**МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ**

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Методичні рекомендації до виконання дипломного проекту освітньо-кваліфікаційного рівня "бакалавр"

**для студентів напряму підготовки 6.050101 "Комп'ютерні науки"
всіх форм навчання**

Харків. Вид. ХНЕУ, 2012

Затверджено на засіданні кафедри інформаційних систем.
Протокол № 2 від 14.09.2011 р.

Укладачі: Золотарьова І. О.
Парфьонов Ю. Е.
Ушакова І. О.

М54 Методичні рекомендації до виконання дипломного проекту освітньо-кваліфікаційного рівня "бакалавр" для студентів напряму підготовки 6.050101 "Комп'ютерні науки" всіх форм навчання / укл. Золотарьова І. О., Парфьонов Ю. Е., Ушакова І. О., – Х. : Вид. ХНЕУ, 2012. – 80 с. (Укр. мов.)

Подано вимоги щодо організації, тематики, змісту та оформлення дипломних проектів. Розроблено з урахуванням галузевого стандарту вищої освіти України з напряму підготовки "Комп'ютерні науки" і досвіду кафедри інформаційних систем з підготовки бакалаврів відповідного напряму.

Рекомендовано для студентів напряму підготовки 6.050101 "Комп'ютерні науки".

Вступ

Дипломне проектування є одним з найважливіших видів самостійної роботи, яка завершує підготовку студентів за програмою бакалавра напряму підготовки "Комп'ютерні науки", а також – основою для проведення державної атестації бакалаврів.

Головним завданням дипломного проектування є підготовка студента до виконання завдань та обов'язків, що передбачені для первинних посад у певному виді економічної діяльності.

Дипломний проект бакалавра може бути початковим етапом виконання дипломного проекту спеціаліста або магістра.

У методичних рекомендаціях викладені загальні вимоги до організації та проведення дипломного проектування, змісту, структури та обсягу дипломних проектів, організації роботи над проектом, оформлення та захисту дипломного проекту.

Зміст, обсяг та структура дипломного проекту, які наведені у методичних рекомендаціях, є типовими і у окремих випадках за письмовим дозволом випускаючої кафедри можуть бути змінені.

Методичні рекомендації призначені для студентів, що навчаються за напрямом підготовки 6.050101 "Комп'ютерні науки".

Вимоги до дипломного проектування повністю відповідають освітньо-кваліфікаційній характеристиці і освітньо-професійній програмі напряму підготовки 6.050101 "Комп'ютерні науки".

1. Мета й завдання дипломного проекту

Дипломне проектування – завершальний етап підготовки бакалаврів.

Дипломний проект – це кваліфікаційна робота, покликана здійснювати об'єктивний контроль за ступенем сформованості вмінь у випускників вирішувати типові завдання діяльності, які належать до проектувальної (проектно-конструкторської) та виконавської (технологічної, технічної) виробничих функцій [3].

Мета дипломного проектування – узагальнити та систематизувати знання та практичні навички студентів, які одержані ними під час вивчення навчальних дисциплін гуманітарної та соціально-економічної підготовки; математичної та природничо-наукової підготовки; професійної та практичної підготовки. У процесі роботи над дипломним проектом

студенти набувають навички з аналізу науково-технічної, нормативної та довідкової літератури, використання державних стандартів, складання пояснювальної записки до проекту, практичного застосування знань під час прийняття конкретних проектних рішень.

Задачами дипломного проектування є:

систематизація, закріплення та розширення теоретичних знань і практичних навичок за напрямом підготовки, застосування цих знань та навичок при виконанні конкретних завдань дипломного проекту;

розвиток та закріплення навичок самостійної роботи;

удосконалення вміння користуватись сучасними системами програмування, вирішувати інженерні задачі з проектування інформаційних систем та їх елементів, використовуючи сучасні методології, інформаційні технології, проводити комп'ютерне моделювання, а також вміння обробляти і систематизувати результати досліджень, використовуючи комп'ютерну техніку та відповідні інструментальні засоби;

визначення відповідності рівня підготовки випускника вимогам освітньо-кваліфікаційної характеристики бакалавра, його готовності та спроможності до самостійної роботи в умовах ринкової економіки, сучасного виробництва, прогресу науки та техніки.

Виконуючи дипломний проект, студент повинен повною мірою використовувати набуті знання з інформаційних технологій та комп'ютерної техніки, інтелектуальних систем та баз знань, існуючі пакети, методи та засоби математичної обробки інформації; поєднувати теоретичні знання з виробничим досвідом, отриманим при проходженні практики; використовувати досягнення вітчизняної та світової науки і техніки; враховувати техніко-економічні показники функціонування створюваних програмно-інформаційних систем та комплексів; на високому теоретичному і професійному рівні виконувати проектування обраних технічних рішень; грамотно, повно і разом з тим лаконічно викладати свої рішення в пояснювальній записці.

Під час захисту дипломної роботи студент має стисло передати її основний зміст, акцентуючи увагу на актуальності та новизні роботи, можливості її практичного застосування, аргументовано подати прийняті в ній технічні рішення та обґрунтувати отримані результати.

Дипломний проект є самостійною роботою студента. За всі розроблені в ньому проектні рішення, а також правильність і обґрунтованість розрахунків і належне оформлення його матеріалів несе відповідальність автор.

До дипломного проектування допускається студент, який пройшов повний курс навчання та склав усі передбачені навчальним планом заліки та екзамени, тобто виконав усі вимоги навчального плану з напряму підготовки.

2. Організація виконання дипломного проекту

Студенту може бути призначена тема дипломного проекту з переліку рекомендованих тем. Також йому надається право самостійного вибору теми з урахуванням його схильностей і можливостей найбільш повно застосувати отримані знання. Якщо тема пропонується студентом, то вона повинна бути обговорена й погоджена з керівником дипломного проекту.

Для затвердження обраної теми студент подає заяву на ім’я завідувача кафедри інформаційних систем. Зразок заяви наведений у додатку А.

Після затвердження обраної теми студентові видається завдання на дипломний проект (додаток Б або В).

Безпосереднє керівництво дипломним проектом покладається на викладачів кафедри, які призначаються завідувачем кафедри інформаційних систем.

Керівник дипломного проекту видає студенту завдання на проект; допомагає студенту в складанні календарного плану; проводить консультації; контролює процес виконання проекту відповідно до календарного плану; рекомендує студенту науково-технічну літературу і нормативно-довідкові джерела з теми проекту; перевіряє матеріали роботи; здійснює попереднє заслуховування результатів виконання дипломного проекту.

Дипломний проект студент виконує самостійно. Це вимагає чіткої організації його роботи з моменту вибору теми роботи й до її захисту.

На початковому етапі студент повинен попередньо ознайомитися з основними публікаціями за темою дипломного проекту та скласти їх список.

На основі вивчення літературних джерел, які мають охоплювати як монографії, підручники та навчальні посібники, статті у періодичних виданнях, так і патентні матеріали, науково-технічні звіти, реферативні видання студент повинен чітко уявити собі, що зроблено в теоретичному та прикладному аспектах теми дипломного проекту, а також докладно ознайомитися з аналогічними рішеннями у відповідній галузі. За резуль-

татами цієї роботи оформляється аналітичний огляд (порівняльний аналіз), із якого мають логічно випливати вибрані методики досліджень.

Після вивчення літературних джерел студент складає попередній план виконання дипломного проекту, обговорює його з керівником. У процесі обговорення уточнюються вихідні дані для проектування й строки, що регламентують роботу студента. Після цього студент складає уточнений план роботи над проектом, узгоджує його з керівником та приступає до проектування. У процесі виконання дипломного проекту студент повинен регулярно відвідувати консультації керівника, подавати йому на перевірку робочі матеріали відповідно до плану-графіка виконання етапів проекту.

Контроль керівника дипломного проекту не звільняє студента від повної відповідальності за обґрунтованість прийнятих рішень, дотримання стандартів, термінів виконання календарного плану.

На засіданнях кафедри інформаційних систем регулярно заслуховуються повідомлення керівників дипломних проектів про хід виконання календарних планів. Студенти, що не дотримуються графіка виконання проекту або значно відстають у його виконанні, запрошуються для звіту на засідання кафедри.

3. Структура, зміст та обсяг дипломного проекту

За своїм характером дипломні проекти підрозділяються на такі типи:
проекти дослідницького характеру;
проекти практичного характеру.

Дипломні проекти дослідницького характеру

Проекти дослідницького характеру спрямовані на розроблення нових або застосування існуючих математичних моделей, методів чи алгоритмів для дослідження інформаційних процесів в галузі інформаційних технологій.

Вони мають містити елементи теоретичних або експериментальних досліджень.

У змістовній частині пояснювальної записки проекту дослідницького характеру, як правило, мають знайти відображення такі питання:

1. Огляд за літературними джерелами стану досліджень з тематики роботи, виявлення теоретичних передумов та можливих напрямків вирішення задач дослідження.

2. Обґрунтування використання основних теоретичних закономірностей та співвідношень.

3. Опис методики досліджень.

4. Вирішення задач дослідження на ПК (складання та налагодження програм для рішення, отримання результатів та їх аналіз).

5. Узагальнення результатів досліджень.

6. Загальні висновки з роботи з оцінкою застосування результатів досліджень.

Структура змістовної частини дипломних проектів дослідницького характеру повинна бути узгоджена з керівником відповідного проекту. Приклад змістовної частини дипломного проекту дослідницького характеру наведений у додатку Д.

Дипломні проекти практичного характеру

Проекти практичного характеру спрямовані на аналіз, моделювання, прогнозування інформаційних процесів, або керування такими процесами в економіці (як приклад, можна розглядати окреме підприємство, організацію, галузь чи географічний регіон). Вони передбачають вибір конкретної предметної області для аналізу та дослідження. Виконання таких проектів передбачає аналіз предметної області на основі вивчення спеціальної літератури та ознайомлення з інформаційними процесами безпосередньо на підприємствах та в організаціях.

Змістовна частина пояснювальної записки такого проекту має містити в собі:

1. Опис напрямів діяльності підприємства, організаційної структури підприємства, структурного підрозділу.

2. Аналіз предметної області, існуючих бізнес-проблем об’єкта проектування, опис бізнес-процесів.

3. Аналіз існуючих програмних продуктів, що реалізують функції предметної області.

4. Специфікацію вимог до програмного забезпечення.

5. Опис проектних та технічних рішень стосовно проектування бази даних, архітектури додатку, його тестування та розгортання.

Структура змістовної частини дипломних проектів практичного характеру, якої необхідно дотримуватися, наведена у додатку Е.

Дипломний проект складається з пояснювальної записки та інших обов’язкових матеріалів (схеми, діаграми, графіки залежностей, таблиці, рисунки, лістинги програм тощо), що розробляються відповідно до завдання.

Обсяг пояснювальної записки – 50 – 70 друкованих сторінок формату А4 (без додатків).

У табл. 1 наведено структура пояснювальної записки дипломного проекту та рекомендована кількість сторінок.

Таблиця 1

Структура пояснювальної записки дипломного проекту

Структурні елементи пояснювальної записки	Кількість сторінок
Титульний аркуш	1
Завдання на дипломний проект	2
Реферат	1 – 2
Зміст	1 – 2
Перелік умовних скорочень	1
Вступ	1 – 2
Розділ 1	9 – 12
Розділ 2	14 – 19
Розділ 3	19 – 25
Висновки	1
Список використаних джерел	1 – 2
Додатки	

Пояснювальна записка виконується у друкований спосіб на аркушах формату А4. Її текст повинен бути виконаний з використанням шрифту Times New Roman (розмір 14), з міжрядковим інтервалом 1,2 (37 рядків на сторінці). Найменшим розміром шрифту може буту розмір 10 (його можна використовувати при поданні вихідного тексту програм).

Поля сторінок пояснювальної записки – 30 мм від лівого краю аркушу, 10 мм – від правого краю аркушу, по 20 мм від верхнього та нижнього країв аркушу.

Весь текст пояснювальної записки, включаючи назви її структурних елементів, виконується шрифтом однакової жирності. Не дозволяється використання курсиву та підкреслення.

Абзацний відступ повинен бути однаковим впродовж усього тексту та дорівнювати 1,25 см.

Вирівнювання основного тексту проводиться "за шириною" сторінки.

Докладні вимоги до оформлення пояснювальної записки наведені у відповідних методичних рекомендаціях [64].

4. Методичні рекомендації до розроблення структурних елементів пояснювальної записки

Загальними вимогами до тексту пояснювальної записки є логічна послідовність викладення матеріалу, чіткість і конкретність викладення теоретичних і практичних результатів роботи, суті постановки завдання та мети роботи, методів дослідження, прийнятих рішень, доведеність висновків і обґрунтованість рекомендацій. У тексті пояснювальної записки необхідно дотримуватись єдиної термінології. Вона не має бути перевантажена малоінформативним матеріалом, описом загальновідомих даних, виведенням формул тощо. Необхідно посилатися на джерела інформації. У тексті пояснювальної записки має бути наведений використаний математичний апарат та результати виконаних розрахунків за допомогою ПК.

Текст пояснювальної записки не слід викладати від першої особи, переважніше безособова форма (наприклад, "обчислюється", "знаходимо") за всім текстом у визначеному відмінку й часі.

При викладенні матеріалу не слід використовувати:

розмовні звороти;

жаргонні слова та звороти;

різні терміни для позначення одного поняття;

іншомовні слова та терміни за наявності в українській мові рівнозначних слів і термінів;

скорочення слів і словосполучень, крім встановлених правилами орфографії та нормативними документами.

Першою сторінкою пояснювальної записки є **титульний аркуш**. Він містить дані, які подають у такій послідовності:

відомості про виконавця роботи – юридичну особу (організацію) або фізичну особу;

повна назва документа;

підписи відповідальних осіб, включаючи керівника роботи;

рік складення пояснювальної записки;

Зразок титульного аркушу дипломного проекту наведений у додатку Ж.

Реферат – це короткий виклад змісту пояснювальної записки, що містіть основні фактичні відомості і висновки, необхідні для початкового ознайомлення з нею. Реферат виконується українською та англійською мовами.

Реферат має бути стислим, інформативним і містити відомості, які дозволяють прийняти рішення про доцільність читання пояснювальної записки.

Реферат повинен містити:

відомості про обсяг звіту, кількість частин звіту, кількість ілюстрацій, таблиць, додатків, кількість джерел згідно з переліком посилань (усі відомості наводять, включаючи дані додатків);

текст реферату;

перелік ключових слів.

Текст реферату повинен відбивати подану в пояснювальній записці інформацію у такій послідовності:

об’єкт дослідження або розроблення;

мета роботи;

методи дослідження та апаратура;

результати та їх новизна;

основні технологічні й техніко-експлуатаційні характеристики та показники;

взаємозв’язок з іншими роботами;

рекомендації щодо використання результатів курсового проекту;

галузь застосування;

значущість роботи та висновки;

прогнозні припущення про розвиток об’єкту дослідження або розроблення.

Реферат належить виконувати обсягом не більш, як 500 слів, і, бажано, щоб він уміщувався на одній сторінці формату А4.

Ключові слова призначені для розкриття суті проекту та для розповсюдження інформації про розробку. Їх розміщують після тексту реферату. Перелік ключових слів містить від 5 до 15 слів (словосполучень), надрукованих великими літерами в називному відмінку в рядок через коми.

Приклади рефератів наведені у додатку З.

До **змісту** включають: вступ; послідовно перелічені назви всіх розділів, підрозділів та пунктів основної частини пояснювальної записки; висновки; перелік посилань; назви додатків і номери сторінок, які містять початок матеріалу.

Приклад змісту пояснювальної записки наведений у додатку И.

Якщо у пояснювальній записці використовуються маловідомі скорочення, нові символи, позначення і таке інше, то в ній має бути **перелік умовних скорочень**, який подається у вигляді окремого списку, що розміщують перед вступом. Незважаючи на це, за першої появи цих елементів у тексті документу надають їх розшифровку.

Якщо в роботі спеціальні терміни, скорочення, символи, позначення повторюються менше трьох разів, перелік не складають, а їх розшифровку наводять у тексті при першому згадуванні.

Приклад переліку умовних скорочень наведений у додатку К.

Вступ – це віддзеркалення роботи, тому слід ретельно його опрацювати. Краще вступ формувати після виконання основного тексту пояснювальної записки роботи.

У вступі до дипломного проекту практичного характеру необхідно ідентифікувати та сформулювати проблему бізнесу, яка виникла на підприємстві, обґрунтувати актуальність теми проекту для вирішення цієї проблеми на основі розроблюваного модуля або системи. Коротко охарактеризувати функціональність модуля або системи. Необхідно охарактеризувати технічну та програмну платформу розробки автоматизованого модуля. Потрібно сформулювати мету та задачі проекту, визначити об’єкт і предметну область проектування. Також необхідно навести інформацію щодо засобів проектування, які використовувались у дипломному проекті, та можливих галузей застосування результатів, отриманих у дипломному проекті.

У вступі до дипломного проекту дослідницького характеру викладається:

оцінка сучасного стану вирішення проблеми;

актуальність роботи;

мета та завдання роботи;

об’єкт та предмет дослідження;

методи дослідження;

наукове та практичне значення роботи;

відомості щодо публікацій за темою роботи.

Висновки до роботи – це резюме за результатами всієї роботи. Ця частина має особливу важливість, оскільки тут повинні бути наведені підсумкові результати роботи.

У пояснювальній записці має міститися висновки з кожного етапу виконаного проекту та по роботі в цілому, які необхідно співвіднести з метою і завданням на дипломне проектування.

Необхідно зазначити практичну цінність результатів роботи, дати рекомендації для подальшого вдосконалення об’єкту проектування. Зазначаючи практичну цінність одержаних результатів, важливо окреслити ступінь їх готовності до використання, масштабів використання, а також надати стислі відомості щодо впровадження результатів досліджень із зазначенням назв організацій, у яких здійснена реалізація, форм реалізації, реквізитів документів тощо. Якщо дипломний проект впроваджений на підприємстві, то до дипломної роботи додається довідка або акт про впровадження.

Наводять відомості де й коли було апробовано отримані результати (на яких конференціях, семінарах вони доповідались), перераховуються публікації студента за матеріалами дипломного проекту.

Список використаних джерел повинен містити відомості про літературні джерела, використані при розробленні проекту. Список використаної літератури рекомендується складати в такому порядку:

1. Основні нормативні документи і матеріали (державні і урядові).
2. Друковані джерела суспільно-політичного, соціального, економічного, природничо-наукового, соціально-культурного характеру.
3. Книги.
4. Статті.
5. Дисертації.
6. Автореферати.
7. Патентні документи.
8. Нормативно-технічні документи.
9. Каталоги промислового устаткування виробів.
10. Депоновані рукописи.

Вимоги до оформлення списку використаних джерел містяться в додатку Л.

У **додатках** вміщують матеріал, який є необхідним для повноти звіту, але не може бути послідовно розміщений в основній частині звіту через великий обсяг або способи відтворення та з інших причин.

Ілюстрації (діаграми бізнес-процесів, сценарії діалогів та ін.), таблиці, проміжні математичні докази, формули та розрахунки, текст допоміжного характеру та ін. можуть бути оформлені у вигляді додатків.

4.1. Рекомендації щодо розроблення розділів пояснювальної записки дипломного проекту дослідницького характеру

У основній частині роботи розглядається методика й техніка дослідження, узагальнюються його результати. Зміст розділів основної частини повинен відповідати темі роботи та повністю її розкривати. Ця частина роботи повинна показати вміння студента стиснуто, логічно й аргументувано викладати матеріал.

Основна частина роботи повинна містити інформацію, що відображає ціль, завдання, сутність, методику й основні результати виконаних досліджень.

Розділ 1. Формулювання наукової задачі, аналіз стану її вирішення за матеріалами вітчизняних та закордонних джерел.

Розділ 2. Аналіз методів, моделей, методик, існуючого методологічного та методичного забезпечення щодо завдання, яке вирішується. Обґрунтування вибору методу(-ів), методики (-ик), показників ефективності та інструментів дослідження.

Розділ 3. Виконання практичних досліджень, аналіз отриманих результатів з графічним та табличним відображенням одержаних переваг (поліпшень) основних характеристик (параметрів). Результати використання розробленого програмного продукту або існуючих інструментальних засобів для отримання та аналізу результатів за обраною моделлю. Аналіз практичної цінності отриманих результатів з вказівкою відповідної галузі щодо їх впровадження та напрямів (перспектив) подальшого розвитку.

4.2. Рекомендації щодо розроблення розділів пояснювальної записки дипломного проекту практичного характеру

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ <НАЗВА ПРЕДМЕТНОЇ ОБЛАСТІ>

Метою розділу 1 є проведення детального аналізу проблеми, яка виникла на об’єкті управління (підприємстві) при веденні бізнесу, вибір шляхів її вирішення.

У **підрозділі 1.1. Коротка характеристика об’єкту управління <назва об’єкту управління>** треба:

коротко описати напрями діяльності об’єкту управління (підприємства, організації);

розробити схему організаційної структури управління підприємством (додаток М, рис. М.1);

визначити проблему бізнесу, яку слід вирішити, та бізнес-процеси, які пов’язані з вирішенням даної проблеми, вказати, які підрозділи організаційної структури їх виконують;

розробити схему організаційної структури підрозділу (підрозділів), пов’язаних з визначеними бізнес-процесами (додаток М, рис. М.2).

У **підрозділі 1.2. Опис предметної області <назва предметної області>** необхідно:

визначити склад функцій, що входять до бізнес-процесу, розробити діаграму дерево функцій (додаток М, рис. М.3);

розробити схему управління бізнес-процесом (додаток М, рис. М.4) та описати його (табл. 2).

Таблиця 2

Характеристика бізнес-процесу <Назва>

Назва характеристики	Значення характеристики
Ім’я бізнес-процесу	
Основні учасники*	
Вхідна подія	
Вхідні документи**	
Вихідна подія	
Вихідні документи**	
Клієнт бізнес-процесу***	

*для кожного учасника указати структурний підрозділ, посаду, його роль у бізнес-процесі,

** навести перелік документів,

*** процес, що використовує інформацію процесу

Для моделювання предметної області використовують CASE-інструменти.

У процесі моделювання необхідно виділити транзакційну складову бізнес-процесу, яка забезпечує збір, накопичення та обробку кількісних даних про поточний стан об’єкта управління, а також аналітичну складову, яка забезпечує аналіз кількісних показників, сформованих у транзакційні складові.

Аналітична складова бізнес-процесу повинна забезпечити дослідження кількісних показників у різних розрізах та вимірах: за періодами часу, за товарами (продукцією), за клієнтами, підрозділами.

Проведення такого багатоаспектного аналізу забезпечить інформаційну підтримку прийняття рішень, спрямованих на вирішення виявленої проблеми.

У **підрозділі 1.3. Огляд і аналіз існуючих аналогів, що реалізують функції предметної області** треба:

виконати аналіз функціональності й інтерфейсу двох або більше програмних продуктів, призначених для автоматизації бізнес-процесів розроблюваного модуля (системи):

навести в таблиці порівняльну характеристику програмних продуктів за такими характеристиками (табл. 3):

фірма-розробник;

назва програмного продукту;

версії продукту;

функціональність;

інтерфейс користувача;

допомога користувачу;

тощо.

Таблиця 3

Порівняльна характеристика програмних продуктів

Фірма-розробник	<Назва фірми-розробника>	<Назва фірми-розробника >	. . .
Назва програмного продукту			
Версії продукту			
. . .			

Для кожного з програмних продуктів навести та коротко описати екранні форми, що характеризують основні варіанти використання продукту.

Зробити висновок щодо можливості використання досвіду ведучих фірм-розробників програмних продуктів, використання їх рішень при розробленні проектних рішень проекту.

РОЗДІЛ 2. СПЕЦИФІКАЦІЯ ВИМОГ ДО МОДУЛЯ (СИСТЕМИ)

Метою розділу 2 є розроблення і детальна специфікація вимог до модуля (системи), що розробляється.

Підрозділ 2.1. Глосарій

Глосарій – це словник основних використовуваних термінів. Цей документ є найпершим результатом концептуального аналізу предметної області. Глосарій можна розглядати як документ, що засвідчує спільне розуміння основної термінології Замовником і Розробником.

Крім того, глосарій є відправною точкою для побудови більш розгорнених моделей предметної області, які на стадії реалізації інформаційної системи лягають в основу об’єктної моделі (для об’єктно-орієнтованих застосувань) і моделі даних (для генерації схеми бази даних).

Глосарій необхідно подати у вигляді табл. 4.

Таблиця 4

Глосарій

Термін	Опис терміну
1. Основні поняття та категорії предметної області та проекту	
2. Користувачі системи	
3. Вхідні та вихідні документи	

Підрозділ 2.2. Розроблення варіантів використання містить:

діаграму варіантів використання;

специфікацію варіантів використання;

розкадровку варіантів використання;

специфікацію функціональних та нефункціональних вимог.

Пункт 2.2.1. Діаграма варіантів використання містіть діаграму варіантів використання.

Діаграма варіантів використання відображає функціональність, яка буде реалізована в програмному продукті. Необхідно навести опис потоків подій і діаграми послідовності.

Варіант використання можна розглядати як функцію, що реалізовується системою. Однак будь-яка функція повинна мати цінність і давати можливість отримати кінцевий результат для кінцевого користувача продукту або послуги. Тому при специфікації варіанта використання виділяють серед усієї функціоналності системи лише ту функціональність, яка:

- **корисна** конкретному кінцевому користувачеві**;**
- дозволяє отримувати користувачеві **конкретні закінчені результати.**

Перш ніж приступити власне до специфікації вимог у формі варіанта використання, в RUP звичайно складають реєстр (список) акторів (actors) і варіантів використання.

Актор – це хтось або щось, що володіє активністю відносно програмної системи. Актором звичайно буде користувач системи. Окрім користувача, як актор може розглядатися інша програмна система, апаратний пристрій, у ряді випадків – активна компонента самої системи. Пошук акторів корпоративної інформаційної системи зазвичай зводиться до аналізу ролей різних користувачів. Вибір акторів залежить від їх функціональних обов'язків, розмежування доступу, способів використання інформаційної системи.

Пункт 2.2.2. Специфікація варіантів використання

Кожен варіант використання повинен мати опис. У дипломному проекті навести опис варіантів використання, що реалізують основну функціональність (зазвичай крім ведення довідників) у вигляді таблиці (табл. 5).

Таблиця 5

Варіант використання <Назва>

Контекст використання	
Дійові особи	
Предумова	
Триггер	
Сценарій	
Постумова	

Назва – коротка фраза в вигляді дієслова в неозначеній формі завершеного виду, яка відображає мету.

Контекст використання містить уточнення мети, а при необхідності – умови її нормального завершення.

Дійові особи:

основна дійова особа – це ім'я ролі основного актора або його опис;

учасники і інтереси – перелік інших акторів-учасників варіанта з вказівкою їх інтересів.

Передумова – варіант використання, який має бути обов'язково виконаний, щоб можна було виконати даний варіант. Передумову описує стан, у якому система повинна перебувати до початку виконання варіанта.

Тригер – подія предметної області, що викликає використання прецеденту (варіанта використання).

Сценарій – перелік кроків сценарію: номер кроку – опис дії

Постумова – варіант використання, який обов'язково має бути виконаний після виконання даного варіанту; це стан, у якому система повинна перебувати після закінчення виконання варіанта; це те, що гарантується акторам-учасникам, не залежно від успіху виконання даного варіанта. Наприклад, у разі невдалої транзакції всі дані, що були в системі до її початку, зберігаються незмінними.

Події, що описуються передумовами або постумовами, мають бути станами, які користувач може спостерігати.

Пункт 2.2.3. Розкадровка варіантів використання

Розкадровка (storyboard) – це логічний і концептуальний опис функціональних можливостей системи для певного сценарію, включаючи необхідну взаємодію між системою та її користувачами. Метою розкадрування є отримання ранньої реакції користувачів на пропоновану концепцію системи за допомогою нецінних засобів. У якості інструментальних засобів розкадровки вимог використовуються Microsoft Word, Microsoft Visio, Microsoft PowerPoint, IBM Rational Requirements Composer, Expression Blend Sketch Flow.

Розкадровка – це представлення варіантів використання по кадрам (ескізам екранних форм).

Розкадровка дозволяє зацікавленим особам описати свої потреби аналітикам, які на основі цих потреб визначають вимоги до системи, здійснюють перевірку відповідності вимог поставленим бізнес-завданням і здійснюють з ними зворотний зв'язок.

За результатами створення розкадрування повинні бути сформульовані вимоги (з можливістю відстежити процес "в зворотному напрямі" аж до конкретного елемента розкадрувань), які підлягають узгодженню з зацікавленими особами. Потім ці вимоги повинні бути включені у відповідний набір вимог. Результати розкадрувань використовуються для отримання схвалення зацікавлених осіб, виявлення джерела вимог в рамках розкадрувань.

Підрозділ 2.3. Специфікація функціональних та не функціональних вимог

Специфікацію функціональних вимог навести в таблиці (табл. 6)

Таблиця 6

Специфікація функціональних вимог

Ідентифікатор вимоги	Назва вимоги (варіанта використання)	Атрибути вимог		
		Пріоритет	Трудність	Контакт

Пріоритет – заповнюється аналітиком, показує пріоритет реалізації вимоги для клієнта. Використовується при управлінні проектом і визначає пріоритет розробки. Можливі значення: обов’язкове, рекомендоване, опційне (за вибором).

Трудність – заповнюється менеджером проекту, показує рівень трудовитрат, пов’язаних з реалізацією вимоги. Використовується при управлінні проектом і визначає пріоритет розробки. Трудність виконання вимоги може виражатися у вигляді трудомісткості і вказувати кількість людино-днів, потрібних для його реалізації або у вигляді значень шкали: висока, середня, низька.

Контакт – заповнюється аналітиком, ідентифікує людину, яка може надати необхідну інформацію про вимогу. Використовується для гарантії, що розробники можуть отримати інформацію, необхідну їм для реалізації вимоги. Якщо в проекті в якості контакту виступає одна людина – цю колонку можна опустити, але треба ідентифікувати контакт один раз у тексті цього пункту.

Специфікацію нефункціональних вимог навести в таблиці (табл. 7)

Таблиця 7

Специфікація нефункціональних вимог

Ідентифікатор вимоги	Назва вимоги	Атрибути вимог		
		Пріоритет	Важкість	Контакт
1. Застосовуваність				
2. Надійність				
. . .				

Нефункціональні вимоги можна поділити на такі групи:

1. Застосовуваність.

Час, необхідний для навчання звичайних і досвідчених користувачів.

Вимірний час відгуку для типових завдань.

Основні вимоги застосовуваності нової системи відносно інших систем, які знають користувачі.

Вимоги відповідності загальним стандартам застосовуваності, наприклад, стандартам інтерфейсу користувача IBM або стандартам графічного інтерфейсу користувача Microsoft для Windows.

2. Надійність.

Доступність – визначає % доступного часу (xx.xx %), час використання, час, що витрачається на обслуговування, порушення режиму роботи і т. д.

Середній час безвідмовної роботи – зазвичай визначається в годинах, але може також визначатися в днях, місяцях або роках.

Середнє напрацювання до ремонту – як довго системі дозволяють працювати до того, коли повинно бути проведене її обслуговування.

Точність – визначає розрядність (роздільну здатність) і точність (за деяким відомим стандартом), які потрібні у вихідних даних системи.

Максимальна норма помилок або дефектів – зазвичай виражається в термінах кількості помилок на тисячу рядків коду або кількості помилок у функціональній одиниці.

3. Робочі характеристики.

Повинні бути виділені характеристики продуктивності системи. Включіть конкретні характеристики швидкодії. Де можливо, потрібно зробити посилання на пов’язані прецеденти по імені.

Швидкодія для транзакції (середнє значення, максимальне).

Продуктивність (наприклад, число транзакцій за секунду).

Місткість (наприклад, число замовників або транзакцій, яке може розміщувати система).

Режими зниженої продуктивності (що є прийнятним режимом роботи, коли система стала деякою мірою гіршою).

Використання ресурсів: пам'яті, дискового простору, комунікацій і т. д.

4. Експлуатаційна придатність.

Указують всі вимоги, які розширюють експлуатаційну придатність або надійність формованої системи, включаючи стандарти кодування, угоди про імена, бібліотеки класів і утиліти підтримки.

5. Проектні обмеження.

Повинні містити всі проектні обмеження до формованої системи. Проектні обмеження становлять рішення, які були сформульовані як обов'язкові і повинні твердо витримуватися. Прикладами можуть бути мови програмування, вимоги до технології програмування, обов'язкове використання інструментальних засобів розробки, архітектурні і конструктивні обмеження купованих компонентів, бібліотек класів і т. д.

6. Вимоги до документації, призначеної для користувача, і до системи допомоги.

Описують вимоги, якщо вони є, до інтерактивної документації користувача, до системи довідки, до попереджувальних повідомлень і т. д.

7. Куповані компоненти.

Описують всі куповані компоненти, які має використовувати система, всі вживані ліцензії або обмеження щодо використання і всі відомості про сумісність і / або здібність до взаємодії або про стандарти інтерфейсу.

8. Інтерфейси.

Визначають інтерфейси, які повинні бути підтримані застосуванням. Він має містити адекватну специфіку, протоколи, порти і логічні адреси та ін. так, щоб програмне забезпечення могло бути розроблене і перевірене на відповідність вимогам інтерфейсів.

8.1. Інтерфейси користувача.

Описуються інтерфейси користувача, які повинні бути реалізовані програмним забезпеченням.

8.2. Апаратні інтерфейси.

Визначаються всі апаратні інтерфейси, які повинні бути підтримані програмним забезпеченням, включаючи логічну структуру, фізичні адреси, очікувану поведінку і т. д.

8.3. Програмні інтерфейси.

Описуються програмні інтерфейси з іншими компонентами програмної системи. Це можуть бути куповані компоненти, багато разів використовувані компоненти з іншої прикладної програми або компоненти, що розробляються для підсистем поза контекстом цих специфікацій вимог до програмного забезпечення (SRS), але з яким ця прикладна програма повинна взаємодіяти.

8.4. Комунікаційні інтерфейси.

Описуються всі комунікаційні інтерфейси до інших систем або пристроїв типу локальних мереж, видалених послідовних пристроїв і т. д.

9. Вимоги до ліцензування.

Визначаються всі вимоги обов'язкового ліцензування або інші вимоги обмеження використання, які повинні виконуватися програмним забезпеченням.

10. Застереження щодо питань, пов'язаних з авторськими правами.

Описуються всі необхідні юридичні застереження, гарантії, оголошення про авторське право, право спадкоємства, торгівельні марки або емблеми для програмного забезпечення.

11. Вживані стандарти.

Вказуються посилання на всі вживані стандарти і на конкретні розділи таких стандартів, які відносяться до описуваної системи. Наприклад, це можуть бути правові і регулюючі стандарти, стандарти якості, промислові стандарти щодо застосовуваності, здібності до взаємодії, інтернаціоналізації, відповідності операційній системі і т. д.

У специфікації вказують лише ті нефункціональні вимоги, які є суттєвими для проекту.

РОЗДІЛ 3. ПРОЕКТНІ ТА ТЕХНІЧНІ РІШЕННЯ

Розділ 3 призначений для проектування та розроблення бізнес-додатка, що призначений для автоматизації бізнес-задач з використанням сучасних інформаційних технологій.

Підрозділ 3.1. Математична постановка задачі

Потрібно обґрунтувати вибір економіко-математичної моделі вирішення задачі, якщо задача є оптимізаційною або задачею прогнозування. Економіко-математична модель включає цільову функцію для обраного критерію ефективності та систему обмежень. Необхідно описати послідовність розрахунку результатних показників відповідно до логіки реалізації економіко-математичної моделі.

Якщо задача вирішується на основі алгоритму прямого лічення, необхідно навести розрахункові формули у вигляді математичної залежності вихідних показників, що розраховуються, від вхідних.

Якщо задача не має математичного формулювання її вирішення, необхідно навести опис логіки послідовних дій у вигляді виконуваних функцій обробки інформації з задачі (добору, порівняння).

Математична і логічна моделі вирішення задачі повинні бути описані з достатнім ступенем деталізації, щоб за ними можна було написати програму розрахунку показників вихідних повідомлень.

Підрозділ 3.2. Проектування структури бази даних

Рекомендується така структура даного підрозділу.

3.2.1. Опис вхідної та вихідної інформації, яка обробляється в рамках функцій предметної області, що автоматизується

3.2.2. Концептуальне інфологічне проектування

Має містити опис локальної та глобальної інфологічних моделей даних.

3.2.3. Проектування глобальної даталогічної моделі даних

Має містити інформацію стосовно проектування глобальної даталогічної моделі даних у вигляді ERD (в нотації IDEF1X), або у вигляді діаграми класів, яка може бути розроблена за допомогою пакетів Rational Rose, ARIS та ін., а також – стосовно проектування глобальної даталогічної моделі даних у вигляді реляційної моделі.

3.2.4. Проектування фізичної моделі даних

3.2.5. Програмна реалізація бази даних

Підрозділ 3.3. Розроблення архітектури програмної системи

У цьому пункті пояснювальної записки має знаходитися:

1. Структурна схема програмної системи та її опис. Опис структурної схеми має містити відомості про призначення елементів програмної системи, їх інформаційні зв’язки та взаємодію.

2. UML-діаграма класів, що реалізують основну бізнес-логіку програмної системи, та її короткий опис, у якому необхідно навести призначення кожного класу.

3. UML-діаграма станів, у яких можуть знаходитися елементи графічного інтерфейсу користувача, та її опис.

4. Лістинг програми, де міститься вихідний код, що відповідає класам, які реалізують основну бізнес-логіку програмної системи. Лістинг програми повинен супроводжуватися коментарями (для класів – призна-

чення класу; для методів – призначення методу, опис параметрів та значення, що повертається). Якщо програма реалізована з використанням програмної платформи .NET або Java, обов'язковим є використання документаційних XML- або JavaDoc-тегів відповідно.

Далі знаходяться методичні рекомендації щодо розроблення архітектури програмної системи.

Рекомендації щодо розроблення структурної схеми програмної системи

Програмна система означає програмне забезпечення, що розроблюється. Це може бути велика колекція з безлічі компонентів програмного забезпечення, один додаток або частина додатку.

Найважливішими характеристиками архітектури будь-якої програмної системи є її структура й процес функціонування.

Під структурою програмної системи розуміють стійку в часі сукупність взаємозв'язків між її елементами. Залежно від рівня деталізації структурні елементи можуть бути підсистемами, модулями, процесами, бібліотеками і т. д. Саме структура зв'язує воєдино всі елементи програмної системи й забезпечує її існування як єдиного цілого.

Структурна схема програмної системи – це її графічне зображення у вигляді сукупності складових елементів та інформаційних зв'язків між ними з зазначенням їх напряму. Приклад структурної схеми програмної системи наведений у додатку Н.

Рекомендації щодо розроблення UML-діаграма класів, що реалізують основну бізнес-логіку програмної системи

Процес функціонування програмної системи тісно пов'язаний зі зміною її властивостей або поводження в часі. Тому поряд з визначенням структурних елементів будь-яка архітектура визначає взаємодію між ними, що забезпечує бажане поводження системи.

Одним із принципів побудови моделей складних систем є принцип багатомодельності. Відповідно до цього принципу ніяка єдина модель не може з достатнім ступенем адекватності описувати різні аспекти складної системи.

Стосовно об'єктно-орієнтованого аналізу й проектування програмних систем це означає, що досить повна модель такої системи повинна містити деяке число взаємозалежних уявлень, кожне з яких адекватно відбиває деякий аспект її поводження або структури.

Найбільш загальними уявленнями складної системи прийнято вважати статичне й динамічне уявлення.

Для уявлення статичної структури моделі системи в термінології класів об’єктно-орієнтованого програмування призначена UML-діаграма класів. Вона може відображати, зокрема, різні взаємозв’язки між окремими сутностями предметної області, такими, як об’єкти й підсистеми, а також описує їхню внутрішню структуру й типи відносин.

Клас у мові UML служить для позначення безлічі об’єктів, які мають однакову структуру, поводження і відносини з об’єктами інших класів. Клас може не мати екземплярів або об’єктів. У цьому випадку він називається абстрактним класом. Графічно клас зображується у вигляді прямокутника, що додатково може бути розділений горизонтальними лініями на секції. У цих секціях можуть вказуватися ім’я класу, атрибути й операції.

Обов’язковим елементом позначення класу є його ім’я. Ім’я класу повинне бути унікальним у межах пакета, що описується деякою сукупністю діаграм класів. Воно вказується в першій верхній секції прямокутника. Рекомендується як імена класів використовувати іменники, записані без пробілів. Необхідно пам’ятати, що саме імена класів утворять словник предметної області. Прикладами імен класів можуть бути такі іменники, як "Співробітник", "Компанія", "Керівник", "Клієнт", "Продавець", "Менеджер", "Офіс" та, що мають безпосереднє відношення до предметної області й функціонального призначення проектованої системи.

У другій зверху секції прямокутника класу записуються його атрибути. Кожному атрибуту класу відповідає окремий рядок тексту, що складається із квантора видимості атрибута, імені атрибута, його кратності, типу значень атрибута й, можливо, його вихідного значення.

Квантор видимості може приймати одне із трьох можливих значень:

1. Загальнодоступний (public). Атрибут із цією областю видимості доступний з будь-якого іншого класу пакета, у якому визначена діаграма.

2. Захищений (protected). Атрибут із цією областю видимості недоступний для всіх класів, за винятком підкласів даного класу.

3. Закритий (private). Атрибут із цією областю видимості недоступний для всіх інших класів без винятку.

Ім’я атрибута становить собою рядок тексту, що використовується як ідентифікатор відповідного атрибута й тому повинен бути унікальним в межах даного класу. Ім’я атрибута є єдиним обов’язковим елементом синтаксичного позначення атрибута.

Кратність атрибута характеризується загальною кількістю конкретних атрибутів даного типу, що входять до складу окремого класу.

Тип атрибута становить вираз, семантика якого визначається мовою специфікації відповідної моделі. У нотації UML тип атрибута іноді визначається залежно від мови програмування, яку передбачається використовувати для реалізації даної моделі. У найпростішому випадку тип атрибута вказується рядком тексту, що має осмислене значення в межах пакета або моделі, до яких ставиться розглянутий клас.

У третій зверху секції прямокутника записуються операції класу. Операція становить деякий сервіс, що надає будь-який екземпляр класу на певну вимогу. Сукупність операцій характеризує функціональний аспект поводження класу.

Кожній операції класу відповідає окремий рядок, що складається із квантора видимості операції, імені операції, вираження типу, що повертається операцією.

Ім’я операції становить рядок тексту, що використовується як ідентифікатор відповідної операції й тому повинен бути унікальним у межах даного класу.

Крім внутрішнього устрою або структури класів на відповідній діаграмі вказуються різні відносини між класами. При цьому сукупність типів таких відносин фіксована в мові UML і визначена семантикою цих типів відносин. Базовими відносинами або зв’язками в мові UML є:

1. Відношення залежності.
2. Відношення асоціації.
3. Відношення узагальнення.
4. Відношення реалізації.

Відношення залежності в загальному випадку вказує деяке семантичне відношення між двома елементами моделі або двома множинами таких елементів, що не є відношенням асоціації, узагальнення або реалізації. Відношення залежності використовується в такій ситуації, коли деяка зміна одного елемента моделі може потребувати зміни іншого залежного від нього елемента моделі.

Відношення асоціації відповідає наявності деякого відношення між класами. Найбільш простий випадок даного відношення – бінарна асоціація. Вона зв’язує в точності два класи й, як виключення, може зв’язувати клас із самим собою.

Тернарна асоціація й асоціації більш високого порядку в загальному випадку називаються N-арною асоціацією. Така асоціація зв’язує деяким відношенням три і більше класів, при цьому один клас може брати участь в асоціації більш ніж один раз.

Окремим випадком відношення асоціації є відношення агрегації, яке, в свою чергу, теж має спеціальну форму – відношення композиції.

Відношення агрегації має місце між декількома класами в тому випадку, якщо один із класів становить деяку сутність, що включає в себе як складові частини інші сутності. Дане відношення має фундаментальне значення для опису структури складних систем, оскільки застосовується для уявлення системних взаємозв’язків типу "частина-ціле". Розкриваючи внутрішню структуру системи, відношення агрегації показує, з яких компонентів складається система і як вони зв’язані між собою. Це відношення по своїй суті описує декомпозицію складної системи на більш прості складові частини, які також можуть бути піддані декомпозиції, якщо в цьому виникне необхідність надалі.

Відношення композиції є окремим випадком відношення агрегації. Це відношення служить для виділення спеціальної форми відносини "частина-ціле", при якій складові частини в деякому змісті перебувають усередині цілого. Специфіка взаємозв’язку між ними полягає в тому, що частини не можуть виступати у відриві від цілого, тобто зі знищенням цілого знищуються й всі його складові частини.

Відношення узагальнення є відношенням між більш загальним елементом (батьком або предком) і більш приватним або спеціальним елементом (дочірнім або нащадком). Стосовно до діаграми класів дане відношення описує ієрархічну будову класів і спадкування їхніх властивостей і поводження. При цьому передбачається, що клас-нащадок має всі властивості й поводження класу-предка, а також має свої власні властивості й поводження, які відсутні у класа-предка.

Інтерфейс користувача складається з елементів, які можуть перебувати в певних станах, а також у процесі взаємодії з користувачем програми переходити зі стану в стан. Він є системою, що управляється подіями (СУП).

Поводження таких систем найкраще характеризується їхньою реакцією на зовнішні події. Як правило, СУП перебуває в стані очікування, поки не одержить повідомлення про подію. Після того як СУП відреагує на подію, вона знову переходить у стан очікування наступної

події. Для таких систем важливо визначити насамперед стійкі стани, події, що ініціюють переходи з одного стану в інший, і дії, що виконуються при зміні стану.

При формальному описі СУП доцільно використовувати UML-діаграми станів.

Діаграма станів зв'язує події й стани. При виникненні події наступний стан системи залежить як від її поточного стану, так і від події. Зміна стану називається переходом. Діаграма станів – це граф, вузли якого моделюють стани, а спрямовані дуги, позначені іменами відповідних подій, – переходи.

При розробленні інтерфейсу користувача необхідно керуватися методичними рекомендаціями, викладеними нижче.

Рекомендації щодо розроблення інтерфейсу користувача

Користувальницький інтерфейс є своєрідним комунікаційним каналом, по якому здійснюється взаємодія користувача й комп'ютера.

Щоб створити ефективний інтерфейс, що здійснював би роботу із програмою приємною, потрібно розуміти, які завдання будуть вирішувати користувачі за допомогою даної програми і які вимоги до інтерфейсу можуть виникнути в користувачів.

Загальні принципи проектування інтерфейсу користувача

1. Програма повинна допомагати виконувати завдання.

Це значить, що інтерфейс повинен бути легким для освоєння й не створювати перед користувачем перешкоду, яку він повинен буде подолати, щоб приступитися до роботи.

2. При роботі із програмою користувач не повинен відчувати дискомфорт.

Для реалізації цього принципу необхідно:

забезпечити перевірку результатів як можна більшого числа "некоректних" дій користувача, але не робити її повсюдно;

вказувати користувачеві, що саме йому робити, і виводити інформаційні повідомлення в ситуаціях, коли це дійсно необхідно;

надати досвідченим користувачам можливість відключення виведення інформаційних повідомлень;

добре продумувати зміст повідомлень, що виводяться користувачеві.

Широко відомі евристичні правила авторитетного американського фахівця в галузі проектування інтерфейсів Якоба Нільсена. Вони визначають мінімальні критерії, яким повинен відповідати інтерфейс будь-якої програми.

1. Наочність стану системи (правило зворотного зв'язку).

Система (у цьому випадку – комп'ютерна програма) повинна завжди інформувати користувача про стан своєї роботи за допомогою засобів зворотного зв'язку в прийнятний час.

При розгляді цього правила потрібно враховувати кілька аспектів:

користувач завжди повинен мати інформацію про поточний стан програми (наприклад, скільки часу пройшло від початку процесу копіювання файлів);

користувач обов'язково повинен бачити, до чого привело будь-яку його дію, наприклад, уведення даних, натискання кнопки;

вибір конкретного засобу зворотного зв'язку залежить від типу інформації, яку потрібно донести до користувача, а також типу дії, що викликає потребу в зворотному зв'язку.

Уважається, що якщо користувач виконав якусь дію й очікує результат його виконання, то цей результат (або повідомлення про помилку) повинен бути виведений в окремому діалоговому вікні. Якщо ж користувачеві необхідно надати поточну інформацію про процес, що не є прямим наслідком його дій, то можна обмежитися відображенням відповідного повідомлення в панелі стану.

Для організації зворотного зв'язку можуть бути використані й інші засоби. Найпопулярніші з них – звуки. Найбільш часто звукове оповіщення допомагає тоді, коли поява на екрані діалогових вікон є небажаною, а повідомлення в панелі стану можуть бути не помічені користувачем.

Важливо пам'ятати, що звукове оповіщення не повинне бути основним засобом організації зворотного зв'язку. Звук повинен лише доповнювати текстові повідомлення.

Проміжок часу, протягом якого користувач одержує інформацію про реакцію на його дію або про подію, повинен бути мінімальним. Це особливо важливо, тому що наявність або відсутність у користувача інформації про поточний стан системи визначає його подальші дії.

2. Відповідність між системою й реальним світом.

Система повинна взаємодіяти з користувачем на його мові. У цьому випадку мається на увазі використання понять, образів і цілих концепцій, які знайомі користувачеві з реальної предметної області. У жодному разі не можна використовувати спеціалізовані терміни, які придатні тільки для професійних довідників з програмування.

Найпоширеніший приклад реалізації цього принципу – побудова користувальницького інтерфейсу, що імітує об’єкти реального світу. Наприклад, більшість із програм, що реалізують функції годинників, калькуляторів, програвачів компакт-дисків, записних книжок виглядають майже точно так, як їхні матеріальні аналоги. Знаменитий "сміттєвий кошик" на "робочому столі" операційної системи Windows, у який можна "кинути" непотрібний файл або папку, – класичний приклад побудови інтерфейсу на основі об’єктів реального світу.

3. Управління користувачами та свобода їхніх дій.

Користувачі повинні мати можливість управління системою й зміни її поточного стану. Однак, вони часто дають різні команди помилково (наприклад, випадково натиснувши кнопку або вибравши не той пункт меню). Отже, у користувача повинен бути "аварійний вихід" із цієї ситуації, чітко позначений у програмі. Найчастіше такий "вихід" реалізується у вигляді кнопки "Відміна", розташованої в діалоговому вікні, що дозволяє припинити виконання поточної операції або закрити це діалогове вікно. Крім цього, традиційним і тому звичним для більшості користувачів засобом "аварійного виходу" є натискання на клавіатурі клавіші <Escape>.

Хорошим тоном вважається сполучення цих способів "аварійного виходу".

Ще один засіб виходу з помилкової ситуації – команди "Undo" ("Відмінити") і "Redo" ("Повторити"). Вони є настільки зручними й підтримуються такою великою кількістю програм, що користувачі підсвідомо очікують, що будь-яку їхню дію можливо відмінити, повернувшись до попереднього стану.

Все це зобов’язує розроблювача дружнього користувальницького інтерфейсу комп’ютерної програми підтримувати дані команди. Якщо ж за якимись причинами дію, на виконання якої дав команду користувач, не можна відмінити, то на екран повинно бути виведене відповідне попередження, а також прохання підтвердити виконання команди.

4. Несуперечність і стандарти.

Цей принцип означає використання тих самих засобів для вираження схожих образів і виконання дій, що мають однакову природу.

По-перше, це означає несуперечність при виборі засобів оповіщення про події та дії. Наприклад, інформація про поточний стан програми, звичайно виводиться в панелі стану, а повідомлення з результатами запитів користувача – в окремих діалогових вікнах.

Повідомлення про критичні помилки при цьому повинні сильно відрізнятися від звичайних інформаційних повідомлень: наприклад, вони можуть супроводжуватися різким звуком.

По-друге, це означає несуперечність при оформленні елементів користувальницького інтерфейсу. Наприклад, якщо він ґрунтується на класичному інтерфейсі Windows-додатків, що характеризується строгою колірною гамою, прямими лініями й кутами, то дуже дивним виглядало б рішення додати одному з вікон програми овальну форму й розфарбувати його яскравими кольорами.

По-третє, це означає несуперечність при виборі термінів. Користувачів не повинно спантеличувати те, що кілька різних понять, які використовуються в програмі, насправді означають те саме.

Головне – вирішити, що для позначення якоїсь конкретної дії або події буде застосовуватися один конкретний термін, що буде використовуватися певним способом (наприклад, слово "Інтернет" буде починатися з великої букви й не відмінюватися).

Принцип несуперечності – одне з найважливіших правил проектування користувальницьких інтерфейсів. Несуперечливий інтерфейс інтуїтивно зрозумілий і дуже легкий для освоєння, тому що при його вивченні користувач не зіштовхується з "сюрпризами", і навіть ті частини інтерфейсу, які він бачить уперше, здаються йому давно знайомими.

5. Запобігання помилок.

Стосовно теми проектування користувальницького інтерфейсу комп'ютерних програм, цей принцип означає: "Дизайн, що запобігає виникненню помилок, краще, ніж найліпше повідомлення про помилку".

6. Розуміння краще, ніж запам'ятовування.

При розробленні інтерфейсу потрібно робити всі об'єкти, функції, дії легкодоступними користувачеві. Користувач не повинен постійно намагатися утримати в пам'яті інформацію з однієї частини програми, щоб застосувати її в іншій. У будь-який момент часу користувачеві повинно бути зрозуміло, що йому зараз потрібно робити. У хорошому інтерфейсі інструкції з використання системи доступні для виклику в будь-який час, коли це потрібно. Це може бути реалізоване як у вигляді продуманої організації елементів інтерфейсу, так і у вигляді підказок користувачеві.

Це правило відбиває принцип "прозорого" інтерфейсу – інтерфейсу, що зрозумілий і не змушує користувача згадувати, яку кнопку потрібно натиснути або який пункт меню вибрати в даний момент.

7. Гнучкість і ефективність використання.

При проектуванні інтерфейсу користувача перед розроблювачем часто постає така проблема: потрібно, щоб інтерфейс був однаково зручний і для новачків, і для досвідчених користувачів.

Для вирішення цієї проблеми використовують простий прийом: функції, які прискорюють роботу, оформлюють так, щоб їх не бачив початківець, але щоб вони були доступні досвідченим користувачам. Найпростіший приклад – це "гарячі клавіші", за допомогою яких можна швидко викликати функції програми, що часто виконуються. Позначення "гарячих клавіш" пишуться поруч із відповідними пунктами меню, тому вони, з одного боку, не заважають новачкам, а, з іншого боку, доступні досвідченим користувачам.

Інший приклад реалізації універсального користувальницького інтерфейсу – можливість виконати складні функції програми як за допомогою "майстра", що, немов за руку, "проведе" починаючого користувача по всіх етапах процесу, так і вручну, за допомогою настроювання опцій у відповідному діалоговому вікні.

8. Естетичний і мінімалістський дизайн.

Це правило означає: "Нічого зайвого". Не потрібно захаращувати користувальницький інтерфейс програми елементами, які є недоречними й малокорисними. Справа в тому, що кожний елемент, будь то кнопка або текстовий підпис, обов’язково відволікає частину уваги користувача. Це може призвести до того, що видимість і, відповідно, легкість сприйняття користувачем дійсно потрібних і корисних частин інтерфейсу буде сильно зменшена за рахунок елементів, без яких цілком можна було б обійтися.

9. Розпізнавання й виправлення помилок.

"Допомагайте користувачеві розпізнавати й виправляти помилки" – говорить це правило.

Воно визначає проектування повідомлень про помилки. "Гарні" повідомлення про помилки – це повідомлення, які пояснюють, у чому полягає проблема й, найголовніше, як її виправити. Таким чином, "гарне" повідомлення про помилку повинне складатися із двох частин: опису помилки й опису вирішення проблеми.

Опис помилки повинен бути чітким, ясним і зрозумілим, давати користувачеві всю необхідну інформацію про причини й місце виникнення помилки. Найпростіше рішення – створити в довідковій системі

програми відповідний розділ, що роз’яснює зміст проблеми й причини її виникнення. У самому ж діалоговому вікні з повідомленням про помилку може бути присутня кнопка "Довідка" для виклику цього розділу.

Ще одним прикладом вирішення даної проблеми є кнопка "Докладніше", при натисканні на яку, діалогове вікно з повідомленням про помилку "розорюється", відображаючи більш докладну інформацію про причину виникнення збою.

Дуже важливо пам’ятати те, що повідомлення про помилку повинно містити її опис, а не числовий код помилки.

Існує багато програм, у яких повідомлення про помилки містять недостовірну інформацію, повідомляючи користувача зовсім не про ті проблеми, які виникли насправді. Тому при складанні описів помилок потрібно не забувати перевіряти коректність повідомлень, що генеруються програмою.

Відомо, що інформація про те, як виправити помилку або вирішити проблему має навіть більше значення, ніж опис помилки або проблеми. При описі шляху вирішення проблеми потрібно уникати складання занадто об’ємних текстів. В іншому разі користувачі будуть просто пробігати їх очами, не доходячи до змісту написаного. Найкраще скласти покрокову інструкцію, кожний крок якої складається з 1 – 2 речень.

10. Довідка й документація.

Це правило полягає в необхідності надання користувачеві довідкової системи й документації до програми.

Проектування форм

Форми – це "будівельні блоки" інтерфейсу користувача.

Щоб створити добре спроектовану форму, необхідно усвідомити її призначення, спосіб і час використання, а також її зв’язки з іншими елементами програми.

Особливий вид форм – форми, призначені для введення даних. При їхній розробці основну увагу варто приділити швидкості їхнього використання. Основне правило – якщо користувач збирається ввести в базу даних велику кількість записів, то він не повинен підтверджувати введення кожної з них.

Щоб прискорити процес уведення даних, необхідно:

1. По можливості використовувати для додавання й редагування даних ту саму форму.
2. Призначати клавіатурні сполучення для команд.

3. Не змушувати користувача "перестрибувати" з однієї частини форми в іншу.

4. Не ставити процес уведення даних у залежність від вмісту окремих елементів управління форми.

5. Використовувати засоби зворотного зв’язку з користувачем.

Робота з декількома формами

Якщо додаток повинен містити кілька форм, то можна використовувати однодокументний (SDI) або багатодокументний (MDI) інтерфейс користувача.

Однак, у кожному разі взаємодія користувача з формами відбувається за допомогою обробки подій, що надходять від елементів управління форми. Тому, якщо в додатку передбачено кілька форм, то програму необхідно спроектувати так, щоб у користувачів не було можливості порушити послідовність її виконання (наприклад, вивести форму, для якої ще не готова інформація).

Ефективні меню

Ще одна важлива частина розробки форм – створення змістовних і ефективних меню. Ось деякі важливі рекомендації:

1. Дотримуйтеся стандартних угод про розташування пунктів меню, прийнятих в операційній системі.

2. Групуйте пункти меню в логічному порядку й за змістом.

3. Для угруповання пунктів у меню, що розкриваються, використовуйте розділові лінії.

4. Уникайте надлишкових меню.

5. Уникайте пунктів меню верхнього рівня, які не мають меню, що розкриваються.

6. Не забувайте використовувати символ <…> для позначення пунктів меню, що активізують діалогові вікна.

7. Обов’язково використовуйте клавіатурні еквіваленти команд і "гарячі" клавіші.

8. Поміщайте на панель інструментів часто використовувані команди меню.

Далі наведені деякі рекомендації із проектування Web-інтерфейсу користувача:

1. Мінімізуйте зусилля, які необхідно зробити користувачеві для прийняття рішень про навігацію.

2. По можливості використовуйте один екран. Використовуйте багатопанельні елементи управління, спливаючі вікна й майстри, щоб

користувачі могли виконувати як можна більшу частину завдань, не використовуючи побічну навігацію.

3. При створенні декількох сторінок поєднуйте їх у розділи. Спростіть переміщення між розділами за допомогою основного елемента управління навігацією та додаткового елемента управління для переміщення між розділами.

4. Переконайтеся, що ключові елементи, такі як меню, заголовки й інша інформація, що відображається на всіх сторінках, оформлені одноманітно з візуальної точки зору.

5. Варто бути дуже обережним відносно навігаційних елементів управління для переміщення користувачів між внутрішніми сторінками різних розділів. Це можливо, але це може дезорієнтувати користувача.

6. Використовуйте додаткові засоби навігації, такі як дерева й карти вузлів, але не покладайтеся на них.

7. Завжди думайте про майбутнє розширення. Якщо в майбутньому очікується включення в продукт нових функцій, спочатку необхідно вирішити, як буде розширюватися навігація для ефективного включення нових екранів.

Підрозділ 3.4. Тестування додатка

При розробленні бізнес-додатка студент має провести модульне, інтеграційне та системне тестування.

У даному пункті пояснювальної записки повинен знаходитися опис процедур системного тестування та їхніх результатів.

Повинні бути описані такі види системного тестування:

1. Функціональне тестування.
2. Тестування безпеки.

Для опису процедури тестування повинні бути складені такі документи:

1. Тест-вимоги.
2. Тест-плани, сполучені зі звітами про проведення тестування.

У звіті про проведення тестування вказуються як позитивні так і негативні результати виконання окремих тестів.

Однак, загальний результат тестування додатка повинен бути позитивним, бо інакше він не відповідає певним вимогам. Для цього в разі необхідності проводяться додаткові заходи щодо виправлення помилок та тестування. Вони також повинні бути описані в даному пункті пояснювальної записки.

Далі наведені методичні рекомендації щодо тестування.

Тестування програми – процес виконання програмного коду, спрямований на виявлення існуючих у ньому дефектів. Під дефектом розуміється ділянка програмного коду, виконання якої за певних умов приводить до несподіваного поводження системи (тобто поводження, що не відповідає вимогам).

Завдання тестування – визначення умов, при яких проявляються дефекти системи, і протоколювання цих умов.

Мета застосування процедури тестування програмного коду – мінімізація кількості дефектів у кінцевому продукті.

Тестування саме по собі не може гарантувати повної відсутності дефектів у програмному коді системи. Однак, у сполученні із процесами верифікації й валідації, спрямованими на усунення суперечливості й неповноти проектної документації (зокрема – вимог на систему), грамотно організоване тестування дає гарантію того, що система задовольняє вимогам і поводиться відповідно до них у всіх передбачених ситуаціях.

Основні методи тестування

"Чорний ящик".

Основна ідея тестування системи як "чорного ящика" полягає в тому, що тестувальнику доступні тільки вимоги на систему, що описують її поводження, і сама система. Всі внутрішні особливості реалізації системи сховані. Аналіз системи полягає в подачі на її "вхід" деяких зовнішніх впливів і спостереженні результату, що з’являється на її "виході".

У програмній системі "чорний ящик" може становити набір класів (або модулів) з відомими зовнішніми інтерфейсами, але недоступними вихідними текстами.

Тестування за методом "чорного ящика" називають також тестуванням за вимогами.

"Скляний (білий) ящик".

При тестуванні програмної системи як "скляного ящика" тестувальник має доступ не тільки до вимог до системи, її входів і виходів, але й до її внутрішньої структури – програмного коду. Доступність програмного коду розширює можливості тестування.

Оскільки сучасні програмні системи мають досить значні розміри, при тестуванні їхнього програмного коду використовується метод

функціональної декомпозиції. Система розбивається на окремі модулі (класи, простори імен та ін.), що мають певну функціональність і інтерфейси. Після цього окремо тестується кожний модуль – виконується модульне тестування. Потім відбувається об’єднання окремих модулів у більші конфігурації – виконується інтеграційне тестування, і нарешті, тестується система в цілому – виконується системне тестування.

Модульне тестування

У ході модульного тестування кожний модуль тестується як відповідно вимогам, так і на відсутність проблемних ділянок програмного коду, які можуть викликати відмови й збої в роботі системи. Як правило, модулі не працюють поза системою – вони приймають дані від інших модулів, переробляють їх і передають далі. Для того, щоб з одного боку, ізолювати модуль від системи й виключити вплив потенційних помилок системи, а з іншого боку – забезпечити модуль всіма необхідними даними, використовується тестове оточення.

Завдання тестового оточення – створити середовище виконання для модуля, емулювати всі зовнішні інтерфейси, до яких звертається модуль.

Типова процедура тестування складається в підготовці й виконанні тестових прикладів. Кожний тестовий приклад перевіряє одну "ситуацію" у поводженні модуля й складається зі списку значень, переданих на вхід модуля, опису запуска й виконання переробки даних – тестового сценарію й списку значень, які очікуються на виході модуля у випадку його коректного поводження. Тестові сценарії складаються таким чином, щоб виключити звертання до внутрішніх даних модуля, вся взаємодія повинна відбуватися тільки через його зовнішні інтерфейси.

Виконання тестового приклада підтримується тестовим оточенням, що містить у собі програмну реалізацію тестового сценарію. Виконання починається з передачі модуля вхідних даних і запуску сценарію. Реальні вихідні дані, отримані від модуля в результаті виконання сценарію, зберігаються й порівнюються з очікуваними. У випадку їхнього збігу тест вважається пройденим, у іншому випадку – не пройденим.

Результатом тестування й верифікації окремих модулів, що становлять програмну систему, повинен бути висновок про те, що ці модулі є внутрішньо несуперечливими й відповідають вимогам.

Інтеграційне тестування

Окремі модулі рідко функціонують самі по собі, тому наступне завдання після тестування окремих модулів – тестування коректності взаємодії декількох модулів, об’єднаних у єдине ціле. Таке тестування називають інтеграційним.

Інтеграційне тестування називають ще тестуванням архітектури системи.

З одного боку, ця назва обумовлена тим, що інтеграційні тести містять у собі перевірки всіх можливих видів взаємодій між програмними модулями й елементами, які визначені в архітектурі системи. Таким чином, інтеграційні тести перевіряють повноту взаємодій у реалізації системи, що тестується.

З іншого боку, результати виконання інтеграційних тестів є одним з основних джерел інформації для процесу поліпшення й уточнення архітектури системи, міжмодульних та міжкомпонентних інтерфейсів. Із цього погляду, інтеграційні тести перевіряють коректність взаємодії компонент системи.

Системне тестування

По завершенню інтеграційного тестування всі модулі системи є погодженими по інтерфейсах і функціональності. Починаючи цього моменту можна переходити до системного тестування, тобто тестування поводження системи в цілому як єдиного об’єкта.

При системному тестуванні проводиться не тільки функціональне тестування, але й оцінка характеристик якості системи – її стійкості, надійності, безпеки й продуктивності. На цьому етапі виявляються багато проблем зовнішніх інтерфейсів системи, пов’язаних з неправильною взаємодією з іншими системами, апаратним забезпеченням, неправильним розподілом пам’яті, відсутністю коректного звільнення ресурсів та ін.

Як правило, для системного тестування застосовується метод "чорного ящика", при цьому як вхідні та вихідні дані використовуються реальні дані, з якими працює система, або дані, подібні їм.

Вхідною інформацією для проведення системного тестування є два класи вимог: функціональні й нефункціональні.

Функціональні вимоги явно описують, що система повинна робити і які перетворення вхідних даних виконувати.

Нефункціональні вимоги визначають властивості системи, прямо не пов’язані з її функціональністю. Прикладом таких властивостей може служити максимальний час відгуку системи на запит користувача, мінімальний час безперебійної роботи системи й ін.

Системне тестування проводиться у кілька етапів, на кожному з яких використовується один з видів системного тестування.

Види системного тестування:

1. Функціональне тестування.

Даний вид тестування призначений для підтвердження того, що вся система в цілому поводиться відповідно до очікувань користувача, формалізованих у вигляді системних вимог. У ході функціонального тестування перевіряються всі функції системи з погляду її користувачів (як людей, так і інших програмних систем). Також необхідно перевірити функціональну повноту користувальницького інтерфейсу й коректність виведення інформації.

При функціональному тестуванні система розглядається як "чорний ящик".

Критерії повноти функціонального тестування:

1. Усі функціональні вимоги повинні бути протестовані.
2. Усі класи припустимих вхідних даних повинні коректно оброблятися системою.
3. Усі класи неприпустимих вхідних даних повинні бути відкинуті системою, при цьому не повинна порушуватися стабільність її роботи.
4. У тестових прикладах повинні генеруватися всі можливі класи вихідних даних системи.
5. Під час тестування система повинна побувати у всіх своїх внутрішніх станах, пройшовши при цьому по всіх можливих переходах між станами.

2. Тестування продуктивності.

Даний вид тестування спрямований на визначення того, що система забезпечує належний рівень продуктивності при обробці користувальницьких запитів. Виділяють три основних фактори, що впливають на продуктивність системи: кількість підтримуваних системою потоків (наприклад, користувальницьких сесій), кількість вільних системних ресурсів, кількість вільних апаратних ресурсів.

Тестування продуктивності дозволяє виявляти вузькі місця в системі, які проявляються в умовах підвищеного навантаження або недостачі системних ресурсів. У цьому випадку за результатами тестування проводиться доробка системи, змінюються алгоритми виділення й розподілу ресурсів системи.

Всі вимоги, що ставляться до продуктивності системи, повинні бути чітко визначені й обов’язково повинні містити в собі числові оцінки параметрів продуктивності. Наприклад, вимога "Система повинна мати

прийнятний час відгуку на запит користувача" є непридатною для тестування. Навпаки, вимога "Час відгуку на запит користувача не повинен перевищувати 2 секунди" може бути протестована.

У звіті за даним видом тестування вказують такі показники, як завантаження апаратного й системного програмного забезпечення (кількість циклів процесора, виділеної пам'яті, кількість вільних системних ресурсів та ін.). Також важливі швидкісні характеристики системи, що тестується: кількість оброблених за одиницю часу запитів, часові інтервали між початком обробки кожного наступного запиту, рівномірність часу відгуку в різні моменти часу та ін.

3. Навантажувальне тестування.

Має багато загального з тестуванням продуктивності, однак його основне завдання – оцінити продуктивність і стійкість системи у випадку, коли для своєї роботи вона виділяє максимально доступну кількість ресурсів або коли вона працює в умовах їхньої критичної недостачі.

Основна мета навантажувального тестування – вивести систему з ладу, визначити ті умови, за яких вона не зможе далі нормально функціонувати.

Навантажувальне тестування дуже важливе при тестуванні Web-систем, рівень навантаження на які найчастіше дуже складно прогнозувати.

4. Тестування конфігурації.

Незважаючи на те, що в теперішній час особливості реалізації периферійних пристроїв приховуються відповідними драйверами операційних систем, проблеми сумісності (як програмної, так і апаратної) однаково існують.

У ході тестування конфігурації перевіряється, що програмна система коректно працює на всьому підтримуваному апаратному забезпеченні й разом з іншими програмними системами.

Також необхідно перевіряти, що система коректно обробляє проблеми, що виникають в устаткуванні, як штатні (наприклад, сигнал закінчення паперу в принтері), так і позаштатні (збій живлення).

5. Тестування безпеки.

Якщо програмна система призначена для зберігання або обробки даних, які становлять таємницю певного роду (наприклад, комерційну), то до властивостей такої системи, пред'являються підвищені вимоги. Виконання цих вимог перевіряється при тестуванні безпеки системи.

При використанні цього виду тестування перевіряється:

1. Організація авторизації й аутентифікації користувачів.
2. Коректність реєстрації в "журналі аудита" всіх подій системи, пов’язаних з безпекою.
3. Архітектура системи з погляду забезпечення захисту інформації від несанкціонованого доступу.
4. Наявність і повноту опису засобів забезпечення безпеки в документації на програмну систему.

6. Тестування надійності й відновлення після збоїв.

Для коректної роботи системи в будь-якій ситуації необхідно впевнитися в тому, що вона відновлює свою функціональність і продовжує коректно працювати після будь-якої проблеми, що перервала її роботу.

При тестуванні відновлення після збоїв імітуються збої устаткування, що оточує програмне забезпечення або збої програмної системи, викликані зовнішніми факторами. При аналізі поводження системи в цьому випадку необхідно звертати увагу на два фактори – мінімізацію втрат даних у результаті збоїв й мінімізацію часу між збоєм і продовженням нормального функціонування системи.

7. Тестування зручності використання.

Зручність використання визначає ступінь простоти доступу користувача до функцій системи, надаваним через користувальницький інтерфейс.

Зручність використання користувальницького інтерфейсу – показник його якості, що визначає кількість зусиль, необхідних для вивчення принципів роботи із програмною системою за допомогою даного інтерфейсу, її використання, підготовки вхідних даних і інтерпретації вихідних даних.

Як правило, при тестуванні зручності використання користувальницького інтерфейсу використовуються деякі евристичні критерії й характеристики.

У результаті виконання всіх розглянутих раніше видів тестування робиться висновок про функціональність і властивості системи, після чого вузькі місця системи допрацьовуються до реалізації необхідної функціональності або до досягнення системою необхідних властивостей.

У ході системного тестування не завжди застосовуються всі з перерахованих видів тестування. Конкретний їхній набір залежить від конкретної програмної системи.

Документування процедури тестування

Основне призначення документації, створюваної при тестуванні, – забезпечення гарантії того, що процес тестування виконується з необхідною якістю й всі аспекти поводження системи протестовані.

Перелік необхідної документації:

1. Тест-вимоги.
2. Тест-план.
3. Звіт про тестування.

Тест-вимоги розробляються на підставі системних і функціональних вимог до додатка. У них докладно описується, які аспекти поводження системи повинні бути протестовані, щоб упевнитися в її коректному функціонуванні, і на підставі якого зовнішнього ефекту можна переконатися, що функціональність, яка перевіряється, реалізована правильно.

Тест-вимоги повинні бути достатніми для побудови тест-плану перевірки програмної системи без знайомства з її програмним кодом.

Структура тест-вимог повинна додержуватися структури розділу функціональних вимог на систему. Як правило, одній системній або функціональній вимозі відповідає мінімум одна тест-вимога.

Класифікація тест-вимог за призначенням:

1. Контроль вхідних даних.
2. Обробки помилок уведення даних і обробки інформації.
3. Одержання основного результату.
4. Оформлення й виведення результатів.

Для кожної тест-вимоги повинна існувати можливість перевірки – виконується ця вимога в реалізованій системі чи ні.

На підставі тест-вимог створюється тест-план – документ, що містить докладний покроковий опис того, як повинні бути протестовані тест-вимоги. На відміну від тест-вимог, у тест-плані описуються конкретні способи перевірки функціональності системи.

Як правило, тест-план складається з окремих тестових прикладів, кожний з яких перевіряє деяку функцію або набір функцій системи. Для кожного тестового приклада однозначно визначається критерій успішного проходження, за допомогою якого можна судити про відповідність поводження системи заданому.

Структура тест-плану повинна відповідати структурі тест-вимог.

Кожний пункт тест-плану повинен містити:

1. Посилання на вимогу(и), що перевіряється цим пунктом.
2. Конкретне значення вхідних даних.

3. Очікувану реакцію програми (тексти повідомлень, значення результатів).

4. Опис послідовності дій, необхідних для виконання пунктів тест-плану.

При ручному тестуванні зручним є подання тест-плану у вигляді текстового документа, у якому окремі розділи становлять описи тестових прикладів. Кожний тестовий приклад повинен містити перерахування послідовності дій, які необхідно виконати для проведення тестування, а також очікувані відгуки системи на ці дії.

За результатами виконання тестів створюється звіт про виконання тестування. Він є основним джерелом для висновку про ступінь відповідності протестованої системи вимогам. Такий звіт, як мінімум, повинен містити інформацію про кожний виконаний тестовий приклад і результат його виконання (успіх або невдача).

Іноді тест-план сполучають зі звітом про проведення тестування, додаючи до нього інформацію про отриману реакцію системи й збіг (розбіжності) отриманих результатів з очікуваними. Наприкінці опису кожного тестового приклада додається інформація про те, чи пройдений тестовий приклад у цілому. Наприкінці всього тест-плану, сполученого зі звітом, міститься графа "Тестових прикладів пройдене/усього", у яку заноситься число пройдених тестових прикладів і загальна їхня кількість.

Приклад тест-плану, сполученого зі звітом про проведення тестування наведений у додатку П.

Підрозділ 3.5. Розгортання програмного продукту

Опис апаратних та програмних засобів, необхідних для функціонування розробленого програмного продукту, дій щодо його інсталяції на комп’ютері користувача.

5. Порядок подання до захисту та захист дипломного проекту

5.1. Попередній захист дипломного проекту

З метою виявлення готовності студента до захисту виконується попередній захист дипломного проекту. Попередній захист складається з двох частин:

доповідь з презентацією за повністю виконаним проектом;

демонстрація роботи програмного продукту.

Мета попереднього захисту – перевірка готовності студента до захисту відповідно до вимог випускаючої кафедри, оцінка обсягу поданого проекту і якості його виконання й оформлення. Незалежно від ступеня готовності проекту, студент повинен з’явитися на попередній захист.

Для проведення попереднього захисту кафедра ІС визначає склад комісій та складає графік попереднього захисту.

На попередній захист подаються:

оформлене і своєчасно затверджене завдання;

повністю оформлена, але не зшита пояснювальна записка з підписами студента, керівника і консультантів на завданні;

готовий програмний продукт та відеоролик його роботи;

план доповіді та презентація, узгоджені з керівником.

На підставі доповіді студента, його відповідей на питання, результатів перевірки пояснювальної записки, презентації комісія визначає рекомендації студенту:

щодо доповіді;

щодо відповідей на запитання;

щодо змісту пояснювальної записки;

щодо оформлення пояснювальної записки;

щодо презентації;

щодо демонстрації програмного додатку.

Допуск до захисту можливий при позитивному оцінюванні за обома видами попереднього захисту (пояснювальна записка; програмна частина). Студенти, що не пройшли попередній захист, не допускаються до захисту.

5.2. Подання дипломного проекту до захисту

Після попереднього захисту та усунення недоліків закінчений дипломний проект подається керівнику проекту.

Він остаточно перевіряє відповідність виконаної роботи завданню та відповідним вимогам, складає письмовий відзив, у якому дає характеристику роботи студента.

Цілком оформлений дипломний проект, що підписаний його керівником, проходить нормоконтроль. Після нормоконтролю дипломний проект підписує завідувач кафедрою і разом з відзивом керівника направляє на рецензування.

5.3. Захист дипломного проекту

Не пізніше ніж за добу до захисту студент подає дипломний проект секретарю державної екзаменаційної комісії (ДЕК). Обов'язковим є роздавальний матеріал щодо виконаної роботи для кожного члена ДЕК, який містить роздруківку слайдів презентації.

У ДЕК можуть бути подані інші матеріали, які характеризують наукову та практичну цінність виконаного дипломного проекту, а саме:

- друковані статті за темою роботи;
- документи, які характеризують практичну цінність розробки студента;
- документи, що вказують на практичне застосування роботи (підписані офіційними особами);
- макети, зразки виробів та ін.

Захист дипломних проектів проводиться на засіданні ДЕК.

Захист одного дипломного проекту, як правило, не має перевищувати 20 – 25 хвилин. Для доповіді щодо проекту студенту надається не більше 10 хвилин.

Захист комплексного дипломного проекту, як правило, планується і проводиться на одному засіданні ДЕК, причому студенту, який захищається першим, доручається доповісти як про загальну частину роботи, так і про індивідуальну частину зі збільшенням (за необхідності) часу на доповідь. Усі студенти, які виконували комплексну роботу, повинні бути повною мірою обізнані з загальною частиною роботи і готові до запитань членів комісії не тільки з індивідуальної, а й з загальної частини роботи.

Доповідь студента має складатися з трьох основних частин, а саме: вступу, основної частини та висновків.

У вступі необхідно зазначити актуальність теми проекту, дати загальний аналіз стану проблеми і сформулювати основні задачі, з вирішенням яких було пов'язано виконання проекту.

У основній частині доповіді у стислій формі необхідно навести звіт про зміст виконаних розробок, показати ефективність прийнятих технічних рішень, навести короткий звіт з отриманих результатів.

У заключній частині доповіді необхідно зробити загальні висновки і дати рекомендації щодо можливої області застосування об'єкта проектування, перелічити публікації за темою роботи, навести відомості про впровадження.

Доповідь повинна супроводжуватися посиланнями на електронну презентацію, яка демонструється студентом. Презентація повинна містити таки слайди:

титульний слайд з вихідними даними щодо дипломного проекту;

зміст презентації (з посиланнями на відповідні слайди);

актуальність теми та мету дипломного проекту;

модель організаційної структури підприємства, підрозділу підприємства;

модель дерева функцій;

модель управління бізнес-процесом;

діаграму варіантів використання;

розкадрування, що реалізують основну функціональність;

математичну модель;

заповнені форми вихідних та вхідних документів, діаграми, карти;

логічну та фізичну моделі бази даних;

UML-діаграму класів, що реалізують основну бізнес-логіку програмної системи;

UML-діаграму станів, у яких можуть знаходитися елементи графічного інтерфейсу користувача;

результати функціонального тестування програмного забезпечення та тестування його безпеки;

використані інструментальні засоби та технології;

висновки по результатам дипломного проекту;

апробацію результатів дипломного проекту;

заключний слайд.

При захисті може додатково використовуватись демонстраційний матеріал у вигляді відеоролика.

Після доповіді студент стисло відповідає на запитання членів ДЕК.

Далі зачитується рецензія і відзив керівника проекту. Студенту надається можливість відповісти на зауваження рецензента.

Після закінчення захисту всіх заявлених студентів комісія проводить закрите обговорення кожного захисту і оцінює його відповідно до критеріїв оцінювання. При цьому приймається до уваги рівень виконаної роботи та розробленого програмного продукту, якість оформлення пояснювальної записки, рівень наукової, практичної та теоретичної підготовки студента, ритмічність роботи над проектом, наявність публікацій, виступів на конференціях тощо.

Результати захисту дипломного проекту визначаються оцінками за дванадцятибальною шкалою та доводяться до відома студентів після завершення роботи ДЕК.

Рекомендована література

Основна

1. Галузевий стандарт вищої освіти України з напряму підготовки 6.050101 "Комп'ютерні науки" : збірник нормативних документів вищої освіти. – К. : Видавнича група BHV, 2011. – 85 с.

2. Комплекс нормативних документів для розробки складових системи стандартів вищої освіти. Додаток 1 до наказу Міносвіти України від 31.07.1998 р. № 285 із змінennями та доповненнями, уведеними розпорядженням Міністерства освіти і науки України від 05.03.2001 р. № 28-р. // Інформаційний вісник "Вища освіта". – 2003. – № 10. – 82 с.

3. Про вищу освіту Закон України. № 2984-III // Відомості Верховної Ради. – 2002. – № 20. – 134 с.

Додаткова література

4. Автоматизированные информационные системы, базы и банки данных. Вводный курс : учебн. пособ. – М. : Гелиос АРВ, 2002. – 368 с.

5. Анфилатов В. С. Системный анализ в управлении. / В. С. Анфилатов, А. А. Емельянов, А. А. Кукушкин – М. : Финансы и статистика, 2002. – 468 с.

6. Астелс Д. Практическое руководство по экстремальному программированию / Д. Астелс, Г. Миллер, М. Новак ; пер. с англ. – М. : ИД "Вильямс", 2002. – 320 с.

7. Баранов О. А. Інформаційне право України: стан, проблеми, перспективи / О. А. Баранов. – К. : ВД "СофтПрес", 2005. – 316 с.

8. Басс Л. Архитектура программного обеспечения на практике / Л. Басс, П. Клементс, Р. Кацман. – СПб. : Питер, 2006 – 576 с.

9. Бек К. Экстремальное программирование / К. Бек. – СПб. : Питер, 2002. – 224 с.

10. Береза А. М. Основи створення інформаційних систем : навч. посіб. / А. М. Береза – К. : КНЕУ, 2001. – 214 с.

11. Блэк Р. Ключевые процессы тестирования. Планирование, подготовка, проведение, совершенствование / Р. Блєк. – М. : Лори, 2011. – 544 с.

12. Богданов В. В. Управление проектами в Microsoft Project / В. В. Богданов. – СПб. : Питер, 2004. – 604 с.

13. Бондаренко М. Ф. Моделирование и проектирование бизнес-систем: методы, стандарты, технологии : учебн. пособ. / М. Ф. Бондаренко, С. И. Маторин, Е. А. Соловьев. – Х. : Компания СМИТ, 2004. – 272 с.

14. Брауде Э. Технологии разработки программного обеспечения / Э. Брауде – СПб. : Питер, 2004. – 655 с. : ил.

15. Веретенников В. І. / Управління проектами : навч. посібн. В. І. Веретенников, Л. М. Тарасенко, Г. І. Гевлич. – К. : Центр навчальної літератури, 2006. – 280 с.

16. Винниченко И. Автоматизация процессов тестирования / И. Винниченко. – СПб. : Питер, 2005 – 203 с.

17. Вигерс К. Разработка требований к программному обеспечению / К. Вигерс ; пер. англ. – М. : Изд.-торговый дом "Русская редакция", 2004. – 576 с.

18. Гайфуллин Б. Н. Автоматизированные системы управления предприятиями стандарта ERP/MRPII. Производственное издание / Б. Н. Гайфуллин, И. А. Обухов. – М. : "Богородский печатник", 2001. –104 с.

19. Гейзер Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем. Б. Гейзер. – СПб. : Питер, 2004 – 320 с.

20. Объектно-ориентированный анализ и проектирование с примерами приложений. / Г. Буч, А. Роберт, Максимчук, У. Майкл. и др. – М. : ИД "Вильямс", 2010. – 720 с.

21. Горбатов В. С. Основы технологии PKI. / В. С. Горбатов, О. Ю. Полянская. – М. : Горячая линия – Телеком, 2004. – 248 с. : ил.

22. Годин В. В. Управление информационными ресурсами / В. В. Годин, И. К. Корнеев. – М. : ИНФРА-М, 2000. – 352 с.

23. ГОСТ 19.701-90. Схемы алгоритмов, данных, программ и систем. Условные обозначения и правила выполнения. – М. : Изд. стандартов, 1990. – 16 с.

24. ГОСТ 34.201-89. Виды, комплектность и обозначение документов при создании автоматизированных систем. – М. : Изд. стандартов, 1989. – 16 с.

25. ГОСТ 34.601-90. Автоматизированные системы. Стадии создания. – М. : Изд. стандартов, 1990. – 12 с.

26. ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы. – М. : Изд. стандартов, 1990. – 24 с.

27. Гужва В. М. Інформаційні системи і технології на підприємствах: навч. посіб. / В. М. Гужва. – К. : КНЕУ, 2001. – 400 с.

28. ДСТУ 2938-94. Системи оброблення інформації. Основні поняття. Терміни та визначення. – К. : Держстандарт України, 1995. – 32 с.

29. ДСТУ 2940-94. Системи оброблення інформації. Керування процесами оброблення даних. Терміни та визначення. – К. : Держстандарт України, 1995. – 28 с.

30. ДСТУ 2941-94. Системи оброблення інформації. Розробки систем. Терміни та визначення. – К. : Держстандарт України, 1995. – 20 с.

31. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Правила оформлення. – К. : Держкомстат України, 1995. – 28 с.

32. ДСТУ 3918-99 (ISO/IEC 12207:1995). Інформаційні технології. Процеси життєвого циклу програмного забезпечення. – К. : Держкомстат України, 1999. – 48 с.

33. ДСТУ ISO/IEC TR 15504-1-2002 (Частини 1–9). Інформаційні технології. Оцінювання процесів програмування. – К. : Держспоживстандарт України, 2002. – 42 с.

34. Дейт К. Дж. Введение в системы баз данных / К. Дж. Дейт. – 8-е изд. – М. : Вильямс, 2005. – 1328 с.

35. Д. О'Лири. ERP системы. Современное планирование и управление ресурсами предприятия. Выбор, внедрение, эксплуатация / О'Лири Дэниел. – М. : ООО "Вершина", 2004. – 272 с.

36. Елиферов В. Г. Бизнес-процессы: Регламентация и управление: учебник / В. Г. Елиферов, В. В. Репин. – М. : ИНФРА-М, 2004. – 320 с.

37. Єрьоміна Н. В. Банківські інформаційні системи : навч. посібн. / Н. В. Єрьоміна. – К. : КНЕУ, 2000. – 272 с.

38. Зима В. М. Безопасность глобальніх сетевых технологий. – 2-е изд. / В. М. Зима, А. А. Молдовян, Н. А. Молдовян. – СПб. : БХВ-Петербург, 2003. – 368 с. : ил.

39. Золотарьова І. О. Інформаційні системи та технології в банківській сфері : навч. посібн. / І. О. Золотарьова, Р. К. Бутова, А. А. Гаврилова. – Х. : Вид. ХНЕУ, 2009. – 332 с. (укр. мов.)

40. Інформатизація бізнесу: концепції, технології, системи / А. М. Карминский, С. А. Карминский, В. П. Нестеров та ін.; під ред. А. М. Карминского. – М. : Фінанси й статистика, 2004. – 624 с.

41. Інформаційні системи в економіці / за ред. В. С. Пономаренка. – К. : Академія, 2002. – 542 с.

42. Кальянов Г. Н. CASE-технологии. Консалтинг в автоматизации бизнес-процессов / Г. Н. Кальянов. – М. : Горячая линия – Телеком, 2002. – 320 с.

43. Каменова М. Моделирование бизнеса. Методология ARIS. / М. Каменова, А. Громов, М. Ферапонтов и др. – М. : Весть-Мета Технология, 2001. – 328 с.

44. С. Канер. Тестирование программного обеспечения / С Канер, Дж. Фолк, У. К. Нгуен. – К. : "ДиаСофт", 2001. – 544 с.

45. Карминский А. М. Информатизация бизнеса / А. М. Карминский, А. С. Карминский, В. П. Нестеров и др. – М. : Финансы и статистика, 2004. – 624 с.

46. Карпова Т. Базы данных. Модели, разработка, реализация : учебник / Т. Карпова. – СПб. : 2001. – 302 с.

47. Конноли Т. Базы данных: проектирование, реализация и сопровождение учебн. пособ. / Т. Конноли // Теория и практика, 2-е изд. : пер. с англ. – М. : ИД "Вильямс", 2000. – 1120 с.

48. Коберн А. Быстрая разработка программного обеспечения / А. Корбен. – М. : Лори, 2002. – 314 с.

49. Коберн А. Современные методы описания функциональных требований к системам / А. Коберн. – М. : Издательство "Лори", 2002. – 263 с.

50. Коробов П. Н. Математическое программирование и моделирование экономических процессов / П. Н. Коробов. – М. : ДНК, 2010. – 376 с.

51. Кузнецов О. О. Захист інформації в інформаційних системах. Методи традиційної криптографії / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Х. : Вид. ХНЕУ, 2010. – 316 с.

52. Кузнецов О. О. Захист інформації в інформаційних системах / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Х. : Вид. ХНЕУ, 2011. – 504 с.

53. Кузнецов О. О. Захист інформації та економічна безпека підприємства / О. О. Кузнецов, С. П. Євсеєв, С. В. Кавун. – Х. : Вид. ХНЕУ, 209. – 360 с.

54. Кузнецов О. О. Сигнали і коди. Алгебраїчні методи синтезу / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Х. : Вид. ХНЕУ, 2009. – 384 с.

55. Ленков С. В. Методы и средства защиты информации. В 2-х томах / С. В. Ленков, Д. А. Перегудов, В. А. Хорошко ; под ред. В. А. Хорошко. – К. : Арий, 2008. – Т. I. Несанкционированное получение информации. – 464 с. : ил.

56. Ленков С. В. Методы и средства защиты информации. В 2-х томах / С. В. Ленков, Д. А. Перегудов, В. А. Хорошко. Под ред. В. А. Хорошко. – К. : Арий, 2008. – Т.II. Информационная безопасность. – 344 с. : ил.

57. Леоненков А. В. Самоучитель UML / А. В. Леоненков. – СПб. : БХВ-Петербург, 2004. – 432 с.

58. Леффингуелл Д. Принципы работы с требованиями к программному обеспечению / Д. Леффингуелл, Д. Уидриг. – М. : ИД "Вильямс", 2002. – 448 с.

59. Лодон Дж. Управление информационными системами / Дж. Лодон, К. Лодон ; пер. с англ. под. ред. Д. Р. Трутнева. – 7-е изд. – СПб. : Питер, 2005. – 912 с.

60. Лямец В. И. Системный анализ. Вводный курс / В. И. Лямец, А. Д. Тевяшев. – Х. : ХНУРЭ, 2004. – 448 с.

61. Маглинец Ю. А. Анализ требований к автоматизированным информационным системам / Ю. А. Маглинец. – М. : БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий – ИНТУИТ. ру, 2008. – 200 с.

62. Макарова Н. В. Информатика : учебн. / Н. В. Макарова. – М. : Финансы и статистика, 2003. – 768 с.

63. Маклаков С. В. Создание информационных систем с AllFusion Modeling Suit / С. В. Маклаков. – М. : Москва ДиалогМифи, 2003. – 432 с.

64. Мацяшек Лешек. Анализ требований и проектирование систем. / Лешек Мацяшек ; пер. с англ. – М. : Издательский дом "Вильямс", 2002. – 432 с.

65. Меняев М. Ф. Информационные технологии управления: Книга 3: Системы управления организацией. / М. Ф. Меняев. – М. : Омега-Л, 2003. – 464 с

66. Методичні рекомендації до виконання комплексного курсового проекту для студентів спеціальності "Інформаційні управляючі системи і технології" всіх форм навчання / укл. І. О. Золотарьова, С. В. Мінухін, Ю. Е. Парфьонов, Р. К. Бутова, Т. О. Свердло. – Х. : Вид. ХНЕУ, 2010. – 82 с.

67. Методичні рекомендації до оформлення звітів, курсових та дипломних проектів для студентів напряму підготовки 0804 "Комп'ютерні науки" всіх форм навчання / укл. : І. О. Золотарьова, О. М. Беседовський, І. Л. Латишева, Г. О. Плеханова. – Х. : Вид. ХНЕУ, 2007. – 32 с. (укр. мов.)

68. Ойхман Е. Г. Реинжиниринг бизнеса: реинжиниринг организаций и информационные технологии / Е. Г. Ойхман, Е. В. Попов. – М. : Финансы и статистика. 1997. – 336 с.

69. Орлова И. В. Экономико-математические методы и модели. Компьютерное моделирование / И. В. Орлова, В. А. Половников. – М. : Инфра-М, 2011. – 368 с.

70. Орлов С. Технологии разработки программного обеспечения : учебник / С. Орлов. – СПб. : Питер, 2002. – 464 с.

71. Петров А. А. Компьютерная безопасность. Криптографические методы защиты / А. А. Петров. – М. : ДМК, 2000. – 448 с. : ил.

72. Петров В. Н. Информационные системы / В. Н. Петров. – СПб. : Питер, 2002. – 688 с.

73. Поповский В. В. Защита информации в елекоммуникационных системах: учебник : в 2 т. / В. В. Поповский, А. В. Персиков. – Х. : ООО "Компания СМИТ", 2006. – Т.1. – 292 с.

74. Поповский В. В. Защита информации в елекоммуникационных системах : учебник : в 2 т. / В. В. Поповский, А. В. Персиков. – Х. : ООО "Компания СМИТ", 2006. – Т.2. – 252 с.

75. Принципы проектирования и разработки программного обеспечения : учебный курс MCSD / пер. с англ. – 2-е изд., испр. – М. : Издательско-торговый дом "Русская Редакция", 2002. – 736 с.

76. РД 50-34.698-90. Руководящий документ за стандартизации. Методические указания. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов. – М. : Изд. стандартов, 1991. – 40 с.

77. Смирнова Г. Н. Проектирование экономических информационных систем : учебник / Г. Н. Смирнова, А. А. Сорокин, Ю. Ф. Тельнов ; под ред. Ю. Ф. Тельнова. – М. : Финансы и статистика, 2001. – 512 с.

78. Соммервилл И. Инженерия программного обеспечения, 6-е издание. / И. Соммервилл. – М. : Издательский дом "Вильямс", 2002. – 624 с.

79. Татарчук М. І. Корпоративні інформаційні системи : навч. посібн. / М. І. Татарчук. – К. : КНЕУ, 2005. – 291 с.

80. Ушакова І. О. Інформаційні системи і технології в статистиці / І. О. Ушакова. – Х. : Вид. ХНЕУ, 2006. – 164 с.

81. Ушакова І. О. Основи системного аналізу об’єктів та процесів комп’ютеризації : навч. посібн. Частина 1 / І. О. Ушакова. – Х. : Вид. ХНЕУ, 2007. – 212 с.

82. Ушакова І. О. Основи системного аналізу об’єктів та процесів комп’ютеризації : навч. посіб. Частина 2. / І. О. Ушакова. – Х. : Вид. ХНЕУ, 2008. – 324 с.

83. Ушакова І. О. Практикум з навчальної дисципліни "Основи системного аналізу об’єктів і процесів комп’ютеризації" : навчально-практ. посібн. / І. О. Ушакова, Г. О. Плеханова. – Х. : Вид. ХНЕУ, 2010. – 344 с.

84. Фаулер М. UML в кратком изложении. Применение стандартного языка объектного моделирования / М. Фаулер, К. Скотт. – М. : Мир, 1999. – 191 с.

85. Шафер Д. Ф. Управление программными проектами: достижение оптимального качества при минимуме затрат / Д. Ф. Шафер, Р. Т. Фатрел, Л. И. Шафер. – М. : Издательский дом "Вильямс", 2003. – 1136 с.

86. Якобсон А. Унифицированный процесс разработки програмного обеспечения / А. Якобсон, Г. Буч, Дж. Рамбо. – СПб. : Питер, 2002. – 496 с.

87. Столлингс В. Криптография и защита сетей: принципы и практика. – 2-е изд. / В. Столлингс ; пер. с англ. – М. : Издательский дом "Вильямс", 2001. – 672 с. : ил.

88. Щеглов А. Ю. Защита компьютерной информации от несанкционированного доступа / А. Ю. Щеглов. – СПб. : Наука и Техника, 2004. – 384 с. : ил.

89. Вербіцький О. В. Вступ до криптології / О. В. Вербіцький. – Л. : ВНТЛ, 1998. – 248 с.

90. Цифровая стеганографія / Грибунин В. Г., Оков И. Н., Туринцев И. В. – М. : СОЛОН-Прес, 2002. – 272 с.

91. Чмора А. Л. Современная прикладная криптография / А. Л. Чмора. – М. : Гелиос АРВ, 2001. – 256 с. : ил.

92. Хорошко В. А. Методы и средства зашиты информации / В. А. Хорошко, А. А. Чекатков. – К. : Юниор, 2003. – 504 с.

93. Хомоненко А. Д. Базы данных : учебник для высших учебных заведений / А. Д Хомоненко, В. М. Цыганков, М. Г. Мальцев. – СПб. : Корона, 2004. – 736 с.

94. Бази даних у питаннях і відповідях: навч. посібн. / В. В. Чубук, Р. М. Чен, Л. А. Павленко та ін. – Х. : Вид. ХНЕУ, 2004. – 288 с.

Ресурси мережі Інтернет

95. Приклади оформлення бібліографічного опису у списку джерел, який наводять у дисертації, і списку опублікованих робіт, який наводять в авторефераті [Електронний ресурс]. Бюлетень ВАК України, № 3, 2008 (Форма 23, С. 9 – 13)). – Режим доступу : http://www.kntu.kr.ua/doc/bibl_opis.pdf ; http://www.kntu.kr.ua/doc/bibl_opis.pdf.

96. Самое интересное о разработке программного обеспечения [Электронный ресурс]. – Режим доступа : http://www.maxkir.com/.

97. CNews. Издание о высоких технологиях [Электронный ресурс]. – Режим доступа: http://www.cnews.ru/.

98. Интернет-университет информационных технологий [Электронный ресурс]. – Режим доступа: http://www.intuit.ru/.

99. Информационный портал CRM [Электронный ресурс]. – Режим доступа : www.crm.com.ua.

100. CITForum.ru. IT-портал [Электронный ресурс]. – Режим доступа : http://www.citforum.ru/.

101. Корпоративне управління [Електронний ресурс]. – Режим доступу : http://www.corporation.com.ua/.

102. Корпоративный менеджмент [Электронный ресурс]. – Режим доступа : http://www.cfin.ru/.

103. Корпорация "Галактика". Информационные технологи управления [Электронный ресурс]. – Режим доступа : http://www.ga-laktika.ru/.

104. Открытые системы. – Режим доступа : http://www.osp.ru/.

105. Офіційній сайт Держкомстату України [Електронний ресурс]. – Режим доступу: // www.ukrstat.gov.ua.

106. Портал "Информационно-коммуникационные технологии в образовании" [Электронный ресурс]. – Режим доступа : http://www.ict.edu.ru/.

107. Портал "Профессионал управления проектами" [Электронный ресурс]. – Режим доступа : http://www.pmprofy.ru/.

108. Сайт компанії IBM [Електронний ресурс]. – Режим доступу : http://www.ibm.com/ru/ru/.

109. Сайт компанії Gartner, Inc. (NYSE: IT) [Електронний ресурс]. – Режим доступу : http://www.gartner.com/.

110. Сайт компанії Interface: Internet and software company [Електронний ресурс]. – Режим доступу : http://www.interface.ru/.

111. Сайт компанії Microsoft [Електронний ресурс]. – Режим доступу : http: //www.microsoft.com/.

112. BYTE-Россия – журнал для ИТ-профессионалов [Электронный ресурс]. – Режим доступа : http://www.bytemag.ru/.

113. ERP-эксперт – Всё о ERP, ERP II, MRP, MRP II. – Режим доступу : http://erp-expert.narod.ru/index.htm.

114. ITC Online [Электронный ресурс]. – Режим доступа : http://itc.ua/.

115. The official UML Web site. – Режим доступу : http://www.uml.org.

116. Ten Usability Heuristics // : Web-сайт Якоба Нильсена. – Режим доступу : http://www.useit.com/papers/.

Додатки

Додаток А

Зразок заяви студента на затвердження теми дипломного проекту

Завідувачу кафедри
інформаційних систем
проф. Пономаренку В. С.
студента 4 курсу <номер> групи
<П.І.Б студента>

ЗАЯВА

Прошу затвердити мені тему дипломного проекту <Назва теми>.

<Дата> <Підпис студента> <П.І.Б студента>

Керівник
дипломного проекту
<Посада
наук. ступінь, вчене звання> <Підпис керівника> <П.І.Б. керівника>

Додаток Б

Зразок завдання на дипломний проект практичного характеру

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Факультет економічної інформатики
Кафедра інформаційних систем
Спеціальність "Інформаційні управляючі системи і технології"

ЗАТВЕРДЖУЮ:
завідувач кафедри ІС
_____
проф. Пономаренко В.С.
" "_____20__ р.

ЗАВДАННЯ
на дипломний проект студента
Іванова Івана Васильовича

1. Тема проекту: <Назва теми>
затверджена наказом по університету від " " _____ 20__ р. № ___
2. Термін здачі студентом закінченого проекту: " " _____ 20__ р.
3. Вхідні дані до проекту: ДСТУ щодо обробки інформації, літературні джерела, матеріали практики.
4. Зміст пояснювальної записки (перелік питань, що підлягають розробленню): Вступ. 1. Аналіз об’єкта проектування. 2. Розроблення вимог до програмного забезпечення. 3. Проектні та технічні рішення.
5. Перелік графічного матеріалу: організаційна структура підприємства та структурного підрозділу (ARIS); діаграма "дерево функцій" (ARIS); схема бізнес-процесів (ARIS); UML-діаграма варіантів використання, UML-діаграма класів додатку, UML-діаграма станів.

Календарній план

№ з/п	Назва етапів виконання проекту	Термін виконання етапів	Примітки
1	Аналіз об’єкта проектування		
2	Розроблення вимог до програмного забезпечення		
3	Проектні та технічні рішення		
4	Підготовка пояснювальної записки		
5	Підготовка презентації та доповіді		
6	Попередній захист		
7	Нормоконтроль, рецензування		
8	Занесення диплома в електронний архів		
9	Допуск до захисту у зав. кафедрою		

Дата видачі завдання " "_____ 20__ р.

Керівник к.т.н., доц. кафедри ІС_____ Ю. С. Петренко
(підпис)

Завдання прийняв до виконання _____ І. В. Іванов
(підпис)

Додаток В

Зразок завдання на дипломний проект дослідницького характеру

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Факультет економічної інформатики
Кафедра інформаційних систем
Спеціальність "Інформаційні управляючі системи і технології"

ЗАТВЕРДЖУЮ:
завідувач кафедри ІС
_____
проф. Пономаренко В.С.
" " _____ 20__ р.

ЗАВДАННЯ
на дипломний проект студента
Іванова Івана Васильовича

1. Тема проекту: *"Методи використання програмних агентів у семантичній "павутині""* затверджена наказом по університету від " " _____ 20__ р. № ___
2. Термін здачі студентом закінченої роботи: " " _____ 20__ р.
3. Вхідні дані до роботи: *Передбачити в роботі аналіз стану досліджень по вдосконаленню мережі Інтернет, розглянути архітектуру семантичної "павутини", напрями використання програмних агентів в мережі Інтернет, існуючі підходи до побудови програмних агентів у семантичній "павутині", дослідити технології використання програмних агентів в семантичній "павутині", розробити методику дослідження ефективності використання пошукових програмних агентів, виконати аналіз показників ефективності використання пошукового програмного агенту. У процесі розроблення програмного засобу використати мову програмування C# та середовище MS Visual Studio 2010, технологію ASP.NET.*

4. Зміст пояснювальної записки (перелік питань, що підлягають розробленню): Вступ. 1. *Аналіз сучасного стану досліджень щодо вдосконалення мережі Інтернет*. 2 *Використання пошукових програмних агентів в семантичній "павутині"*. 3. *Дослідження ефективності використання пошукового програмного агенту в семантичній "павутині"*. Висновки. Додатки.

Календарній план

№ з/п	Назва етапів виконання проекту	Термін виконання етапів	Примітки
1	*Аналіз сучасного стану досліджень щодо вдосконалення мережі Інтернет*		
2	*Використання пошукових програмних агентів в семантичній "павутині"*		
3	*Дослідження ефективності використання пошукового програмного агенту в семантичній "павутині"*		
4	Підготовка пояснювальної записки		
5	Підготовка презентації та доповіді		
6	Попередній захист		
7	Нормоконтроль, рецензування		
8	Занесення диплома в електронний архів		
9	Допуск до захисту у зав. кафедрою		

Дата видачі завдання " " _____ 20__ р.

Керівник к.т.н., доц. кафедри ІС_____ Ю. С. Петренко
(підпис)

Завдання прийняв до виконання _____ І. В. Іванов
(підпис)

Примітка:

1. *Зміст тексту, що виділений курсивом, залежить від теми дипломного проекту.*

Додаток Д

Приклад змістовної частини дипломного проекту дослідницького характеру

1. АНАЛІЗ СУЧАСНОГО СТАНУ ДОСЛІДЖЕНЬ З ПРОБЛЕМИ СЕГМЕНТУВАННЯ БАЗИ ДАНИХ КЛІЄНТІВ
1.1. Необхідність сегментування бази даних клієнтів
1.2. Аналіз існуючих методів сегментування
2. МЕТОДИКА ВИБОРУ МЕТОДУ СЕГМЕНТУВАННЯ БАЗИ ДАНИХ КЛІЄНТІВ
2.1. Критерії вибору методу сегментування бази даних клієнтів
2.2. Етапи методики вибору методу сегментування бази даних клієнтів
3. РОЗРОБЛЕННЯ ПРАКТИЧНИХ РЕКОМЕНДАЦІЙ ЩОДО ВИБОРУ МЕТОДУ СЕГМЕНТУВАННЯ БАЗИ ДАНИХ КЛІЄНТІВ
3.1. Опис бази клієнтів, що аналізується
3.2. Аналіз результатів сегментування бази даних клієнтів
3.3. Висновки щодо вибору методу сегментування

Додаток Е

Структура змістовної частини дипломного проекту практичного характеру

№ з/п	Назва структурного елементу	Примітки
1	Аналіз предметної області <назва предметної області>	9 – 12 с.
1.1	Коротка характеристика об’єкту управління <назва об’єкту управління>	2 – 3 с.
1.2	Опис предметної області <назва предметної області>	3 – 4 с.
1.3	Огляд і аналіз існуючих аналогів, що реалізують функції предметної області	4 – 5 с.
2	СПЕЦИФІКАЦІЯ ВИМОГ ДО МОДУЛЯ (СИСТЕМИ)	14 – 19 с.
2.1	Глосарій	2 с.
2.2	Розроблення варіантів використання	
2.2.1	Діаграма варіантів використання	2 – 3 с.
2.2.2	Специфікація варіантів використання	4 – 5 с.
2.2.3	Розкадровка варіантів використання	2 – 3 с.
2.3	Специфікація функціональних та не функціональних вимог	2 – 3 с.
3	Проектні та технічні рішення	19 – 25 с.
3.1	Математична постановка задачі	2 – 3 с.
3.2	Проектування структури бази даних	4 – 5 с.
3.3	Розроблення архітектури програмної системи	4 – 5 с.
3.4	Тестування додатку	5 – 7 с.
3.5	Розгортання програмного продукту	4 – 5 с.

Додаток Ж

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ

Допускається до захисту
завідувач кафедри ІС
д.е.н., проф. Пономаренко В.С. ____

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

"РОЗРОБЛЕННЯ МОДУЛЯ "КОНТРОЛЬ РОЗРАХУНКІВ ЗА ПОСЛУГИ ТЕПЛОПОСТАЧАННЯ" АВТОМАТИЗОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ ТОВ "БРОК-ЕНЕРГІЯ""

освітньо-кваліфікаційний рівень – бакалавр

Студент 4 курсу 5 гр. Турчак Р. С.

Керівник проекту
к.е.н., проф. кафедри ІС Золотарьова І. О.

Харків, 20__

Додаток З

Приклад рефератів українською та англійською мовами

1. **Для проектів дослідницького характеру.**

РЕФЕРАТ

Пояснювальна записка до дипломного проекту: 81 с., 34 рис., 6 табл., 2 додатка, 35 джерел.

Об’єктом дослідження є методи математичного моделювання інформаційних систем.

Метою роботи є дослідження підходів до розроблення імітаційних моделей процесів функціонування інформаційних систем та автоматизація процесу імітаційного моделювання.

Методами розробки обрано метод аналізу для дослідження існуючих методів моделювання, метод синтезу для поєднання переваг існуючих методів моделювання, методи моделювання для представлення та дослідження процесів функціонування інформаційних систем, метод порівняльного аналізу для оцінювання адекватності моделі процесів функціонування інформаційних систем.

У результаті виконання роботи обґрунтований раціональний підхід до розроблення імітаційних моделей процесів функціонування інформаційних систем та розроблений програмний засіб автоматизації імітаційного моделювання, який дозволяє створювати імітаційні моделі процесів функціонування інформаційних систем та досліджувати їх.

Результати дослідження можуть бути використані в науково-дослідницьких закладах та підрозділах підприємств, що займаються розробленням імітаційних моделей.

ІНФОРМАЦІЙНА СИСТЕМА, Е-МЕРЕЖА, АДЕКВАТНІСТЬ МОДЕЛІ, ІМІТАЦІЙНА МОДЕЛЬ, МАТЕМАТИЧНА СХЕМА, МЕТОДИ МОДЕЛЮВАННЯ.

ABSTRACT

The bachelor’s thesis report: 81 pages, 34 figures, 6 tables, 2 appendices, 35 sources.

The object of the research is methods of mathematical modeling of information systems.

The purpose of the work is to research approaches to development of simulation models of functioning processes of information systems and automation of simulation process.

Research methods are analysis to research existent methods of modeling, synthesis to unite advantages of the existent methods of modeling, modeling to present and research functioning processes of information systems, comparison to estimate the adequacy of the models.

As a result of the work the rational approach to developing the simulation models of the functioning processes of the information systems was substantiated. Also the software product, which allows creating simulation models of functioning processes of information systems and researching the models, was developed.

The results of the research can be used in research establishments and the departments of enterprises, which develop simulation models.

INFORMATION SYSTEM, E-NET, MODEL ADEQUACY, SIMULATION MODEL, MATHEMATICAL SCHEME, METHODS OF MODELING

2. Для проектів практичного характеру.

РЕФЕРАТ

Пояснювальна записка до дипломного проекту: 90 с., 30 рис., 20 табл., 12 додатків, 55 джерел.

Об’єктами проектування є функціональні елементів, архітектура, інформаційне і програмне забезпечення модуля моніторингу складських поставок ТОВ "СІГМА".

Мета проектування – створення модуля "Моніторинг складських поставок".

Метод проектування – використання програмних систем ARIS Toolset, IBM Rational, Microsoft Visual Studio.

Створений модуль дозволяє підвищити достовірність обліку товарів на складі, обґрунтованість та оперативність прийняття рішень при управлінні поставками товарів і їх запасами на складі.

Результати розробки можуть бути впроваджені на торгових підприємствах.

СИСТЕМА УПРАВЛІННЯ ЛАНЦЮГАМИ ПОСТАВОК, ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОЕКТУВАННЯ, CASE-ДІАГРАМИ, БАЗА ДАНИХ, МОНІТОРІНГ СКЛАДСЬКИХ ПОСТАВОК, WEB-СЛУЖБА

ABSTRACT

The bachelor’s thesis report: 90 pages, 30 figures, 20 tables, 12 appendices, 55 sources.

The objects of designing are functional elements, architecture and software of a module which provides monitoring of warehouse supplies.

The purpose of designing is to create "Monitoring of warehouse supplies" software module for Sigma Ltd.

Method of designing – using ARIS Toolset, IBM Rational, Microsoft Visual Studio software systems.

Advantage of the developed module is increasing reliability of accounting goods, validity and timeliness of decision-making.

The obtained results can be applied at commercial enterprises.

SUPPLY CHAIN MANAGEMENT SYSTEM, OBJECT ORIENTED DESIGN, CASE-DIAGRAMS, DATABASE, MONITORING OF WAREHOUSE SUPPLIES, WEB-SERVICE

Додаток И

Зразок оформлення змісту

Зміст

Додаток К

Зразок переліку умовних скорочень

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних;
КІС – корпоративна інформаційна система;
ПС – програмна система.

Додаток Л

Вимоги до оформлення списку використаних джерел

Книги

Один автор

1. Коренівський Д. Г. Дестабілізуючий ефект параметричного білого шуму в неперервних та дискретних динамічних системах / Д. Г. Коренівський. – К. : Ін-т математики, 2006. – 111 с. – (Математика та її застосування) (Праці / Ін-т математики НАН України ; т. 59).

2. Матюх Н. Д. Що дорожче срібла-золота / Н. Д. Матюх. – К. : Асамблея діл. кіл: Ін-т соц. іміджмейкінгу, 2006. – 311 с. – (Ювеліри України; т. 1).

Два автори

1. Суберляк О. В. Технологія переробки полімерних та композиційних матеріалів : підручник для студ. вищ. навч. закл. / О. В. Суберляк, П. І. Баштанник. – Львів : Растр-7, 2007. – 375 с.

Три автори

1. Акофф Р. Л. Идеализированное проектирование: как предотвратить завтрашний кризис сегодня. Создание будущего организации / Акофф Р. Л., Магидсон Д., Зддисон Г. Д. ; пер. с англ. Ф. П. Тарасенко. – Днепропетровск : Баланс Бизнес Букс, 2007. – XIЛI, 265 с.

Чотири автори

1. Механізація переробної галузі агропромислового комплексу : підручник для учнів проф.-техн. навч. закл. / О. В. Гвоздєв, Ф. Ю. Ялпачик, Ю. П. Рогач та ін. – К. : Вища освіта, 2006. – 478 с. – (ПТО: Професійно-технічна освіта).

П’ять і більше авторів

1. Психология менеджмента / Власов П. К., Липницкий А. В., Лущихина И. М. и др.; под ред. Г. С. Никифорова. – 3-е изд. – Х. : Гуманитарный центр, 2007. – 510 с.

Без автора

1. Проблеми типологічної та квантитативної лексикології : зб. наук. праць / наук. ред. Каліущенко В. та ін. – Чернівці : Рута, 2007. – 310 с.

Багатотомний документ

1. Межгосударственные стандарти : каталог в 6 т. / сост. Ковалева И. В., Рубцова Е. Ю. ; ред. Иванов В. Л. – Львов : НТЦ "Леонорм-Стандарт", 2005. – (Серия "Нормативная база предприятия"). Т. 1. – 2005. – 277 с.

2. Дарова А. Т. Неисповедимы пути Господни...: (Дочь врага народа): трилогия / А. Дарова. – Одесса : Астропринт, 2006. – (Сочинения : в 8 кн. / А. Дарова; кн.4).

3. Бондаренко В. Г. Теорія ймовірностей і математична статистика. Ч.1 / В. Г. Бондаренко, І. Ю. Канівська, С. М. Парамонова. – К. : НТУУ "КПГ, 2006. – 125 с.

Матеріали конференцій, з'їздів

1. Кібернетика в сучасних економічних процесах : зб. текстів виступів на республік, міжвуз. наук.-практ. конф. / Держкомстат України, Ін-т статистики, обліку та аудиту. – К. : ICOA, 2002. – 147 с.

2. Матеріали IX з'їзду Асоціації українських банків, 30 червня 2000 р. інформ. бюл. – К. : Асоц. укр. банків, 2000. – 117 с.

3. Оцінка й обґрунтування продовження ресурсу елементів конструкцій : праці конф., 6 – 9 черв. 2000 р., Київ. Т. 2 / відп. ред. В. Т. Трощенко. – К. : НАН України, Ін-т пробл. міцності, 2000. – С. 559–956 – (Ресурс 2000).

Препринти

1. Шиляев Б. А. Расчеты параметров радиационного повреждения материалов нейтронами источника ННЦ ХФТИ/ANL USA с подкритической сборкой, управляемой ускорителем злектронов / Шиляев Б. А., Воєводин В. Н. – Х. : ННЦ ХФТИ, 2006. – 19 с . – (Препринт / НАН Украины, Нац. науч. центр "Харьк. физ.-техн. ин-т" ; ХФТИ 2006-4).

Депоновані наукові праці

1. Социологическое исследование малых групп населення / В. И. Иванов и др. ; М-во образования Рос. Федерации, Финансовая академия. – М., 2002. – 110 с. – Деп. в ВИНИТИ 13.06.02, № 145432.

2. Разумовский В. А. Управление маркетинговыми исследованиями в регионе / В. А. Разумовский, Д. А. Андреев. – М., 2002. – 210 с. – Деп. в ИНИОН Рос. акад. наук 15.02.02, № 139876.

Словники

1. Українсько-німецький тематичний словник // уклад. Н. Яцко та ін. – К. : Карпенко, 2007. – 219 с.

2. Європейський Союз : словник-довідник / ред.-упоряд. М. Марченко. – 2-ге вид., оновл. – К. : К.І.С., 2006. – 138 с.

Законодавчі та нормативні документи

1. Кримінально-процесуальний кодекс України : за станом на 1 груд. 2005 р. / Верховна Рада України. – Офіц. вид. – К. : Парлам. вид., 2006. – 207 с. – (Бібліотека офіційних видань).

2. Медична статистика статистика : зб. нормат. док. / упоряд. та голов. ред. В. М. Заболотько. – К. : МНІАЦ мед. статистики : Медінформ, 2006. – 459 с. – (Нормативні директивні правові документи).

3. Експлуатація, порядок і терміни перевірки запобіжних пристроїв посудин, апаратів і трубопроводів теплових електростанцій : СОУ-Н ЕЕ 39.501:2007. – Офіц. вид. – К. : ГРІФРЕ : М-во палива та енергетики України, 2007. – VI, 74 с. – (Нормативний документ Мінпаливенерго України. Інструкція).

Стандарти

1. Графічні символи, що їх використовують на устаткуванні. Покажчик та огляд (ІЅО 7000:2004, ЮТ) : ДСТУ ІЗО 7000:2004. – Чинний від 2006-01-01. – К. : Держспоживстандарт України, 2006. – IV, 231 с. – Національний стандарт України).

2. Якість води. Словник термінів : ДСТУ ІЗО 6107-1:2004 – ДСТУ ІЗО 6107-9:2004. – Чинний від 2005-04-01. – К. : Держспоживстандарт України, 2006. – 181 с. – (Національні стандарти України).

Каталоги

1. Межгосударственные стандарти : каталог : в 6 т. / сост. Ковалева И. В., Павлюкова В. А. ; ред. Иванов В. Л. – Львов : НТЦ "Леонорм-стандарт", 2006. – (Серия "Нормативная база предприятия"). Т. 5. – 2007. – 264 с.

Т. 6. – 2007. – 277 с.

2. Горницкая И. П. Каталог растений для работ по фитодизайну / Горницкая И. П., Ткачук Л. П. – Донецк : Лебедь, 2005. – 228 с.

Бібліографічні показчики

1. Куц О. С. Бібліографічний покажчик та анотації кандидатських дисертацій, захищених у спеціалізованій вченій раді Львівського державного університету фізичної культури у 2006 році / О. Куц, О. Вацеба. – Львів : Укр. технології, 2007. – 74 с.

Дисертації

1. Петров П. П. Активність молодих зірок сонячної маси: дис. ... доктора фіз.мат, наук : 01.03.02 / Петров Петро Петрович. – К., 2005. – 276 с.

Автореферати дисертацій

1. Новосад І. Я. Технологічне забезпечення виготовлення секцій робочих органів гнучких гвинтових конвеєрів : автореф. дис. на здобуття наук, ступеня канд. техн. наук : спец. 05.02.08 "Технологія машинобудування" / І. Я. Новосад. – Тернопіль, 2007. – 20 с.

2. Нгуен Ші Данг. Моделювання і прогнозування макроекономічних показників в системі підтримки прийняття рішень управління державними фінансами : автореф. дис. на здобуття наук, ступеня канд. техн. наук : спец. 05.13.06 "Автоматиз. системи упр. та прогрес, інформ. технології" / Нгуен Ші Данг. – К., 2007. – 20 с.

Авторські свідоцтва

1. А. с. 1007970 СССР, МКИЗ В 25 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.

Патенти

1. Пат. 2187888 Российская Федерация, МПК Н 04 В 1/38, Н 04 и 13/00. Приемопередающее устройство / Чугаева 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.).

Частина книги, періодичного продовжуваного видання

1. Козіна Ж. Л. Теоретичні основи і результати практичного застосування системного аналізу в наукових дослідженнях в області спортивних ігор / Ж. Л. Козіна // Теорія та методика фізичного виховання. – 2007. – № 6. – С. 15–18, 35–38.

2. Гранчак Т. Інформаційно-аналітичні структури бібліотек в умовах демократичних перетворень / Тетяна Гранчак, Валерій Горовий // Бібліотечний вісник. – 2006. – № 6. – С. 14–17.

3. Валькман Ю. Р. Моделирование НЕ-факторов – основа интеллектуализации компьютерных технологий / Ю. Р. Валькман, В. С. Быков, А. Ю. Рыхальский // Системні дослідження та інформаційні технології. – 2007. – № 1 – С. 39–61.

Електронні ресурси

1. Бібліотека і доступність інформації у сучасному світі: електронні ресурси в науці, культурі та освіті : (підсумки 10-ї Міжнар. конф. "Крим-2003") [Електронний ресурс] / Л. Й. Костенко, А. О. Чекмарьов, А. Г. Бровкін, І. А. Павлуша // Бібліотечний вісник – 2003. – № 4. – С. 43. – Режим доступу : http://www.nbuv.ua/articles/2003/03klinko.htm.

2. Богомольний Б. Р. Медицина екстремальних ситуацій [Електронний ресурс] : навч. посібн. для студ. мед. вузів III – IV рівнів акредитації / Б. Р. Богомольний, В. В. Кононенко, П. М. Чуєв. – 80 Min / 700 MB. – Одеса : Одес. мед. ун-т, 2003. – (Бібліотека студента-медика) – 1 електрон, опт. диск (CD-ROM) ; 12 см. – Систем. вимоги: Pentium ; 32 Mb RAM ; Windows 95, 98, 2000, XP ; MS Word 97-2000. – Назва з контейнера.

Приклад діаграм опису предметної області

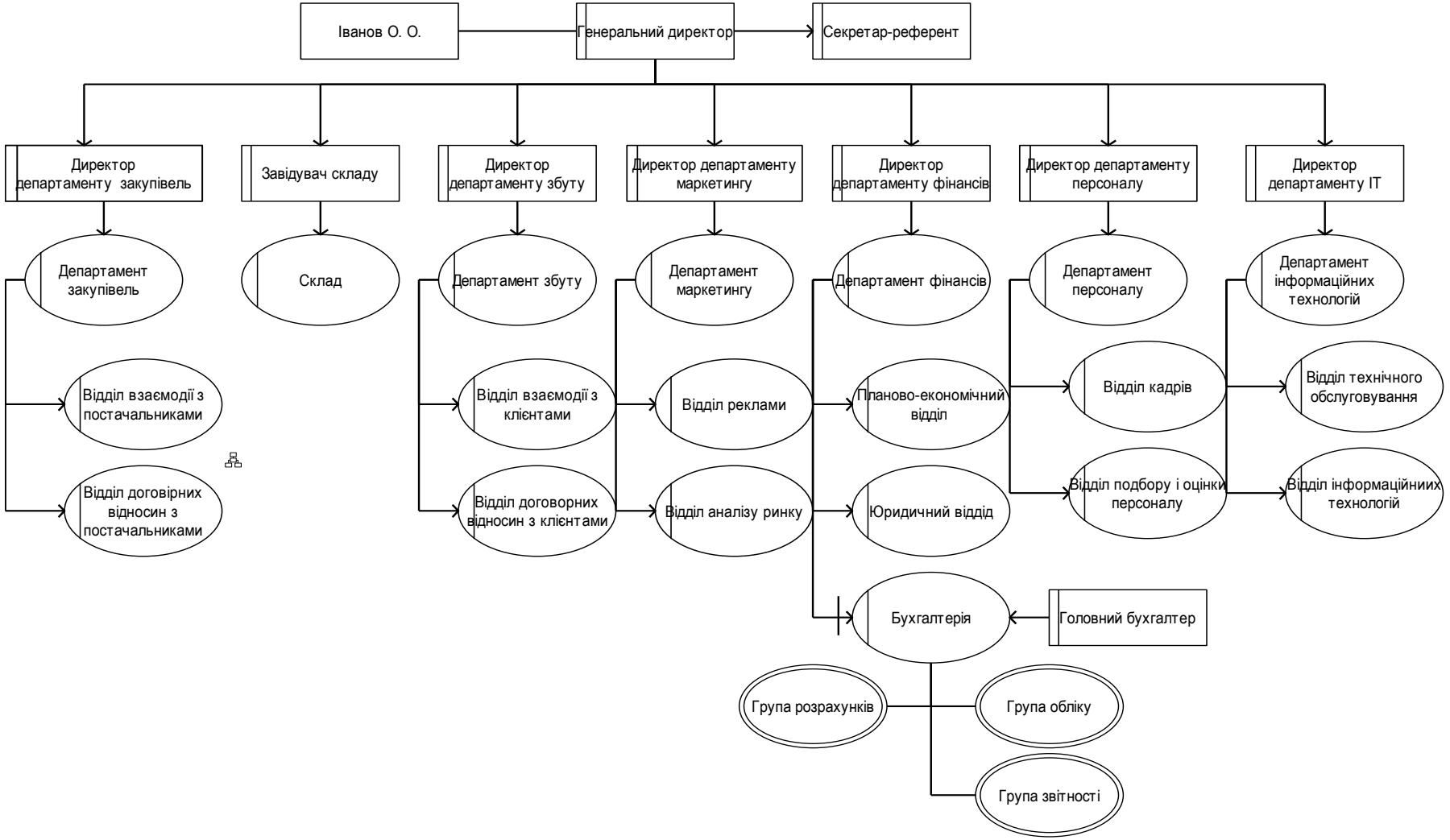

Рис. М.1. **Організаційна структура підприємства дрібнооптової торгівлі**

Приклад діаграм опису предметної області

Рис. М.2. **Організаційна схема відділу по взаємодії з постачальниками**

Приклад діаграм опису предметної області

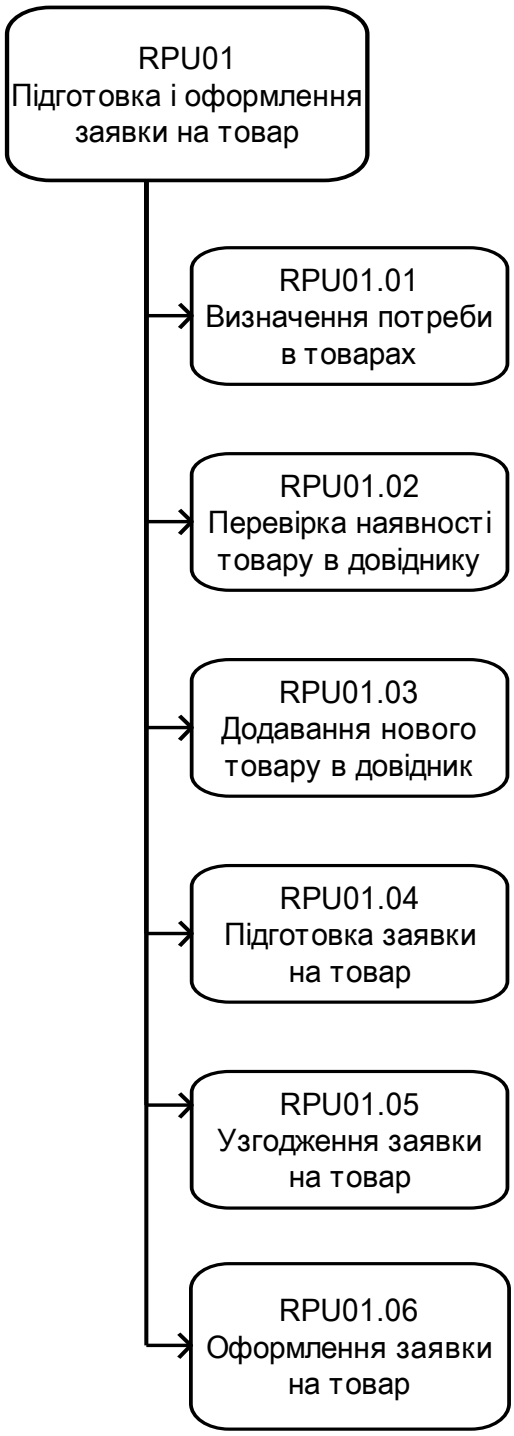

Рис. М.3. **Діаграма "Дерево функцій"**

Приклад діаграм опису предметної області

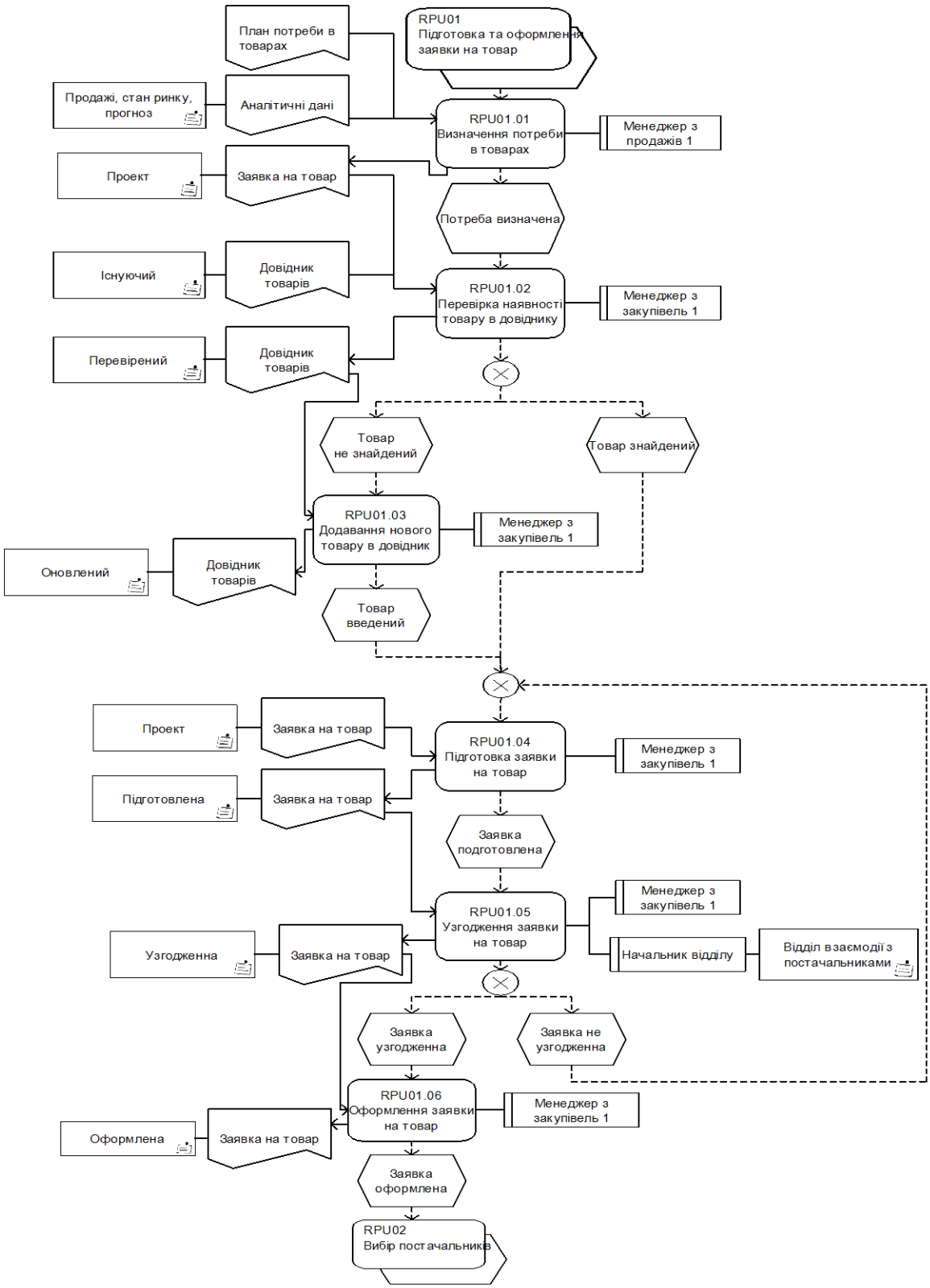

Рис. М.4. **Процес "PR01. Підготовка та оформлення заявки на товар"**

Приклад структурної схеми програмної системи

Рис. Н.1. **Структурна схема розробленої програмної системи**

Додаток П

Приклад тест-плану, сполученого зі звітом про проведення тестування

Група тестів: Робота з обліковими записами

Тестовий приклад: № 1

Призначення: перевірка того, що перед початком передачі даних перевіряється обліковий запис користувача й у випадку введення імені користувача за замовчуванням при знаходженні системи в стані "Максимальний захист" передача даних не відбувається.

Тест-вимоги, що перевіряються: 1.1, 1.2

Передумови для тесту: система повинна бути переведена в стан "Максимальний захист", а її настройки повинні мати значення за замовчуванням.

Критерій проходження тесту: всі реальні значення збігаються з очікуваними.

Таблиця П.1

№ з/п	Крок сценарію	Очікуваний результат	Отриманий результат	Відмітка про проходження кроку сценарію (Так/Ні)
1	2	3	4	5
1.	Запустити термінальний клієнт і ввести IP-адресу 127.0.0.1	Повинне з’явитися запрошення: **TRANSFER>**	Запрошення **TRANSFER>** з’являється	Так
2.	Запустити процес передачі даних за допомогою введення команди **SEND DATA**	Повинне з’явитися запрошення: **Enter your credentials... Login:**	Запрошення **Enter your credentials... Login:** з’являється	Так

Закінчення табл. П.1

1	2	3	4	5
3.	Увести ім’я користувача **default**	Повинне з’явитися повідомлення: **Password:**	Повідомлення **Password:** з’являється	Так
4.	Увести пароль **default**	Повинне з’явитися повідомлення: **Default user blocked - system set to High security.**	З’являється повідомлення: **Your credetentials accepted. Data transfer started.**	Ні

Відмітка про проходження тесту (пройдений/не пройдений): не пройдений

Тестовий приклад: № 2

……

Тестовий приклад: № 3

……

Тестових прикладів виконано: 3

Тестових прикладів пройдено: 1

Зміст

НАВЧАЛЬНЕ ВИДАННЯ

Методичні рекомендації до виконання дипломного проекту освітньо-кваліфікаційного рівня "бакалавр"

для студентів напряму підготовки 6.050101 "Комп'ютерні науки" всіх форм навчання

Укладачі: **Золотарьова** Ірина Олександрівна
Парфьонов Юрій Едуардович
Ушакова Ірина Олексіївна

Відповідальний за випуск **Пономаренко В. С.**

Редактор **Пушкар І. П.**

Коректор **Бриль В. О.**

План 2012 р. Поз. № 283.
Підп. до друку Формат 60х90 1/16. Папір MultiCopy. Друк Riso.
Ум.-друк. арк. 5,0. Обл.-вид. арк. 6,25. Тираж прим. Зам. №
Видавець і виготівник – видавництво ХНЕУ, 61166, м. Харків, пр. Леніна, 9а
Свідоцтво про внесення до Державного реєстру суб'єктів видавничої справи ***Дк № 481 від 13.06.2001 р.***

Методичні рекомендації
до виконання дипломного проекту
освітньо-кваліфікаційного рівня "бакалавр"
для студентів напряму підготовки 6.050101
"Комп'ютерні науки"
всіх форм навчання