привласнюється при створенні й не міняється ніколи. Навіть якщо робота буде вилучена, її ідентифікатор не буде знову використовуватися для інших робіт. Звичайно номер роботи складається з номера батьківської роботи й порядкового номера на поточній діаграмі.

Зв'язки показують взаємини робіт. Усі зв'язки в IDEF3 односпрямовані й можуть бути спрямовані куди завгодно, але звичайно діаграми IDEF3 намагаються побудувати так, щоб зв'язки були спрямований ліворуч праворуч. В IDEF3 розрізняють три типи стрілок, що зображують зв'язки, стиль яких установлюється через меню Edit/Arrow Style:

Назва звя’зку	Вид звя’зку	Смисл звя’зку
Старший зв’язок		Означає, що друга робота почне виконуватись після завершення першої роботи.
Зв’язок відносини		Означає, що друга робота може початися і даже закінчитися до того моменту, як закінчиться виконання першої роботи.
Зв’язок потоки об’єктів		Одночасно означає часову послідовність робіт й матеріальний або інформаційний потік. В даному випадку друга робота почне виконуватися після завершення першої роботи. При чому виходом першої роботи є об’єкт назва якого написана над стрілкою (в даному випадку це документ). Цей зв’язок також означає, що об’єкт породжуємий першою роботою використовується в наступних роботах

Крім наявності декількох типів зв'язків між роботами в стандарті IDEF3 використовуються логічні оператори, які в цьому випадку називаються **перехрестями** також діляться на кілька типів: "ЩоВиключає АБО", "И" і "АБО".

Перехрестя "ЩоВиключає АБО" позначає, що після завершення роботи "А" (рисунок 1), починає виконуватися тільки одна із трьох розташованих паралельно робіт B, З або D залежно від умов 1, 2 і 3. Перехрестя "И" позначає, що після завершення роботи "А", починають виконуватися одночасно три паралельно

розташовані роботи B, C и D. Перехрестя "АБО" позначає, що після завершення роботи "A", може запуститися будь-яка комбінація трьох паралельно розташованих робіт B, C и D. Наприклад може запуститися тільки одна з них, можуть запуститися три роботи, а також можуть запуститися подвійні комбінації B и C, або C і D, або B і D. Перехрестя ", ЩоВиключає АБО" є самим невизначеним, тому що припускає кілька можливих сценаріїв реалізації бізнес- процесу й застосовується для опису слабко формалізованих ситуацій.

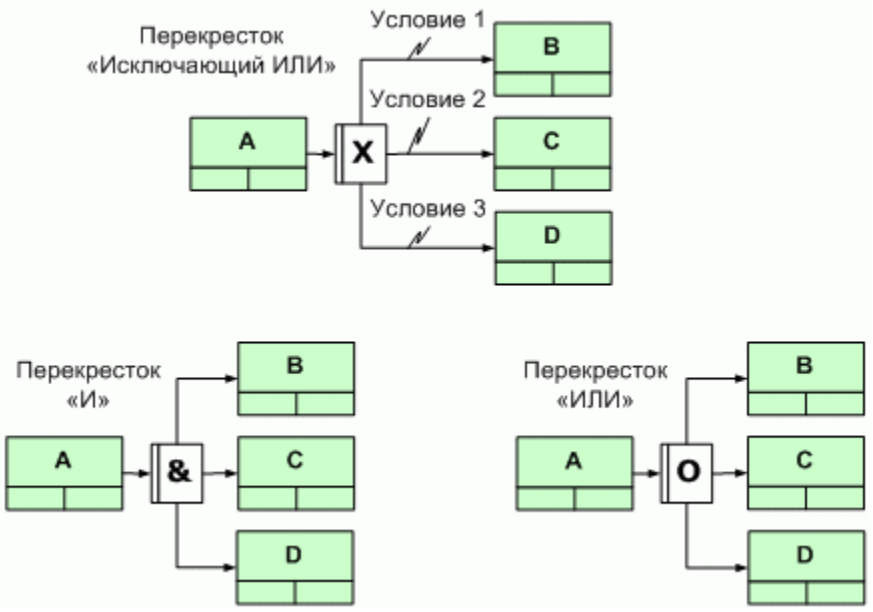


Рисунок 1 – Типи перехресть

Перехрестя "И" і "АБО" підрозділяються ще на два підтипи - синхронні й асинхронні. Перехрестя **синхронного типу** позначають, що роботи B, C и D запускаються одночасно після завершення роботи A. Перехрестя **асинхронного типу** вимог до одночасності не пред'являють.

Наведені на рисунку 2 схеми взаємозв'язку робіт і перехресть називаються схемами **розбіжності**, тому що від перехресть розходяться кілька робіт. Існує й інші схеми взаємозв'язку перехресть і робіт - це так звані схеми **сходження**, коли до перехрестя підходить кілька робіт.

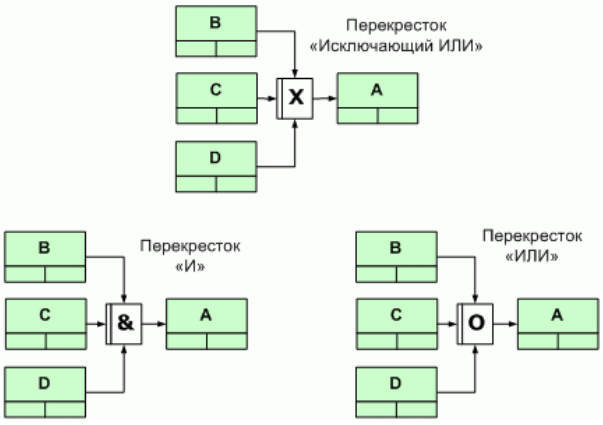


Рисунок 2 – Схеми сходження

Приклад: *Деталь надходить у фарбувальний цех, підготовленої до фарбування. У процесі фарбування наноситься один шар емалі при високій температурі. Після цього, проводиться сушіння деталі, після якого починається етап перевірки якості нанесеного шару. Якщо тест підтверджує недостатня якість нанесеного шару (недостатню товщину, неоднорідність і т.д.), то деталь заново пропускається через цех фарбування. Якщо деталь успішно проходить контроль якості, то вона відправляється в наступний цех для подальшої обробки.*

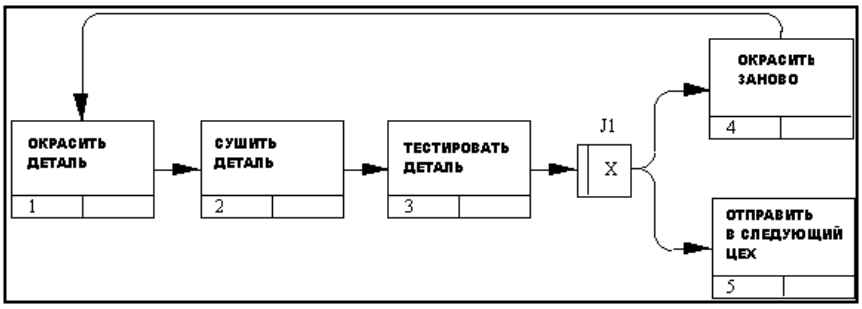


Рисунок 3 – Приклад діаграми PFDD

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Для предметної області розробити діаграму PFDD.
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Для чого використовують методологію IDEF-3?
2. Як відображаються процеси?
3. Що таке перехрестя і для чого використовуються?
4. Які відношення використовуються в PFDD?

Практична робота №10 Розробка тест-кейсів

Мета: Отримати практичні навики створення та проведення кейс-тестів.

Теоретичні відомості:

ТЕСТ КЕЙС – це професійна документація тестувальника, послідовність дій спрямована на перевірку будьякого функціоналу, що описує як дійти фактичного результату.

Набір тест-кейсів називають тест-комплектом. Іноді тестнабір плутають із тест-планом. Тест-план визначає які роботи, як і коли повинні бути проведені в рамках тестування продукту, а так само, що необхідно для їх виконання.

Тест-кейси мають допомогти нам провести перевірку продукту без ознайомлення з усією документацією. Написаний один раз, зручний у підтримці тесткейс заощадить багато часу та сил тестувальникам.

АТРИБУТИ ТЕСТ-КЕЙСУ

Унікальний ідентифікатор тест-кейсу – необхідний для зручної організації зберігання та навігації за нашими тест-наборами.

Назва – основна тема, або ідея тест-кейсу. Кратний опис його суті.

Передумови – опис умов, які не мають прямого відношення до функціоналу, що перевіряється, але повинні бути виконані. Наприклад, залишити коментар на вашому порталі може лише зареєстрований користувач. Значить для тест-кейсу «Створення коментаря» буде необхідно виконання передумови «користувач зареєстрований» , та «користувач авторизований».

Кроки – опис послідовності дій, яка повинна привести нас до очікуваного результату.

Очікуваний результат – результат: що ми очікуємо побачити після виконання кроків.

Кожен виконаний тест-кейс, дає нам один із трьох результатів:

1. Позитивний результат, якщо фактичний результат дорівнює очікуваному результату.

2. Негативний результат, якщо фактичний результат не дорівнює очікуваному результату. У цьому випадку знайдено помилку.

3. Виконання тесту блоковано, якщо після одного з кроків продовження тесту неможливе. У цьому випадку так само знайдено помилку.

ПРИКЛАД №1 Тест-кейс №1. Створення мешканця без ПІБ.

Кроки:

1. Зайти на сайт www.dev_test.ua (логін – test, пароль – test).
2. Увійти під обліком адміністратора (логін - admin, пароль - 1)
3. Перейти на вкладку "Мешканці".
4. Натиснути кнопку "Створити картку мешканця".
5. Натиснути кнопку "Зберегти" , не заповнюючи жодних даних.

Очікуваний результат З'являється повідомлення про помилку "Заповніть обов'язкові поля, позначені *" , картка не зберігається.

ПРИКЛАД №2 (ТЕСТ ДЕКІЛЬКОМА ОЧІКУЄМИМИ РЕЗУЛЬТАТАМИ)
Декілька варіантів даних, що вводяться

Тест-кейс №2. Створення мешканця, перевірка поля "ПІБ".

Кроки:

1. Зайти на сайт www.dev_test.ua (логін – test, пароль – test).
2. Увійти під обліком адміністратора (логін - admin, , пароль - 1)
3. Перейти на вкладку "Мешканці" Натиснути кнопку "
4. Створити картку мешканця".
5. Заповнити поле ПІБ (див. "Очікуваний результат")
6. Натиснути кнопку "Зберегти".

Таблиця 1 – очікувані результати

Значення що вводиться	Очікуваний результат
Кісільова Ольга Євгенівна	ОК, картка зберігається
"Залишити поле порожнім"	Помилка, "Заповніть обов'язкові поля, позначені *", картка не зберігається.
2*4*6*8*11*14*17*20*23*26*29*32*35*38*41*	Помилка – «Максимальна довжина поля – 40 символів, введено – 41», картка не зберігається.
Kiseleva Olga Evgenievna	Помилка – «Поле ПІБ може містити лише літери українського алфавіту», картка не зберігається
...	...

Порядок виконання роботи

- Вивчити пропонований теоретичний матеріал.
- Візьміть свій курсовий проєкт з БД.
- Оберіть 2-3 функції для тестування.
- Складіть по 2 тести для кожної функції
- Скласти звіт.

Зміст звіту

У звіті необхідно вказати:

- Мету роботи
- Відповідь на контрольні питання
- Основну частину (опис самої роботи), виконану відповідно до вимог щодо результатів виконання практичної роботи.
- Висновок (висновки)
- Список використаної літератури

Контрольні питання:

1. Для чого використовують тест-кейси?
2. З чого склаються тест-кейси?
3. Чого не повинно бути в тест-кейсах?

Список використаних джерел інформації

1) І. Бородкіна, Г. Бородкін «Інженерія програмного забезпечення. Навчальний посібник», Центр учбової літератури, 2021 – 204 с.;

2) Р. Мартін «Чиста архітектура», Фабула, 2019 – 416 с.;

3) Ю. Рамський «Проектування й опрацювання баз даних: Посібник для вчителів», Навчальна книга Богдан, 416 с;

4) Ерік Еванс «Предметно-орієнтоване проектування (DDD): структуризація складних програмних систем», Діалектика, 2016 – 448 с..

5) Ю. Грицюк «Аналіз вимог до програмного забезпечення», Львівська Політехніка, 2018 – 456 с.

6) О. Перевозчикова «Інформаційні системи і структури даних», Києво-Могилянська академія, 2007 – 288с.

Додаток 1 Предметні області для лабораторних робіт

1. Розробити програмний модуль "Облік успішності студентів". Програмний модуль призначений для оперативного обліку успішності студентів у сесію деканом, заступниками декана і співробітниками деканату. Зведення про успішність студентів повинні зберігатися протягом усього терміну їхнього навчання і використовуватися при складанні довідок про прослухані курси і додатків до диплома.

2. Розробити програмний модуль "Особисті справи студентів". Програмний модуль призначений для одержання відомостей про студентів співробітниками деканату, профкому і відділу кадрів. Відомості повинні зберігатися протягом усього терміну навчання студентів і використовуватися при складанні довідок і звітів.

3. Розробити програмний модуль "Рішення комбінаторно-оптимізаційних задач". Модуль повинний містити алгоритми пошуку циклу мінімальної довжини (задача комівояжера), пошуку найкоротшого шляху і пошуку мінімального єднального (такого, що зв’язує) дерева.

4. Розробити додаток Windows "Органайзер".

Додаток призначений для запису, збереження і пошуку адрес і телефонів фізичних осіб і організацій, а також розкладу, зустрічей т. і. Додаток призначений для будь-яких користувачів комп'ютера.

5. Розробити додаток Windows "Калькулятор".

Додаток призначений для будь-яких користувачів і повинний містити всі арифметичні операції (з дотриманням пріоритетів) і бажано (але не обов'язково) кілька математичних функцій.

6. Розробити програмний модуль "Відділення", який містить відомості про співробітників відділення (ФИО, посада, вчена ступінь, дисципліни, навантаження, суспільний робота, сумісництво т. і.). Модуль призначений для використання співробітниками відділу кадрів і адміністрації.

7. Розробити програмний модуль "Лабораторія", що містить відомості про співробітників лабораторії (ФИО, підлога, вік, родиний стан, наявність дітей, посада, вчена ступінь). Модуль призначений для використання співробітниками профкому і відділу кадрів.

8. Розробити програмний модуль "Автосервіс". При записі на обслуговування заповнюється заявка, у якій указуються ФИО власника, марка автомобіля, вид роботи, дата прийому замовлення і вартість ремонту. Після виконання робіт роздруковується квитанція.

9. Розробити програмний модуль "Облік порушень правил дорожнього руху". Для кожної автомашини (і її власника) у базі зберігається список порушень. Для кожного порушення фіксується дата, час, вид порушення і розмір штрафу. При оплаті всіх штрафів машина відаляється з бази.

10. Розробити програмний модуль "Картотека агентства нерухомості", призначений для використання працівниками агентства. У базі містяться відомості

про квартири (кількість кімнат, поверх, метраж і ін.). При надходженні заявки на обмін (куплю, продаж) виробляється пошук придатного варіанта. Якщо такого немає, клієнт заноситься в клієнтську базу і оповіщається, коли варіант з'являється.

11. Розробити програмний модуль "Картотека абонентів АТС". Картотека містить зведення про телефони і їхніх власників. Фіксує заборгованості по оплаті (абонентській і погодинній). Вважається, що погодинна оплата місцевих телефонних розмов уже введена.

12. Розробити програмний модуль "Авіакаса", що містить відомості про наявність вільних місць на авіа маршрути. У базі повинні міститися відомості про номер рейса, екіпажі, типі літака, даті і часу вильоту, а також вартості авіаквитків (різного класу). При надходженні заявки на квитки програма робить пошук придатного рейса.

13. Розробити програмний модуль "Книгарня", що містить відомості про книги (автор, назва, видавництво, рік видання, ціна). Покупець оформляє заявку на потрібні йому книги, якщо таких немає, він заноситься в базу й оповіщається, коли потрібні книги надходять у магазин.

14. Розробити програмний модуль "Автостоянка". У програмі міститься інформація про марку автомобіля, його власнику, даті і часі в'їзду, вартості стоянки, знижках, заборгованості по оплаті т.і.

15. Розробити програмний модуль "Кадрове агентство", що містить відомості про вакансії і резюме. Програмний модуль призначений як для пошуку співробітника, що відповідає вимогам керівників фірми, так і для пошуку придатної роботи.

Примітка. При розробці програми не обмежуватися функціями, приведеними у варіанті, додати кілька своїх функцій. Використовуйте структурний, модульний або об'єктний підходи до програмування.