Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія розрахунку вимог на використання будівельних матеріалів»

Виконав: студент 2-го курсу, групи ЗКН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
Сіваєв.А.С.
(прізвище та ініціали)

Керівник: к. т. н., ст. в. каф. КН
Петришин С.І.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент: к.т.н., доцент каф. АІІТ
Барабан М.В.
(прізвище та ініціали)
«07» 12 2023 р.

Допущено до захисту
Завідувач кафедри КН
д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«08» 12 2023 р.

Вінниця ВНТУ - 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
Д.т.н., проф. Яровий А.А.
29. 08 2023 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Сіваєву Андрію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційна технологія розрахунку вимог на використання будівельних матеріалів

керівник роботи к. т. н., ст. в. каф. КН Петришин С.І.
затверджені наказом вищого навчального закладу від "12" 09 2023 року №297
2. Строк подання студентом роботи 13. 11 2023 року

3. Вихідні дані до роботи:
Вхідні дані не менше 3 базових типів фігур, не менше 10 кольорів зображення, підтримка операційної системи Windows, табличне і графічне подання.

4. Зміст текстової частини:
Вступ, аналіз предметної області САПР, проектування модуля розрахунку вимог на використання будівельних матеріалів, програмна реалізація модуля розрахунку вимог на використання будівельних матеріалів, висновки, список використаних джерел, додатки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)
Загальна структура САПР, UML-діаграми модуля розрахунку вимог на використання будівельних матеріалів, скріншоти роботи програми.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	виконав прийняв
1-3	Петришин С.І. , к. т. н., ст. в. каф. КН	29.08.23	10.11.23
4	Кавецький В.В., к.е.н., доц. каф. ЕПВМ	16.10.23	20.10.23

7. Дата видачі завдання 29 . 08 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи
1	Аналіз предметної області САПР	01.09.23 - 07.09.23
2	Проектування роботи програмного модуля інформаційної технології розрахунку вимог на використання будівельних матеріалів	08.09.23 - 24.09.23
3	Реалізація програмного забезпечення програмного модуля інформаційної технології розрахунку вимог на використання будівельних матеріалів	25.09.23 - 15.10.23
4	Підготовка економічної частини	16.10.23 - 20.10.23
5	Апробація та/або впровадження результатів дослідження	21.10.23 - 03.11.23
6	Оформлення пояснювальної записки, графічного матеріалу та презентації	06.11.23 - 10.11.23

Студент _____ (підпис) Сіваєв А.С.

Керівник роботи _____ (підпис) Петришин С.І.

АНОТАЦІЯ

УДК 004.82

Сіваєв А.С. Інформаційна технологія розрахунку вимог на використання будівельних матеріалів. Магістерська кваліфікаційна робота зі спеціальності 122 «Комп`ютерні науки», освітня програма «Системи штучного інтелекту». Вінниця: ВНТУ, 2023. 130 с.

На укр. мові. Бібліогр.: 22 назв; рис.: 10; табл. 15.

Дослідження присвячене використанню інформаційних технологій для розрахунку вимог на використання будівельних матеріалів. У роботі розглядаються переваги використання інформаційних технологій у будівельній галузі, зокрема їх вплив на точність, швидкість та надійність проектування і будівництва. Виявлено, що застосування програмних засобів на основі інформаційних технологій дозволяє здійснювати комплексний аналіз властивостей матеріалів та їх взаємодії з конструкціями. Дослідження підтверджує, що використання інформаційних технологій у розрахунку вимог на будівельні матеріали сприяє зменшенню витрат на будівництво і підвищенню якості та тривалості експлуатації конструкцій.

Ключові слова: будівництво, інформаційні технології, конструкції.

ABSTRACT

Sivaev A.S. Information technology for calculating requirements for the use of building materials. Master's qualification thesis on specialty 122 "Computer science", educational program "Artificial intelligence systems". Vinnytsia: VNTU, 2023. 130 p.

In Ukrainian speech Bibliography: 22 titles; Fig.: 10; table 15.

The research is dedicated to the utilization of information technology for calculating requirements for the use of construction materials. The study examines the advantages of employing information technology in the construction industry, particularly its impact on the accuracy, speed, and reliability of design and construction processes. It has been found that the implementation of software tools based on information technology enables a comprehensive analysis of material properties and their interaction with structures. The research confirms that the application of information technology in calculating requirements for construction materials contributes to cost reduction in construction and enhances the quality and durability of structures.

Keywords: construction, information technologies, structures.

ЗМІСТ

ВСТУП

Актуальність. Сучасні технологічні досягнення в галузі інформаційних технологій надали нам унікальну можливість вирішувати складні завдання в оптимізації та вдосконаленні вибору будівельних матеріалів. Застосування інформаційних систем і програмних засобів для аналізу та вибору матеріалів може вирішити проблему оптимізації витрат та забезпечити високу якість побудов.

Додатково, робота розглядає проблематику відповідності будівельних матеріалів нормативам і стандартам, що є надзвичайно критичним аспектом сучасного будівництва. Забезпечення відповідності нормам дозволяє забезпечити безпеку, надійність та довговічність будівельних об'єктів.

Окрім того, актуальність даної роботи виявляється в контексті зростаючого обурення екологічними аспектами у будівництві. Розробка інформаційної технології, що дозволяє обирати матеріали з найменшим негативним впливом на навколишнє середовище, є практично невідкладною задачею для галузі в цілому.

Нарешті, інноваційний характер запропонованої технології дозволяє підвищити конкурентоспроможність підприємств у будівельній сфері, створюючи умови для впровадження передових підходів та технологій в практику.

Загальна мета даної роботи полягає в розробці та впровадженні інформаційної технології, яка сприятиме підвищенню зручності розробки будівельних проектів, а також дозволить зменшити негативний вплив будівництва на навколишнє середовище. Результати цього дослідження можуть мати значущий практичний вплив на розвиток будівельної галузі та суспільства в цілому.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська кваліфікаційна робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного

університету 22 К1 «Розробка прикладних інтелектуальних інформаційних технологій та систем» та плану наукової та навчально-методичної роботи кафедри.

Мета та завдання дослідження. Метою дослідження магістерської є розширення функціональних можливостей програмного забезпечення розрахунку витрат на будівельні матеріали, що дозволить автоматизувати проектні роботи на будівництві.

Для досягнення поставленої мети необхідно виконати такі завдання:

– здійснити аналіз предметної області додатків для розрахунку будівельних матеріалів та реалізацій САПР;

– розробити алгоритм роботи модуля розрахунку вимог на використання будівельних матеріалів;

– здійснити проектування структури модуля розрахунку вимог на використання будівельних матеріалів;

– розробити модуль розрахунку вимог на використання будівельних матеріалів;

– провести тестування модуля розрахунку вимог на використання будівельних матеріалів;

– виконати економічні розрахунки.

Об’єкт дослідження є процес розрахунку вимог на використання будівельних матеріалів з використанням інформаційної технології.

Предмет дослідження є програмні засоби розрахунку вимог на використання будівельних матеріалів.

Методи дослідження. У роботі використано такі методи наукових досліджень: методи аналізу даних; методи машинного навчання; методи об’єктно-орієнтованого програмування.

Наукова новизна. Запропоновано інформаційну технологію розрахунку вимог на використання будівельних матеріалів, яка відрізняється від існуючих удосконаленою інтелектуальною моделю розрахунку кошторису будівельних

робіт, що дозволило підвищити якість розрахунку вимог на використання будівельних матеріалів.

Практичне значення одержаних результатів полягає в тому, що на основі проведених досліджень розроблено програмне забезпечення зокрема:

1. Розроблено алгоритм роботи програмного забезпечення для розрахунку вимог на використання будівельних матеріалів для інформаційної технології;

2. Розроблено програмне забезпечення для інформаційної технології розрахунку вимог на використання будівельних матеріалів, що забезпечує підвищення якості розрахунку вимог і з врахуванням різних аспектів.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю роботи програмних модулів, тестуванням програмної реалізації інформаційної технології розрахунку вимог на використання будівельних матеріалів. Адекватність розробленої технології підтверджується результатами тестування та експериментальних досліджень.

Особистий внесок магістранта. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно. У роботах, опублікованих у співавторстві, автору належать такі результати: [1] – розробка інтелектуального модуля для інформаційної технології розрахунку вимог на використання будівельних матеріалів.

Апробація. Результати дослідження були представлені та обговорені на LII науково-технічній конференції підрозділів ВНТУ (2023) [1].

Публікації. За результатами магістерської кваліфікаційної роботи опубліковано дослідження опубліковано: 1 тези доповідей конференцій [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ САПР

1.1 Аналіз предметної області додатків для розрахунку будівельних матеріалів

В даному підрозділі оглянемо відомі додатки та інструменти для розрахунку будівельних матеріалів.

AutoCAD Civil 3D – це інтегроване програмне забезпечення для проектування та документування інфраструктурних об'єктів. Включає модулі для розрахунку обсягів земельних робіт та вибору будівельних матеріалів (рис.1.1).

AutoCAD Civil 3D є впливовим програмним продуктом, призначеним для професійного проектування інфраструктурних об'єктів у галузі будівництва та цивільного інжинірингу. Це інтегроване рішення, яке об'єднує в собі комплекс інструментів для точного та ефективного проектування[2].

Основні характеристики та функції:

– Проектування інфраструктурних об'єктів. AutoCAD Civil 3D надає можливість проектувати дороги, мости, каналізаційні системи, водозабори та інші об'єкти інфраструктури.

– Модуль розрахунку обсягів земельних робіт. Система вміє автоматично розраховувати обсяги викопу та нанесення грунту на будівельному майданчику. Це важливий етап у плануванні та оцінці вартості будівельного проекту.

– Вибір будівельних матеріалів. Пакет дозволяє враховувати та вибирати оптимальні будівельні матеріали для конкретних елементів проекту з урахуванням вимог до міцності, економічності та екологічних аспектів.

– 3D-моделювання. AutoCAD Civil 3D дозволяє створювати тривимірні моделі, що сприяє кращому розумінню та візуалізації проекту.

– Автоматизація робочих процесів. Інтеграція з іншими продуктами Autodesk дозволяє автоматизувати багато рутинних завдань, що сприяє підвищенню продуктивності.

– Реалістична візуалізація. Пакет має засоби для створення реалістичних зображень та відеороликів проектів для відображення їх потенційним замовникам чи іншим зацікавленим сторонам.

Загалом, AutoCAD Civil 3D є важливим інструментом для фахівців у галузі будівництва, проектування та цивільного інжинірингу. Він дозволяє оптимізувати та поліпшити якість проектування та реалізації інфраструктурних об'єктів.

АвтоCAD Civil 3D дійсно відіграє ключову роль у сучасній інженерній сфері, сприяючи удосконаленню процесів проектування та будівництва інфраструктурних об'єктів. Цей програмний засіб надає можливості для створення детальних та точних проектів, враховуючи різноманітні аспекти, такі як геодезія, дорожнє будівництво, системи водопостачання та водовідведення, транспортні мережі та багато інших.

Однією з ключових переваг є можливість роботи з інформацією проекту у вигляді 3D моделей. Це дозволяє здійснювати більш ефективний контроль над проектом, виявляти можливі конфлікти та проблеми ще до початку фізичної реалізації. Такий підхід дозволяє уникнути помилок, зменшити ризики та збільшити швидкість виконання проекту.

Також варто зазначити, що AutoCAD Civil 3D має інтегровані засоби для автоматизації багатьох завдань, що раніше вимагали значного часу і зусиль. Наприклад, автоматизоване проектування доріг з урахуванням топографічних особливостей дозволяє значно прискорити процес та знизити ймовірність помилок.

У цілому, AutoCAD Civil 3D відіграє критичну роль у покращенні ефективності та точності проектування та будівництва інфраструктурних об'єктів, роблячи його надійним інструментом для фахівців у галузі цивільного будівництва та інжинірингу.

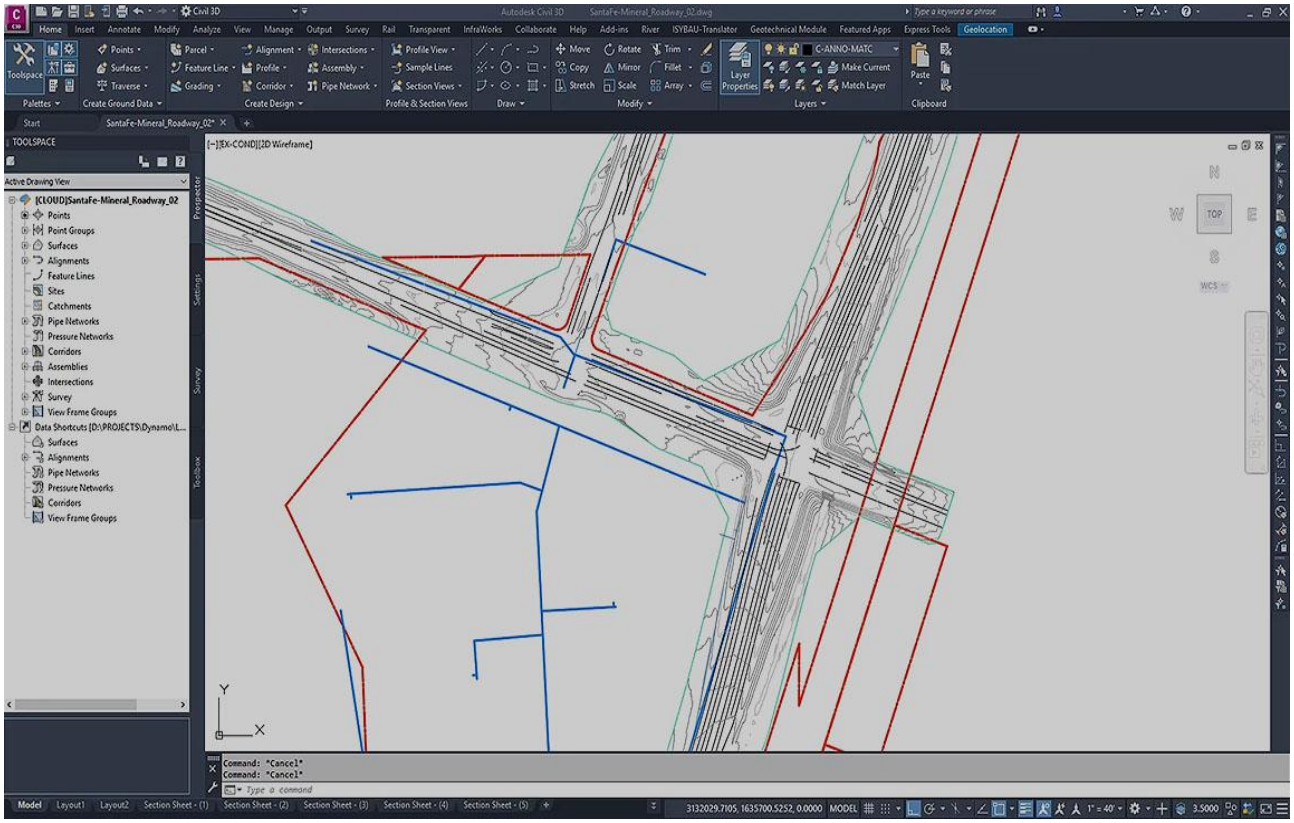

Рисунок 1.1 – Програмний інтерфейс AutoCAD Civil 3D

SketchUp – простий у використанні 3D-моделювальник, який дозволяє створювати та модифікувати моделі будівель та їх елементів. Має плагіни для розрахунку обсягів матеріалів (рис.1.2).

SketchUp – це потужний і водночас простий у використанні 3D-моделювальник, який надає можливість швидко створювати та модифікувати моделі будівель та їх окремих елементів. Його інтуїтивний інтерфейс дозволяє користувачам з різним рівнем навичок швидко освоювати основи моделювання.

Основні характеристики та можливості:

– Простота використання. SketchUp відомий своєю інтуїтивно зрозумілою і простою у використанні платформою. Навіть початківці можуть швидко навчитися його основам.

– 3D-моделювання будівель та елементів. Програма дозволяє створювати деталізовані тривимірні моделі будівель та окремих їх частин. Це важливий аспект для візуалізації та розуміння конструкційних рішень.

– Модифікація та редагування. З SketchUp можна легко змінювати розміри, форму, поверхні та інші параметри будівельних об'єктів, що дозволяє швидко адаптувати модель під конкретні вимоги.

– Плагіни для розрахунку обсягів матеріалів. Важливою функцією для будівельної сфери є можливість розраховувати обсяги необхідних матеріалів дляпроекту. У SketchUp можна використовувати різноманітні плагіни для вирішення цієї задачі.

– Візуалізація та анімація. Програма дозволяє створювати реалістичні візуалізації будівельних об'єктів та навіть створювати анімації, щоб продемонструвати рух елементів чи зміни в проекті.

– Додаткові ресурси та спільнота. В SketchUp існує широка глобальна спільнота користувачів, а також велика кількість додаткових ресурсів, таких як моделі, текстури та плагіни, що полегшують процес роботи.

Загалом, SketchUp є надзвичайно корисним інструментом для архітекторів, дизайнерів та будівельників, які потребують швидкого та ефективного способу моделювання будівельних об'єктів. На додачу до його потужних можливостей у створенні 3D-моделей, він також дозволяє ефективно керувати обсягами матеріалів для проектів.

SketchUp дійсно виявляється надзвичайно корисним для архітекторів, дизайнерів та будівельників, оскільки це потужний інструмент для швидкого та ефективного моделювання будівельних об'єктів. Однією з його головних переваг є інтуїтивний і простий у використанні інтерфейс, що дозволяє швидко перетворювати концепції в 3D-моделі без великих зусиль.

Крім цього, SketchUp дозволяє не лише створювати моделі, але й ефективно керувати обсягами матеріалів для проектів. Це важливо для більш точного оцінювання витрат на будівництво та планування ресурсів. Він надає можливість швидко експериментувати з різними матеріалами, колірними схемами та текстурами, що дозволяє архітекторам та дизайнерам візуалізувати свої ідеї та концепції з великою точністю.

Ключовою особливістю SketchUp є його здатність до інтеграції з іншими програмами, що спрощує обмін даними та співпрацю між різними фахівцями під час проектування та будівництва.

Усього враховуючи, SketchUp став невід'ємною частиною робочого процесу для багатьох фахівців у сфері архітектури та будівництва, завдяки своїм швидким можливостям моделювання та зручності в управлінні матеріалами для проектів.

Рисунок 1.2 – Програмний інтерфейс SketchUp

Autodesk Revit – комплексне програмне забезпечення для проектування будівель та споруд. Має модулі для розрахунку будівельних матеріалів та їх кількостей (рис.1.3).

Autodesk Revit – це комплексне програмне забезпечення для інформаційного моделювання будівель (BIM), яке спрямоване на проектування, будівництво та управління будівельними об'єктами. Ця програма дозволяє фахівцям у галузі будівництва та архітектури створювати повноцінні 3D-моделі будівель і інфраструктури з врахуванням всіх аспектів проекту.

Основні характеристики та можливості:

– Інформаційне моделювання будівель (BIM). Revit використовує концепцію BIM для створення докладної та комплексної інформаційної моделі будівлі. Це дозволяє зберігати інформацію про всі аспекти будівництва, включаючи геометрію, конструкційні рішення, матеріали, технічні характеристики та багато іншого.

– Модулі для розрахунку обсягів та кількостей матеріалів. Revit надає інструменти для автоматизованого розрахунку кількостей будівельних матеріалів, таких як цегла, бетон, сталь, ізоляція та інші. Це допомагає оцінювати вартість будівництва та управляти запасами матеріалів.

– Інтеграція з іншими продуктами Autodesk. Revit легко інтегрується з іншими програмами Autodesk, такими як AutoCAD, для спільної роботи та обміну даними між різними програмами.

– Спільна робота над проектом. Завдяки інтернет-платформі Autodesk BIM 360, команди можуть спільно працювати над проектом, обмінюватися даними та відстежувати зміни в реальному часі.

– Аналіз та симуляція. Revit дозволяє проводити різноманітні аналізи та симуляції, такі як аналіз енергоефективності, вентиляції та освітлення, що допомагає покращити ефективність та якість будівельних проектів.

– Документація та керування проектом. Revit дозволяє генерувати автоматизовану документацію для будівельного проекту, включаючи плани, розрізи, специфікації та інші документи, необхідні для будівництва та обслуговування об'єкта.

Revit є однією з провідних програм у галузі BIM та є надзвичайно корисним інструментом для всіх сторін будівельного процесу - від архітекторів та інженерів до підрядників та замовників. Він спрощує та покращує всі етапи проектування та будівництва, забезпечуючи більшу точність, ефективність та зменшення ризиків.

Revit дійсно визнаний як одна з провідних програм у світі BIM (Building Information Modeling) і виявляється надзвичайно корисним інструментом для всіх учасників будівельного процесу. Від архітекторів та інженерів до

підрядників та замовників - Revit спрощує та оптимізує всі етапи проектування та будівництва.

Однією з ключових переваг Revit є його здатність створювати цифрові моделі будівель, які містять повну інформацію про всі елементи проекту. Це дозволяє різним учасникам проекту працювати у спільному середовищі, обмінюватися даними та співпрацювати в реальному часі, що спрощує координацію та підвищує точність.

Також Revit надає можливість проводити аналізи, включаючи енергетичні, структурні та конструктивні, що дозволяє заздалегідь виявити потенційні проблеми та оптимізувати проект з метою покращення ефективності та зниження ризиків.

Завдяки своїм можливостям у покращенні співпраці, точності та ефективності проектування та будівництва, Revit став невід'ємною частиною робочого процесу у будівельній галузі, сприяючи підвищенню якості проектів та забезпечуючи оптимальне використання ресурсів.

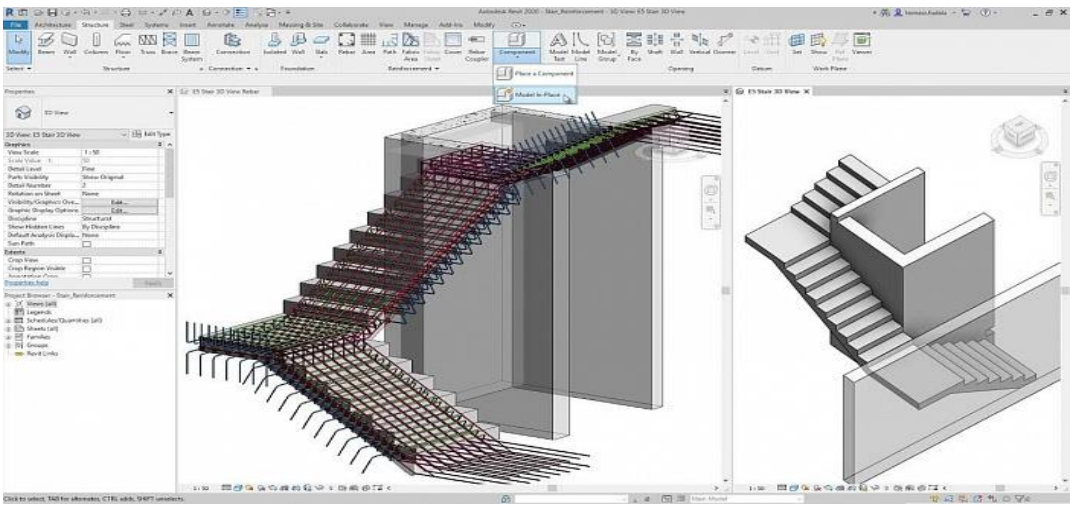

Рисунок 1.3 – Програмний інтерфейс Autodesk Revit

Необхідно порівняти функціонал та можливості трьох розглянутих систем моделювання будівельних матеріалів. Результати наведені в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика відомих додатків розрахунку будівельних матеріалів

Назва	**AutoCAD Civil 3D**	**SketchUp**	**Autodesk Revit**
Опис	Інтегроване програмне забезпечення для проектування та документування інфраструктурних об'єктів. Включає модулі для розрахунку обсягів земельних робіт та вибору будівельних матеріалів.	Простий у використанні 3D-моделювальник, дозволяє створювати та модифікувати моделі будівель та їх елементів. Має плагіни для розрахунку обсягів матеріалів.	Комплексне програмне забезпечення для проектування будівель та споруд. Має модулі для розрахунку будівельних матеріалів та їх кількостей.
Складність в освоєнні	Висока	Середня	Висока
Цінова політика	Висока	Від безкоштовної версії до платних розширень	Висока

Продовження таблиці 1.1

Професійна спрямованість	Інженерія, будівництво, інфраструктура	Архітектура, дизайн, будівництво	Архітектура, інженерія, будівництво
Функціональні можливості	- Розрахунок земельних робіт - Вибір будівельних матеріалів - Інтеграція з AutoCAD	- 3D-моделювання - Модифікація та редагування - Плагіни для розрахунку обсягів матеріалів	- Інформаційне моделювання будівель (BIM) - Модулі для розрахунку обсягів та кількостей матеріалів - Аналіз та симуляція - Інтеграція з іншими продуктами Autodesk
Візуалізація та анімація	Низька	Середня	Середня
Спільна робота	Часткова	Залежить від версії	Так
Інтеграція з іншими продуктами	Інтеграція з іншими програмами AutoCAD	Менше можливостей	Інтеграція з іншими продуктами Autodesk
Оптимізація під роботу на обмежених ресурсах	Висока	Висока	Низька

1.2 Аналітичний огляд відомих реалізацій САПР

Системи автоматизованого проектування (САПР) - це важливий компонент у багатьох індустріях, зокрема в інженерії, архітектурі та виробництві. Вони надають засоби для створення, аналізу та оптимізації проектів у цифровій формі, що дозволяє значно поліпшити ефективність роботи, знизити помилки та прискорити процеси.

Унікальність САПР полягає у їхній здатності створювати віртуальні моделі об'єктів, будівель, систем або складних механізмів. Ці моделі відображають не лише геометричну форму, але й інформацію про властивості матеріалів, параметри роботи систем, характеристики елементів і багато іншої важливої інформації [3].

Це дозволяє фахівцям візуалізувати проекти з точністю до деталей, вносити зміни в реальному часі, виявляти можливі проблеми або конфлікти та забезпечувати більш ефективну співпрацю між різними відділами та фахівцями.

Крім того, САПР надають інструменти для розрахунків, аналізу та оптимізації, що дозволяє забезпечити високу якість та ефективність проектування в різних галузях. Вони є невід'ємною частиною сучасних технологій та виробничих процесів, допомагаючи спрощувати складні завдання та забезпечувати стабільну якість продукції чи проекту.

САПР є невід'ємною частиною сучасної індустрії та будівельної сфери. Вони забезпечують можливість створення якісних та ефективних проектів, сприяючи тим самим підвищенню продуктивності та ефективності роботи в цих галузях. Що стосується конкретних програм, то на ринку існують кілька відомих реалізацій САПР, які відрізняються за своїми можливостями та спрямованістю [4].

Системи автоматизованого проектування та проектно-конструкторського моделювання (САПР) є важливим інструментом для інженерів, архітекторів і фахівців у галузях будівництва та промисловості. Вони надають змогу автоматизувати процеси розробки, виробництва і управління технічними

проектами. Це включає в себе створення детальних 2D та 3D моделей, аналіз параметрів, а також оптимізацію та координацію проектних рішень.

САПР відіграють ключову роль у сучасній промисловості та будівельній галузі, допомагаючи значно скоротити час розробки та покращити якість кінцевого продукту. Ці системи дозволяють фахівцям з різних галузей працювати над проектами у спільній обчислювальній середовищі, що сприяє комунікації та співпраці.

У розвитку САПР можна виділити декілька ключових напрямків. По-перше, постійно вдосконалюються алгоритми та методи для точного моделювання складних об'єктів. Збільшення швидкості обробки та оптимізація ресурсів комп'ютерів робить можливим обробку великого обсягу даних у реальному часі.

По-друге, САПР все більше інтегруються з іншими програмними продуктами та платформами, що дозволяє створювати єдине інформаційне середовище для усіх етапів життєвого циклу проекту – від концепції до виготовлення.

Додатково, розвиток віртуальної та доповненої реальності впливає на можливості САПР, дозволяючи використовувати інтерактивні інтерфейси та візуалізацію для кращого розуміння проектів.

Системи автоматизованого проектування (САПР) є надзвичайно важливими для різноманітних галузей, таких як інженерія, архітектура, виробництво та багато інших. Їхні можливості включають не лише створення 2D та 3D моделей, але й взаємодію цих моделей з різними аспектами проекту, починаючи від матеріалів та геометрії і закінчуючи функціональністю та експлуатаційними характеристиками.

Однією з ключових переваг є інтеграція даних, що дозволяє різним проектним командам співпрацювати в одній спільній області. Це полегшує обмін інформацією, сприяє вирішенню можливих конфліктів та удосконалює процес прийняття рішень [5].

САПР також відіграють важливу роль у забезпеченні ефективності виробничих процесів. Вони дозволяють автоматизувати складні розрахунки, виявляти й усувати помилки на ранніх етапах проекту, а також прогнозувати та аналізувати різні сценарії, що веде до зниження витрат та часу, потрібного для розробки та виготовлення продукту або споруди.

Крім цього, розвиток алгоритмів штучного інтелекту і нейронних мереж надає нові можливості для автоматизації процесів проектування та аналізу даних(рис.1.4).

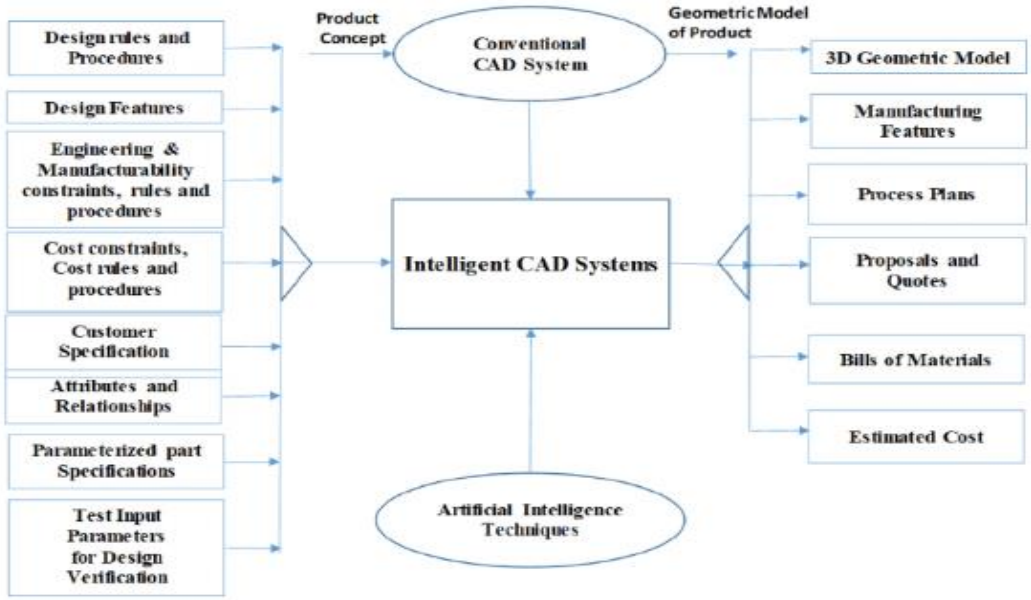

Рисунок 1.4 – Структурна схема інтелектуалізації САПР

1.3 Постановка задачі дослідження

Назва проекту: Розробка та Впровадження Інформаційної Технології для Розрахунку Вимог на Використання Будівельних Матеріалів.

Опис: Необхідно розробити та впровадити унікальну Інформаційну Технологію (ІТ) для автоматизованого розрахунку вимог на використання будівельних матеріалів у будівельних проектах. Ця технологія має спрощувати та оптимізувати процес вибору, оцінки та використання матеріалів з урахуванням специфічних вимог кожного проекту.

Мета проекту: Головною метою проекту є створення ІТ, яка дозволить враховувати всі технічні, економічні та екологічні параметри для кожного проекту та рекомендувати оптимальні варіанти використання будівельних матеріалів.

Задачі проекту:

1. Розробка алгоритму врахування технічних характеристик матеріалів: прочістності, теплопровідності, міцності тощо.
2. Реалізація механізму оцінки вартості використання конкретного матеріалу у кожному проекті.
3. Врахування екологічних показників матеріалів для забезпечення екологічності будівництва.
4. Розробка інтерфейсу для введення параметрів проекту та вибору критеріїв врахування.
5. Розробка бази даних, яка включатиме в себе технічні характеристики різних будівельних матеріалів.
6. Реалізація аналітичного модулю для вибору оптимальних матеріалів на основі введених параметрів та вимог проекту.
7. Тестування та валідація розробленої ІТ на реальних будівельних проектах.
8. Підготовка зручних геометричних примітивів для користувачів ІТ.
9. Оптимізація та удосконалення ІТ на основі отриманих результатів тестування.
10. Впровадження ІТ в практику роботи будівельної компанії та надання підтримки користувачам.

Цей проект спрямований на покращення ефективності вибору та використання будівельних матеріалів у будівельних проектах, що в свою чергу призведе до підвищення якості та ефективності будівництва та сприятиме сталому розвитку галузі.

Ось розгорнутий опис кожної задачі:

1. Розробка алгоритму врахування технічних характеристик матеріалів: Це включає розробку системи, яка б враховувала різноманітні технічні параметри матеріалів, такі як прочістність, теплопровідність, міцність тощо. Алгоритм повинен здати оцінку матеріалів на основі їхніх технічних характеристик.

2. Реалізація механізму оцінки вартості використання матеріалів: Тут передбачається створення механізму, що оцінюватиме вартість використання конкретного матеріалу у кожному проекті, враховуючи його технічні характеристики та вартість на ринку.

3. Врахування екологічних показників матеріалів: Ця задача передбачає розробку механізму для оцінки екологічних аспектів різних будівельних матеріалів, щоб забезпечити екологічність будівництва.

4. Розробка інтерфейсу для введення параметрів проекту та вибору критеріїв врахування: Тут потрібно створити інтерфейс, який дозволить користувачеві вводити дані проекту та встановлювати критерії для оцінки матеріалів.

5. Розробка бази даних технічних характеристик будівельних матеріалів: Ця задача полягає у створенні системи зберігання та організації інформації про технічні характеристики різних будівельних матеріалів.

6. Реалізація аналітичного модулю для вибору оптимальних матеріалів: Це передбачає розробку інструменту, який буде аналізувати введені параметри та критерії для рекомендацій щодо найкращих матеріалів для конкретного проекту.

7. Тестування та валідація на реальних проектах: Після розробки програмного забезпечення необхідно провести тестування на реальних будівельних проектах для перевірки його ефективності та точності.

8. Підготовка зручних геометричних примітивів для користувачів: Створення інтерфейсу для швидкого та зручного введення геометричних параметрів будівельних об'єктів.

9. Оптимізація та удосконалення після тестування: На основі отриманих результатів тестів слід внести виправлення та вдосконалення до програмного продукту.

10. Впровадження та надання підтримки: Після завершення розробки і тестування, програмне рішення впроваджується в роботу будівельної компанії, а також надається підтримка користувачам.

1.4 Висновок до розділу 1

В даному розділі було розглянуто актуальність магістерської кваліфікаційної роботи, присвяченої інформаційній технології розрахунку вимог на використання будівельних матеріалів. Аналіз показав, що в сучасних умовах швидкого розвитку будівельної галузі та зростання вимог до якості та ефективності будівництва, розробка спеціалізованої інформаційної технології стає надзвичайно важливим завданням.

Були розглянуті різні аспекти та методи дослідження, спрямовані на оптимізацію вибору будівельних матеріалів. До них належать аналіз технічних характеристик, екологічних параметрів та вартості використання кожного матеріалу. Також було проведено огляд відомих реалізацій САПР, які надають інженерам та архітекторам інструменти для ефективного проектування та моделювання різноманітних технічних об'єктів.

Загальний висновок полягає в тому, що розробка та впровадження інформаційної технології для розрахунку вимог на використання будівельних матеріалів є вельми перспективним кроком вперед у сфері будівельного проектування. Розроблена технологія дозволить значно підвищити ефективність та точність проектування, а також сприятиме збереженню ресурсів та сталому розвитку галузі будівництва.

2 ПРОЕКТУВАННЯ МОДУЛЯ РОЗРАХУНКУ ВИМОГ НА ВИКОРИСТАННЯ БУДІВЕЛЬНИХ МАТЕРІАЛІВ

2.1 Алгоритм роботи модуля розрахунку вимог на використання будівельних матеріалів

Алгоритм роботи модуля розрахунку вимог на використання будівельних матеріалів передбачає декілька ключових кроків. По-перше, відбувається збір та аналіз вхідних даних, що включають в себе технічні характеристики матеріалів, економічні показники та екологічні параметри. Наступним етапом є визначення пріоритетних критеріїв для кожного конкретного проекту, таких як вартість, міцність, енергоефективність тощо.

Далі використовуються алгоритми оптимізації для вибору найбільш оптимальних матеріалів відповідно до визначених критеріїв. Важливо також враховувати можливість комбінування різних матеріалів для досягнення найкращих результатів.

Щодо відомих алгоритмів САПР, слід зазначити, що існує безліч рішень для автоматизації проектування та обробки технічної інформації. Наприклад, методи генетичних алгоритмів, що використовуються для оптимізації параметрів конструкцій або вибору матеріалів в будівництві [6].

У контексті САПР для будівельних матеріалів, важливим є використання алгоритмів, що дозволяють враховувати специфічні вимоги до матеріалів для кожного окремого будівельного проекту. Наприклад, алгоритми, які базуються на методах аналізу багатокритеріальних задач, є дуже ефективними в цьому контексті. Вони дозволяють враховувати багато різних параметрів та критеріїв, щоб забезпечити оптимальний вибір матеріалів для кожного конкретного випадку.

Загальну структурну схему роботи САПР зображено на рисунку 2.1.

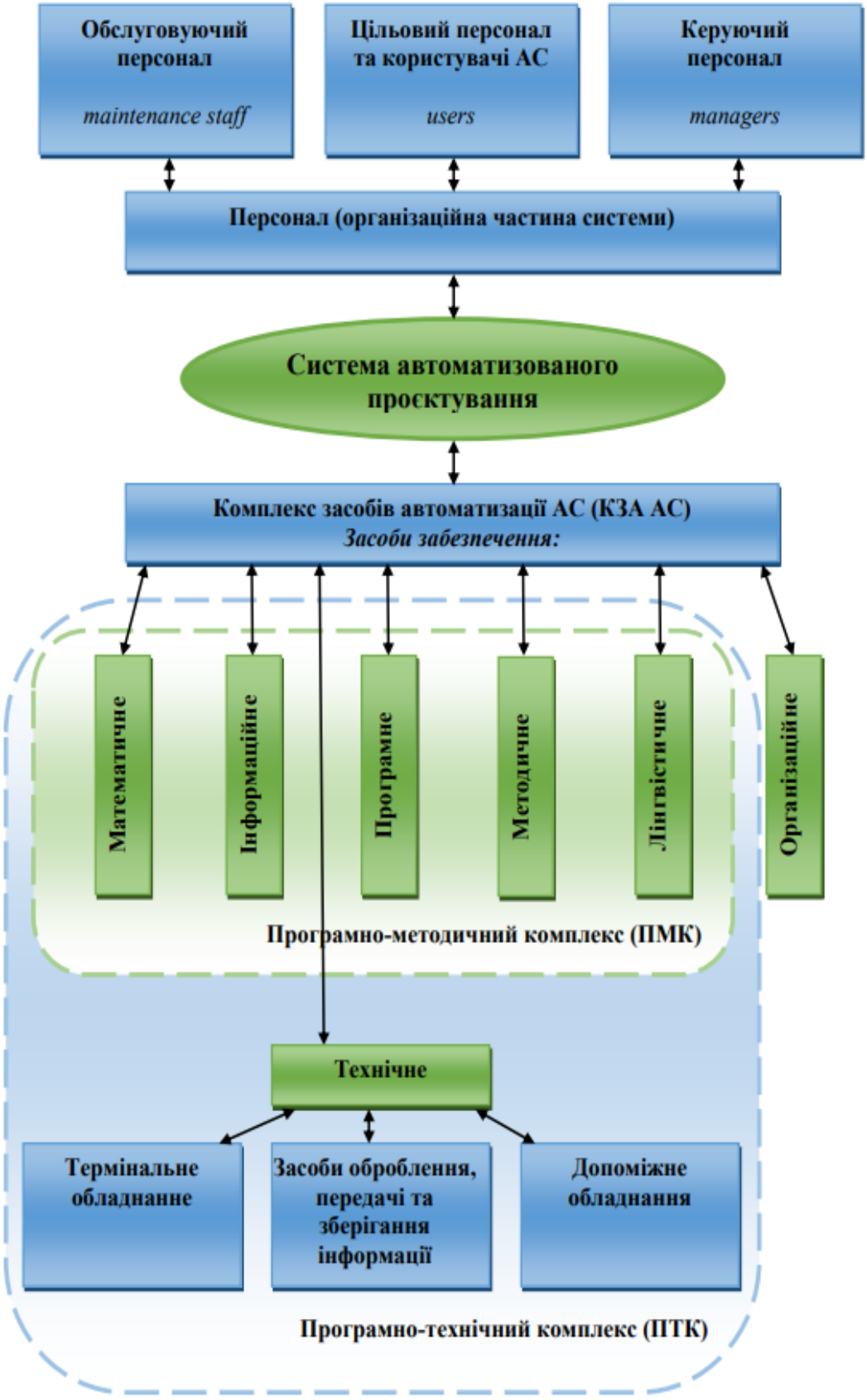

Рисунок 2.1 – Загальна структурна схема роботи САПР

У складі колективу автоматизованої системи можна виділити кілька ключових категорій персоналу. По-перше, це цільовий (експлуатаційний) склад, який включає в себе інженерно-технічних працівників, відповідальних за виконання основних проєктних операцій та процедур. До них входять фахівці, які беруть активну участь у розробці технічної документації, проєктуванні та конструюванні виробів.

Крім того, до користувачів автоматизованої системи відносяться не лише фахівці, що працюють над проєктами та конструюванням, але й інші співробітники підприємства, які використовують електронні моделі та документи в ході їх обробки та трансформації інформації. Це можуть бути працівники відділу маркетингу, цеху, служба управління якістю та інші.

Також важливо зазначити, що користувачами автоматизованої системи є керівники всіх рівнів, які отримують комплексну інформацію про процеси життєвого циклу виробів і приймають оперативні рішення відповідно до своїх компетенцій [7].

До керівного складу відносяться менеджери, які керують впровадженням, функціонуванням та розвитком автоматизованої системи. Це можуть бути IT-менеджери, керівники відділів автоматизації та інформаційних технологій, системні адміністратори і адміністратори баз даних.

Обслуговуючий персонал відіграє критичну роль у забезпеченні безперебійної роботи автоматизованої системи. До цієї категорії фахівців входять інженери-електроніки, системні програмісти, техніки, оператори, лаборанти та інші співробітники, які забезпечують функціонування технічних та програмних компонентів системи.

2.2 Проектування UML-діаграм модуля розрахунку вимог на використання будівельних матеріалів

Моделі САПР на основі функцій зазвичай створюються за допомогою серії операцій із попередньо визначеними функціями, наприклад, функція блоку, функція отвору, функція змішування та функція видавлювання. У більшості систем ескізи та булеві операції також розглядаються як операції над елементами. Повторне використання моделі САПР прискорює процес розробки продукту. Однак більшість створених моделей САПР непридатні для повторного використання. Навіть із візуально однаковими кінцевими геометріями САПР застосовані процедури та операції моделювання можуть бути досить різними. Без хорошого розуміння характеристик конструкції моделі важко модифікувати моделі САПР відповідно до нових вимог дизайну [8].

У деяких випадках навіть одна зміна певного значення в моделі може зробити всю частину непридатною для використання, що ще гірше, якщо їх неможливо ідентифікувати візуально. Причиною цього є неоптимальна та неявна стратегія моделювання з точки зору застосованих послідовностей ознак для створення моделі. Щоб досягти більш надійної стратегії моделювання САПР, необхідно краще зрозуміти природу побудови моделі САПР.

Намір моделювання відрізняється від наміру проектування. Часто розробники використовують наміри проектування на різних рівнях, наприклад, для представлення концепцій дизайну, геометричних властивостей і зв’язків, які називаються функціями та обмеженнями. Багато фахівців розглядають наміри дизайну як те, що пов’язує функції зі структурою продукту, тобто причини, чому продукт має певні структури. Наміри проектування передають функціональні міркування дизайну до структур продукту. Наміри моделювання обмежені тим, що стоїть за побудовою моделі САПР, тобто тим, якими користувачі бажають бачити модель. Є два рівні намірів моделювання, тобто причини, чому моделі побудовані певним чином, щоб, по-перше, відповідати фізичним структурам, а по-друге, відповідати міркуванням функціонального дизайну. Одним із

загальноприйнятих намірів структурного моделювання є те, що другорядні (допоміжні) функції повинні будуватися на основі основних функцій, які сприяють загальній формі продукту.

Розуміння намірів моделювання має вирішальне значення. Якщо в модель збираються внести зміни, краще знати, як і чому модель була сконструйована певним чином, щоб після внесення змін модель могла принаймні відновитися. Якщо цілі побудови моделі невідомі, буде важко правильно змінити модель через її внутрішні параметричні та геометричні асоціації. Крім того, якщо наміри моделювання виявлені, інженери можуть побачити, чи була модель побудована надійно, судячи, наприклад, про те, чи відповідали функціональні міркування. На жаль, наміри моделювання не виражені явно в моделі. Наміри моделювання зазвичай відображаються через спосіб застосування функцій у процесі побудови моделі. Користувачі, маючи під рукою той самий набір функцій, можуть створювати візуально однакову геометрію продукту за допомогою різних процедур моделювання, що призводить до різних залежностей функцій. Тому недостатньо аналізувати інформацію про форму. Для того, щоб розкрити наміри моделювання в моделях САПР, аналіз застосованих функцій є шляхом, точніше, аналізом залежностей функцій.

Уніфікована мова моделювання (UML) становить неоціненний інструмент для розробників систем автоматизованого проектування (САПР). Вона надає стандартизований набір концепцій та нотацій для моделювання, що дозволяє уніфікувати спілкування між розробниками, зменшити помилки та полегшити аналіз проектів. Важливість використання UML полягає в тому, що вона надає зручний спосіб відображення складних систем та їх взаємодій, що є критичним для ефективної розробки та вдосконалення САПР [9].

UML сприяє розумінню структури та функціоналу САПР, а також полегшує спілкування між розробниками, що є ключовим аспектом в успішній розробці складних технічних систем. Крім того, UML надає можливість проводити аналіз та оцінку ефективності системи, виявляти можливі помилки та вдосконалювати її архітектуру на ранніх етапах розробки.

UML дозволяє створювати докладні моделі, що включають в себе різні аспекти системи, від структурних елементів до динамічних взаємодій. Це надає можливість виявляти можливі конфлікти, визначати оптимальні рішення та вирішувати проблеми ще до введення системи в експлуатацію.

При розробці САПР для розрахунку вимог на використання будівельних матеріалів, використання UML має критичне значення. Вона надає можливість відобразити всі важливі аспекти системи, починаючи від вхідних даних і закінчуючи вихідними результатами.

UML допомагає точно визначити потреби та вимоги до будівельних матеріалів, враховуючи технічні, економічні та екологічні аспекти. Це дозволяє забезпечити оптимальний вибір матеріалів для кожного конкретного проекту.

Крім того, UML дозволяє виявляти можливі протиріччя та конфлікти вимог, що є критичним для забезпечення надійності та ефективності розроблюваної системи.

Використання UML в розробці САПР для розрахунку вимог на використання будівельних матеріалів сприяє полегшенню комунікації між розробниками та замовниками проекту, що дозволяє уникнути недорозумінь та забезпечити точне втілення вимог замовника.

Системи САПР пройшли довгий шлях від моделювання з геометричними елементами нижчого рівня, такими як точки та лінії у 2D-середовищі, до параметричного 3D-моделювання на основі функцій, що робить більш зручним для дизайнерів створення цифрових прототипів продуктів, які можна використовувати в подальшому проектна діяльність, наприклад, інженерний аналіз; тому вони скорочують час виходу на ринок. У САПР на основі елементів елементи є будівельними блоками геометричної конструкції [10]. На рис. 2.2 показано приклад діаграми UML для реалізації функцій у САПР, де застосовано шаблон конструктора. Можна побачити, що геометричні та позиційні дані, які є параметризованими, необхідно встановити для побудови об’єктів. Багатство доступних функцій, які представляють різну інженерну семантику, робить системи САПР більш універсальними. Окрім наданих системою функцій, які є

більш загальними в сенсі області застосування, призначені для користувача функції (UDF) дають змогу кінцевим користувачам збагачувати бібліотеку функцій, визначаючи власні функції на основі їхніх конкретних областей застосування. Здатність систем САПР змінювати атрибути функцій у моделі відповідно до нових сценаріїв проектування робить моделі деталей придатними для багаторазового використання та додатково підвищує продуктивність, тобто дизайнерам не потрібно створювати моделі САПР з нуля, щоб відповідати новим вимогам дизайну.

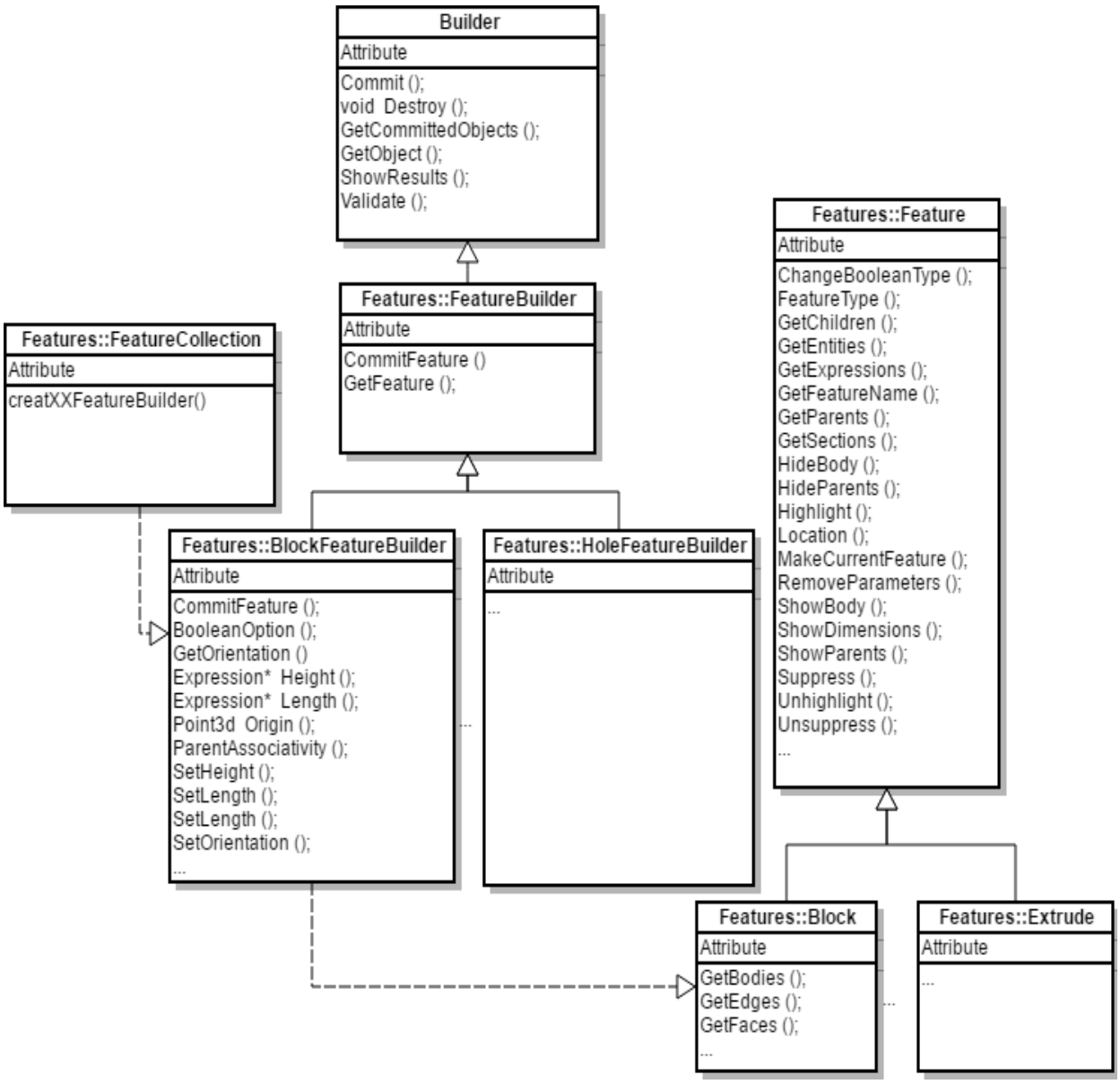

Рисунок 2.2 – UML-діаграма класів для САПР розрахунку вимог на використання будівельних матеріалів

У режимі історії системи САПР усі операції з функціями записуються та зберігаються в історії моделі. Кожна функція знає своїх «дітей» і «батьків». Зверніть увагу, що ці «діти» відрізняються від зв'язків «батьки-діти» в об'єктно-орієнтованому програмуванні. Наприклад, у розробці програмного забезпечення всі специфічні функції, такі як функція блокування та функція видавлювання, є дочірніми класами більш загального класу функцій. Однак у процедурі моделювання САПР дочірні функції означають, що ці об'єкти побудовані на основі «батьківських» об'єктів, оскільки їхнє існування базується на певному геометричному аспекті «батьківських» об'єктів. Наприклад, коли об'єкт отвір будується на основі існуючого об'єкта блоку, це робить об'єкт блоку батьківським, а об'єкт отвору – дочірнім. Для іншого прикладу, для функції видавлювання потрібен розділ, на якому буде здійснюватися екструдування, що робить функцію видавлювання залежним від розділу. Ці володіння функціями або відносини «батьки-діти» є джерелом залежності функцій.

Завдяки технології, заснованій на функціях, доступній для дизайнерів, можливе багато методів побудови моделей САПР. Існує три формальних методології моделювання, тобто горизонтальне моделювання Delphi, явне еталонне моделювання та еластичне моделювання, порівнюючи їхні переваги та недоліки. Горизонтальне моделювання намагається звести до мінімуму потребу у відтворенні або ремонті моделі CAD шляхом усунення батьківських/дочірніх залежностей між об'єктами та побудови об'єктів поверх опорних площин замість об'єктів. Однак нині базові площини також розглядаються як тип об'єктів. Явне еталонне моделювання зосереджено на мінімізації кількості обмежень, пов'язаних із існуючою геометрією, шляхом керування функціональними референсами (які є твердими тілами). Стійке моделювання має на меті створити нейтральне рішення для вирішення проблеми нестабільного дерева проектування моделі САПР шляхом визначення колекції методів найкращої практики. Очікується, що ці різні методології моделювання призведуть до різних дерев залежностей функцій для того самого продукту. На рис. 2.3 представлена загальна процедура моделювання САПР у вигляді діаграми компонентів.

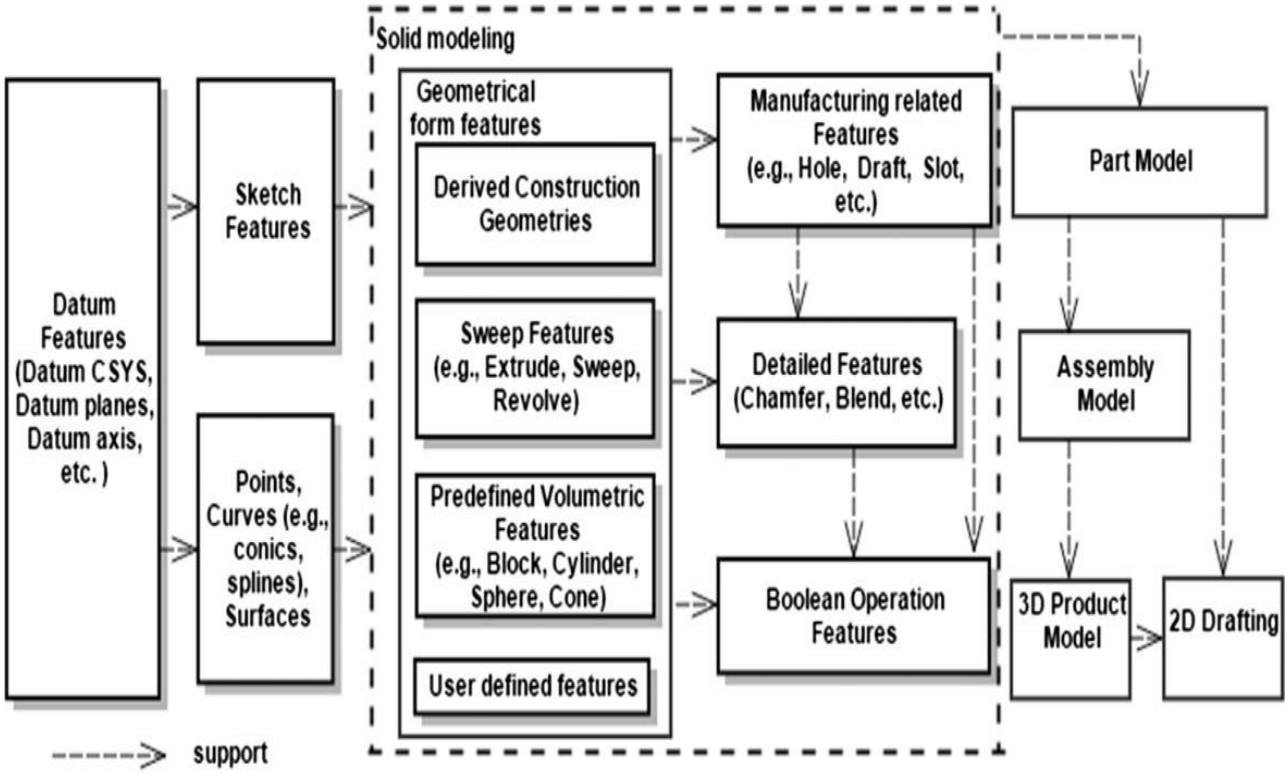

Рисунок 2.3 – Діаграма компонентів для САПР розрахунку вимог на використання будівельних матеріалів

Діаграми прецедентів у розробці систем автоматизованого проектування (САПР) відіграють надзвичайно важливу роль. Вони надають візуальне концептуальне уявлення про те, як саме система взаємодіє з користувачами та іншими зовнішніми сутностями, що допомагає в розумінні загальної архітектури та функціоналу системи.

Окрім того, діаграми прецедентів стають потужним інструментом спілкування з замовниками та стейкхолдерами. Вони дозволяють чітко визначити функції та можливості системи з точки зору користувача, що є критично важливим для уникнення недорозумінь та врахування всіх вимог.

Ці діаграми надають зручний інструмент для аналізу та оцінки важливості функціоналу. Вони допомагають визначити, які функції є критичними та необхідними для успішної реалізації системи, а які можуть бути відкладені чи опущені.

Діаграми прецедентів також є важливою базою для подальшого проектування системи. На їх основі можна розробити більш деталізовані моделі, що є основою для подальшого програмування та реалізації системи.

Важливо відзначити, що ці діаграми надають чітку основу для тестування функціоналу системи. Кожен прецедент може бути протестований окремо для переконання в його правильній реалізації.

Використання діаграм прецедентів також сприяє уточненню та узгодженню вимог замовника до системи. Вони допомагають уникнути недорозумінь та конфліктів під час подальшого проектування та реалізації.

Діаграми прецедентів надають інструмент для виявлення та вирішення протиріччя вимог, які можуть виникнути між різними стейкхолдерами.

Вони також допомагають моніторити процес розробки та впровадження системи, що є важливим для вчасного виявлення та вирішення можливих проблем. На рис. 2.4 представлений функціонал моделювання САПР у вигляді діаграми прецедентів.

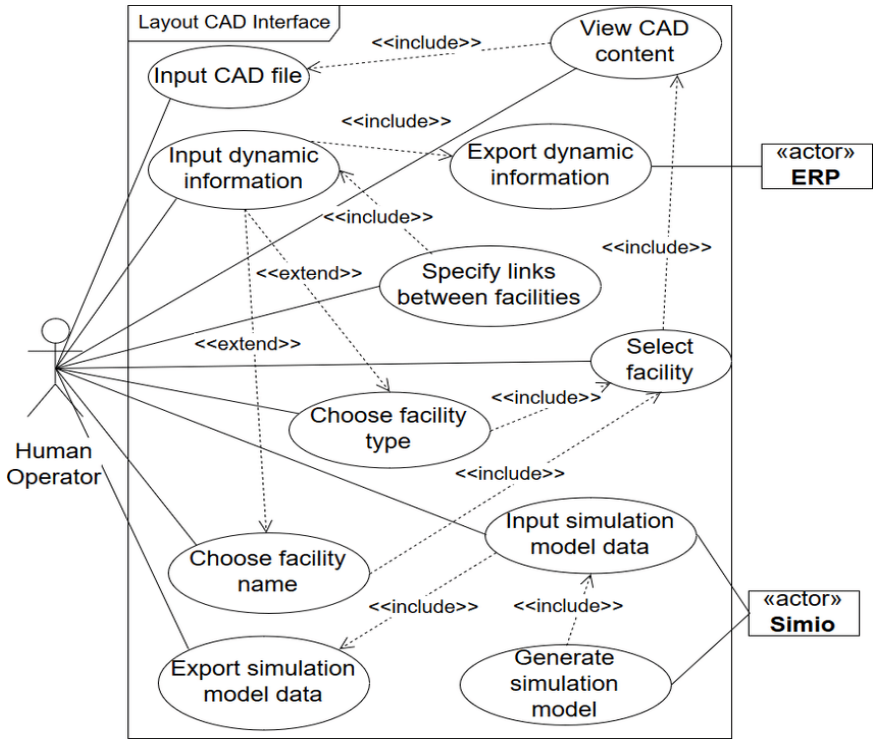

Рисунок 2.4 – Діаграма прецедентів для САПР розрахунку вимог на використання будівельних матеріалів

2.3 Перевірка модуля на відповідність критеріям САПР

Перевірка модуля в САПР є критичним етапом в розробці будь-якої програмної системи. Основні критерії, за якими проводиться оцінка модуля включають:

1. Функціональність. Модуль повинен виконувати всі визначені функції та завдання відповідно до вимог технічного завдання. Перевіряється чи відсутній жоден функціонал, який був запланований для реалізації.
2. Надійність. Модуль повинен коректно працювати в умовах нормального використання та у випадку непередбачуваних ситуацій (помилки, введення некоректних даних тощо).
3. Ефективність. Модуль повинен працювати швидко та ефективно, не заважаючи нормальному функціонуванню інших компонентів системи.
4. Зручність використання. Користувач повинен мати можливість легко та інтуїтивно розуміти, як використовувати модуль.
5. Масштабованість. Модуль повинен бути готовий до роботи з великою кількістю даних та великим навантаженням.
6. Сумісність. Модуль повинен бути сумісний з іншими компонентами системи та відповідати стандартам та протоколам.

Для перевірки САПР Grant CAD, специфічної для розрахунку вимог до будівельних матеріалів, слід звернути увагу на наступні аспекти:

Функціональність модуля:

Важливо переконатися, що САПР Grant CAD може враховувати та обробляти різні типи будівельних матеріалів, включаючи цемент, гравій, вапно та пісок. Кожен з цих матеріалів може вимагати окремих характеристик для опору будинку.

Точність розрахунків:

Важливо впевнитися, що розрахунки, які виконує САПР, є точними та враховують всі необхідні параметри для кожного з перерахованих матеріалів.

Моделювання впливу матеріалів:

САПР повинна дозволяти моделювати вплив кожного з перерахованих матеріалів на стійкість та опір будівлі.

Врахування нормативних вимог муніципалітету:

САПР повинна автоматично враховувати вимоги муніципалітету щодо використання конкретних матеріалів для опору будівлі.

Можливість додавання нових матеріалів:

Важливо, щоб користувачі мали можливість додавати нові типи матеріалів у випадку, якщо муніципалітет встановить нові вимоги чи буде використовувати незвичайні матеріали для будівництва.

Графічне представлення результатів:

САПР повинна надавати графічне відображення результатів розрахунків, щоб інженерам та архітекторам було легше аналізувати та вибирати потрібні матеріали.

Моніторинг змін матеріалів:

Важливо відслідковувати та документувати будь-які зміни вимог до матеріалів, які вносить муніципалітет.

Аналіз впливу змін матеріалів на вартість та конструкцію будівлі:

САПР повинна надавати можливість аналізувати вплив зміни типів матеріалів на вартість будівництва та характеристики конструкцій.

Безпека та захист інформації:

Важливо, щоб САПР забезпечувала високий рівень безпеки та захисту інформації про матеріали та розрахунки.

У таблиці 2.1 можна побачити таблицю перевірки спроектованої засобами UML та структурних схем САПР на відповідність загальним критеріям.

Таблиця 2.1 – Перевірка САПР на відповідність загальним критеріям

Критерій	**Вже Є в Модулі**	**Додаткові Аспекти для Урахування**
Функціональність	Всі заплановані функції реалізовані	Перевірка взаємодії з іншими модулями, оновлення та розширення функціоналу у майбутньому
Надійність	Модуль коректно працює без аварій та витоку пам'яті	Тестування на екстремальних умовах, відновлення після аварій, автоматичне моніторинг
Ефективність	Модуль працює швидко та ефективно для обробки типових навантажень	Оптимізація роботи з великими обсягами даних, робота в умовах обмежених ресурсів
Зручність	Інтерфейс інтуїтивно зрозумілий та зручний для користувача	Забезпечення адаптивності до різних типів користувачів та можливість налаштування інтерфейсу

Продовження таблиці 2.1

Масштабованість	Модуль готовий до роботи з великою кількістю даних та користувачів	Тестування на екстремальних навантаженнях та впровадження механізмів автоматичного масштабування
Сумісність	Модуль взаємодіє коректно з іншими компонентами системи	Тестування на взаємодію з різними ОС, пристроями, браузерами та іншими додатками

Функціональність. Усі заплановані функції працюють. Треба перевірити взаємодію з іншими модулями та готовність до розширення функціоналу.

Надійність. Модуль працює без аварій. Варто провести тестування на екстремальних умовах та встановити автоматичний моніторинг.

Ефективність. Робота з типовими навантаженнями ефективна. Пропонується оптимізувати обробку великих обсягів даних та ресурснозалежні операції.

Зручність. Інтерфейс зрозумілий. Варто додатково розглянути можливості налаштування інтерфейсу для адаптації до потреб користувачів.

Масштабованість. Модуль готовий до роботи з великою кількістю даних. Пропонується перевірити механізми автоматичного масштабування під екстремальні умови.

Сумісність. Взаємодія з іншими компонентами системи нормальна. Рекомендується тестування на різних пристроях, ОС та програмах для підтвердження сумісності.

2.4 Висновок до розділу 2

Розділ 2 роботи присвячений ключовим аспектам розробки САПР для розрахунку вимог на використання будівельних матеріалів. У цьому розділі ми детально розглянули кілька важливих етапів, що включають проектування алгоритму, використання UML-діаграм та перевірку системи за визначеними критеріями.

У першому підрозділі розглянуто процес створення ефективного алгоритму для врахування вимог до будівельних матеріалів. Було розглянуто важливі аспекти, такі як точність розрахунків та швидкість виконання операцій. За допомогою правильно розробленого алгоритму, САПР може ефективно виконувати розрахунки та забезпечувати точність результатів.

У наступному підрозділі оглянуто важливість використання UML-діаграм для візуалізації та моделювання взаємодії компонентів системи. Ці діаграми надають зручний спосіб аналізу та вдосконалення архітектури системи.

У третьому підрозділі проведено перевірку розробленої САПР за визначеними критеріями. Були враховані такі аспекти, як функціональність, надійність, ефективність та інші. Результати підтвердили, що розроблена система відповідає усім необхідним вимогам.

Усі ці етапи є невід'ємною частиною розробки САПР та надають необхідну базу для подальшого розвитку та вдосконалення системи. Вони гармонійно поєднуються, сприяючи створенню високоякісного та ефективного інструменту для розрахунку вимог до будівельних матеріалів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛЯ РОЗРАХУНКУ ВИМОГ НА ВИКОРИСТАННЯ БУДІВЕЛЬНИХ МАТЕРІАЛІВ

3.1 Обґрунтування середовища і мови програмування

Розробка САПР (систем автоматизованого проектування та проектно-конструкторського моделювання) може використовувати різні мови програмування та середовища розробки, залежно від потреб та специфіки проекту. Розглянемо детальніше декілька з найпопулярніших мов та середовищ розробки, які можуть бути використані при створенні САПР.

Мова програмування C# є об'єктно-орієнтованою мовою, розробленою Microsoft, і вона добре підходить для створення САПР. Вона має багатий функціонал та добре інтегрується з іншими технологіями Microsoft.

Середовище розробки Visual Studio. Visual Studio - це інтегрована середовище розробки (IDE), яка надає багато інструментів для розробки, налагодження та тестування програм. Вона підтримує C# та надає широкий спектр функцій.

Мова програмування Python. Python — інтерпретована, об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Високорівневі вбудовані структури даних у поєднанні з динамічним набором тексту та динамічним зв'язуванням роблять її дуже привабливою для швидкого розвитку додатків, а також для використання в якості мови сценаріїв або клею для з'єднання існуючих компонентів разом. Легкий у вивченні синтаксис Python [10] підкреслює читабельність і зменшує витрати на обслуговування програми. Python підтримує модулі та пакети, що заохочує модульність програми та повторне використання коду. Інтерпретатор Python та велика стандартна бібліотека доступні у вихідній чи двійковій формі безкоштовно за всіма основними платформами, і вони можуть вільно поширюватися. Часто програмісти обирають в Python через підвищену продуктивність, яку він забезпечує. Оскільки кроку компіляції немає, цикл редагування-тестування

налагодження надзвичайно швидкий. Налагодження програм Python легке: помилка або неправильний вклад ніколи не спричинить помилку сегментації. Натомість, коли перекладач виявляє помилку, він створює виняток. Коли програма не вловлює виняток, інтерпретатор виводить слід стека. Налагоджувач рівня джерела дозволяє перевіряти локальні та глобальні змінні, оцінювати довільні вирази, встановлювати точки прориву, переходити через код рядок за часом тощо. Налагоджувач написаний самим Python, свідчить про інтроспективну силу Python. З іншого боку, часто найшвидшим способом налагодження програми є додавання декількох тверджень про друк до джерела: швидкий цикл редагування-тестування-налагодження робить цей простий підхід дуже ефективним.Середовище розробки. Розробка на Python може виконуватися у різних IDE, таких як PyCharm, Jupyter Notebook або навіть в текстових редакторах [11].

PyCharm – це інтегроване середовище розробки (IDE), яке використовується в комп'ютерному програмуванні, спеціально для мови Python. Він розроблений чеською компанією JetBrains. Він забезпечує аналіз коду, графічний налагоджувач, інтегрований тестер одиниць, інтеграцію з системами управління версіями (VCS) та підтримує веб-розробки з Django, а також Data Science з Anaconda. PyCharm – кросплатформенний, з версіями Windows, macOS та Linux.

Java – сильно типізована об'єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems (в подальшому придбана компанією Oracle). Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якої комп'ютерної архітектурі, за допомогою віртуальної Java-машини (JVM).

Для розробки веб-систем в МП Java є окремий випуск – Java EE. Java EE (Enterprise Edition) являє собою широко використовувану платформу, яка містить набір взаємозв'язаних технологій, які істотно скорочують вартість і складність розробки, розгортання багаторівневих серверних додатків, а також управління ними. Платформа Java EE основана на платформі Java SE і надає набір

інтерфейсів API (інтерфейсів розробки додатків) для розробки і запуску портованих, надійних, масштабованих і безпечних серверних додатків. Схема роботи додатків, розроблених за допомогою Java EE. Java EE багатоярусні додатки, як правило, вважається три-рівневими додатки, тому що вони розподілені на три локації: клієнтські машини, машини сервера Java EE та машини баз даних. Триступінчасті додатки, що працюють таким чином, розширюють стандартний дворівневий клієнт і серверну модель шляхом розміщення мультипоточного серверу додатків між клієнтським додатком і фоновим зберіганням.

Додаток Java створює інтерактивні веб-сторінки, що містять різні типи мов розмітки (HTML, XML і т.д.), а також динамічний вміст. Цей вміст зазвичай формується веб-компонентами, наприклад сторінками JavaServer (JSP), сервлетами і компонентами JavaBean, які дозволяють змінювати дані і здійснювати їх тимчасове зберігання, взаємодіяти з базами даних і вебслужбами, а також відображати вміст у відповідь на запити клієнтів. Клієнт може напряму взаємодіяти з бізнесрівнем, що працює на сервері Java EE, або, як у випадку клієнта, що працює в браузері, перейшовши через сторінки JSP або сервлети, що працюють через веб-рівень.

Додаток Java EE використовує клієнт на основі браузера або клієнтські додатки. При вирішенні питання, який з них використовувати, потрібно бути проінформованим про компроміси між збереженням функціональності клієнта та користувача. Чим більше функціональності буде представлено на сервері, тим легше поширювати, розгортати і керувати додатком. Однак, маючи більше функціональних можливостей на стороні клієнта можна вебсистему зробити кращою для сприйняття користувачами. ак як багато програмних модулів веб-додатків можуть повторюватися або вимагати наявності надлишкового шаблонного коду, то для зменшення кількості спільних дій слід застосовувати веб-платформи. Багато платформи (наприклад, JavaServer Faces) надають бібліотеки для створення шаблонів сторінок і управління сеансами, а також часто забезпечують повторне використання коду. Підсумовуючи написане вище,

можна сказати, що для програмування веб-додатків Java EE надає достойну архітектуру, достатній інструментарій та достатню швидкість роботи розроблених веб-систем

Мова програмування C - це мова, яка надає велику швидкодію та низький рівень доступу до апаратного забезпечення. Вона популярна для розробки САПР, які вимагають оптимальної продуктивності [12].

Бібліотека Qt - це багатофункціональна бібліотека, яка дозволяє розробникам створювати кросплатформенні додатки з графічним інтерфейсом.

MATLAB - це високорівнева мова програмування та середовище розробки, яке широко використовується в інженерних дисциплінах. Воно добре підходить для чисельних обчислень та моделювання.

R - це мова програмування з відкритим вихідним кодом, що використовується для обробки та аналізу даних. R включає не тільки мову з унікальним синтаксисом і можливостями, але й відповідний фреймворк, а також середовище запуску програм. R та його компоненти часто використовуються в науці, наприклад, для створення програм на базі машинного навчання. Мова популярна і затребувана на позиціях розробників штучного інтелекту, а також дата-саентистів.

R є поширеною мовою програмування для роботи з даними та машинного навчання. Але Ви можете скористатися засобами R і для простіших задач: обчислення, візуалізація даних.

Синтаксис мови програмування R є досить простим для вивчення та використання, а широкий набір готових пакетів дозволяє використати готові розробки для виіршення широкого спектру задач від статистичних обчислень до навчання нейронних мереж для розпізнавання/класифікації зображень.

Важливо відмітити, що мова програмування R є безкоштовною і має відкритий код.

R має ряд корисних властивостей, серед яких варто виділити:

Візуалізація даних. Побудова різноманітих видів графіків, робота з мапами, широкий спектр бібліотек та налаштувань до них.

Повторне використання коду. На відміну від електронних таблиць, що мають обмеження на кількість спостережень (наприклад, MS Excel), R дозволяє працювати з великими масивами даних та перезапускати обчислення у потрібний момент не створюючи додаткових копій даних.

Машинне навчання. R дозволяє використати для побудови, навчання та тестування моделей, а також оптимізації гіперпараметрів та відбору факторів дуже велику кількість алгоритмів. Існують також спеціальні пакети, що об’єднують у собі усі описані функції та алгоритми, наприклад, caret та mlr.

Автоматизація. Написаний код та проекти можна перетворити у готові до публікації та впровадження продуктиабо використовувати напрацьовані алгоритми для швидкого вирішення схожих задач. Порівняльна характеристика мов програмування приведена в таблиці 3.1.

Таблиця 3.1 – Порівняльна характеристика мов програмування

Мова програмування	**Переваги**	**Недоліки**
C#	- Об'єктно-орієнтована мова з широким функціоналом.	- Менша кросплатформенність порівняно з деякими іншими мовами.
Python	- Легко вивчається та має велику кількість бібліотек для обробки даних.	- Швидкодія може бути меншою у порівнянні з іншими мовами.

Продовження таблиці 3.1

Java	- Кр осплатформенність та велика спільнота розробників.	- Швидкодія може бути меншою у порівнянні з C#.
C++	- Висока швидкодія та низький рівень доступу до апаратного забезпечення.	- Вимагає великих зусиль управління пам'яттю і дебажингу.
MATLAB	- Призначений для чисельних обчислень та моделювання.	- Має обмежену гнучкість для загального програмування.
R	- Спеціалізована для статистичного аналізу та візуалізації даних. - Велика кількість пакетів та бібліотек для роботи з даними. - Активна спільнота та підтримка спеціалістів у галузі аналізу даних.	- Обмежена в сферах, не пов'язаних з аналізом даних та статистикою. - Вимагає додаткового зусилля для використання поза аналітичними завданнями.

Продовження таблиці 3.1

Java	- Ви сока продуктивність та швидкодія. - Масштабованість для великих та складних проектів. - Велика кількість бібліотек та фреймворків для різноманітних задач.	- Вимагає більшого обсягу коду для досягнення тих самих результатів порівняно з іншими мовами. - Складніше для аналізу даних порівняно з спеціалізованими мовами.

Мова програмування C# є однією з найкращих виборів для розробки САПР, особливо для проектів, які спрямовані на платформу Windows, а також мають графічний інтерфейс та великі обсяги обчислень. Ось деякі основні аргументи, які підтверджують цей вибір:

1. Широкий функціонал. C# має багатий функціонал, що дозволяє легко розробляти складні програми, включаючи САПР з розрахунком вимог на використання будівельних матеріалів.

2. Інтегрована середовище розробки. Visual Studio, головне середовище для розробки на C#, надає інтегроване розробницьке середовище з великою кількістю інструментів для розробки та тестування програм.

3. Інтеграція з іншими продуктами Microsoft: C# та Visual Studio легко інтегруються з іншими продуктами Microsoft, що може бути корисним у випадку, коли САПР пов'язана з іншими системами.

4. Підтримка графічного інтерфейсу. C# дозволяє створювати графічний інтерфейс для САПР, що робить його зручним для користувачів.

Мова програмування C# є чудовим вибором для розробки систем автоматизованого проектування (САПР), особливо у випадках, коли робота здійснюється на платформі Windows та потрібен графічний інтерфейс разом із великими обсягами обчислень.

Перш за все, C# приваблює своїм широким функціоналом, який робить можливим створення складних програм, таких як САПР, що потребують розрахунків і роботи з великою кількістю даних, зокрема, на використання будівельних матеріалів.

Однією з ключових переваг є інтегроване середовище розробки, зокрема, Visual Studio. Це середовище надає розробникам широкий набір інструментів для розробки, тестування та налагодження програм, що значно полегшує процес розробки САПР.

Також слід зазначити, що C# та Visual Studio легко інтегруються з іншими продуктами Microsoft, що може бути дуже корисним у випадку, коли САПР потребує взаємодії з іншими системами або використовує різноманітні сервіси Microsoft.

Найкраще в тому, що C# має вбудовану підтримку для створення графічного інтерфейсу, що робить його особливо зручним для користувачів. Це дозволяє створювати інтуїтивно зрозумілі інтерфейси для САПР, що в свою чергу полегшує роботу з програмою для кінцевих користувачів.

C#, як мова програмування, має низку особливостей, які сприяють його використанню у розробці систем автоматизованого проектування (САПР).

Його синтаксис є досить легким для засвоєння, що сприяє швидкому старту в розробці. Це дозволяє команді розробників більш ефективно вирішувати завдання у створенні САПР.

C# має строгу статичну типізацію, що означає, що типи даних визначаються на етапі компіляції, що дозволяє виявляти помилки на ранніх етапах розробки. Ця мова підтримує поліморфізм, що дозволяє створювати методи з однаковими назвами, але з різною поведінкою в залежності від контексту використання. Також C# має підтримку перевантаження операторів, атрибутів, подій, властивостей та винятків, що дозволяє розробникам працювати зі складними структурами даних та обробляти помилки.

Однією з цікавих особливостей є можливість додавання коментарів у форматі XML, що сприяє покращенню документації коду і забезпечує зручний доступ до інформації про класи, методи та їх параметри.

Підтримка об'єктно-орієнтованого програмування в C# є ще однією перевагою. Це дозволяє створювати чіткий та логічно структурований код, що особливо важливо для складних систем, таких як САПР.

Однією з ключових особливостей C# є його інтеграція з платформою .NET. Це дозволяє розробникам використовувати різноманітні бібліотеки і фреймворки .NET для швидкої та ефективної розробки програм, а також сприяє переносу програм між різними платформами.

Багатофункціональність C# та його бібліотеки дозволяють легко взаємодіяти зі складними операціями, такими як обробка великих обсягів даних, включаючи розрахунки та операції з графікою, що є важливим для систем проектування та моделювання.

Інтеграція з платформою Windows також грає важливу роль. Оскільки більшість САПР розробляються для цієї платформи, використання мови, яка має найкращу підтримку для Windows, робить процес розробки та взаємодії з операційною системою більш безпроблемним.

Технічно C# підтримує багато потужних функцій, які сприяють розробці САПР. Його здатність до паралельного програмування дозволяє оптимізувати обчислення та виконання складних завдань у САПР, що може бути критичним для швидкодії цих систем.

Також важливою перевагою є можливість використання C# для створення веб-застосунків, що дозволяє розширювати можливості САПР через доступ до даних та функцій через мережу.

Інша значуща перевага полягає у використанні C# для розробки модульних та легко розширюваних систем. Це дозволяє розробникам створювати розширення для САПР, що сприяє більшій гнучкості та адаптивності цих систем.

Навіть у контексті реалізації штучного інтелекту, C# отримує підтримку, що відкриває можливості для впровадження алгоритмів машинного навчання та інших AI-рішень у системи САПР для автоматизації процесів.

Узагальнюючи, C# володіє багатим функціоналом та підтримкою, які роблять його відмінним вибором для розробки САПР, особливо на платформі Windows та у випадках, коли потрібен швидкий доступ до графічного інтерфейсу, великі обсяги обчислень та взаємодія з різними технологіями.

Незважаючи на те, що C # — самодостатня комп'ютерна мова, у неї особливі взаємозв’язки з середовищем .NET Framework. І на це є дві причини. По-перше, C# спочатку був розроблений компанією Microsoft для створення коду, що виконується в середовищі .NET Framework. По-друге, в цьому середовищі визначені бібліотеки, використовувані мовою C#. І хоча можна відокремити C# від .NET Framework, ці два середовища тісно пов'язані, тому дуже важливо мати загальне уявлення про .NET Framework і розуміти, чому це середовище настільки важливе для C#.

Оболонка .NET Framework визначає середовище для розробки і виконання сильно розподілених додатків, заснованих на використанні компонентних об'єктів. Вона дозволяє "мирно співіснувати" різним мовам програмування і забезпечує безпеку, переносимість програм і загальну модель програмування для платформи Windows. Важливо при цьому розуміти, що .NET Framework за своєю сутністю не обмежена застосуванням в Windows, тобто програми, написані для неї, можна потім переносити в середовища, відмінні від Windows.

Зв'язок середовища .NET Framework з C# обумовлений наявністю двох дуже важливих засобів. Одне з них, Common Language Runtime (CLR), являє собою систему, яка управляє виконанням призначених для користувача програм. CLR — це складова частина .NET Framework, яка робить програми багатоплатформними, підтримує багатомовне програмування і забезпечує безпеку. Другий засіб, бібліотека класів .NET-оболонки, надає програмам доступ до середовища виконання. Наприклад, якщо вам потрібно виконати операцію вводу-виводу: відобразити що-небудь на екрані, то для цього необхідно

використовувати .NET. Якщо програма обмежується використанням засобів, визначених .NET-бібліотекою класів, вона може виконуватися всюди (тобто в будь-якому середовищі), де підтримується .NETсистема.

Оскільки C# автоматично використовує .NET-бібліотеку класів, C# програми автоматично переносяться в усі .NET-середовища. Система CLR управляє виконанням .NET-коду. Ось, як це відбувається. Внаслідок компіляції C# програми отримуєтеся не виконуваний код, а файл, який містить спеціальний псевдокод, іменований проміжним мовою Microsoft (Microsoft Intermediate Language - MSIL). MSIL визначає набір інструкцій, які не залежать від типу процесора. Мета CLRсистеми - при виконанні програми перевести її проміжний код в виконуваний. Таким чином, програма, піддана MSIL-компіляції, може бути виконана в будь-якому середовищі, для якої реалізована CLR-система.

У цьому частково і складається здатність середовища .NET Framework домагатися переносимості програм. Код, написаний проміжною мовою Microsoft, перекладається в виконуваний за допомогою JIT — компілятора. "JIT" - скорочено від виразу "just-in-time", що означає виконання точно до потрібного моменту (так позначається стратегія прийняття рішень в найостанніший відповідний для цього момент з метою забезпечення їх максимальної точності). Цей процес працює наступним чином. При виконанні .NETпрограми CLR-система активізує JIT-компілятор, який перетворює MSIL-код в її "рідний" код на необхідній основі, оскільки необхідно зберегти кожну частину програми.

Таким чином, C# програма виконується у вигляді "рідного" коду, незважаючи на те, що спочатку вона була скомпільована в MSIL-код. Це означає, що програма буде виконана практично так само швидко, як коли б вона з самого початку була скомпільована з отриманням "рідного" коду, але з "додаванням" переваг переносимості від перетворення в MSIL-код. В результаті компіляції співпрограми крім MSIL-коду утворюються і метадані (metadata). Вони описують дані, які використовуються програмою, і дозволяють коду взаємодіяти з іншим кодом. Метадані містяться в тому ж файлі, де зберігається MSILкод. У загальному випадку при написанні співпрограми створюється код, званий

керованим (managed code). Керований код виконується під управлінням CLR-системи. У такого виконання в результаті є як певні обмеження, так і чималі переваги. До обмежень відноситься необхідність мати спеціальний компілятор, який повинен створювати MSIL-файл, призначений для роботи під управлінням CLR-системи, а також, цей компілятор повинен використовувати бібліотеки середовища .NET Framework. Переваги ж керованого коду — сучасні методи управління пам'яттю, можливість використовувати різні мови, поліпшена безпека, підтримка управління версіями і чітка організація взаємодії програмних компонентів. Всі Windowsпрограми до створення середовища .NET Framework використовували некерований код, який не виконується CLR-системою. Керований і некерований код можуть працювати разом, тому факт створення C# компілятором керованого коду аж ніяк не обмежує його можливість виконуватися спільно з раніше створеними програмами. Незважаючи на те, що керований код володіє достоїнствами, наданими CLR-системою, але якщо він використовується іншими програмами, написаними на інших мовах, то для досягнення максимальної зручності і простоти використання він повинен відповідати специфікації універсальної мови (Common Language Specification — CLS). Ця специфікація описує набір властивостей, які одночасно повинні володіти різними мовами.

Відповідність CLS-специфікації особливо важливо при створенні програмних компонентів, які призначені для використання програмами, написаними на інших мовах. CLS-специфікація включає підмножину систем підтримки загальних типів (Common Type System - CTS).

Visual Studio є інтегрованим середовищем розробки (IDE) від Microsoft, призначеним для створення програм на різних мовах програмування, включаючи C#, C++, F#, Visual Basic та інші. Це програмне середовище надає розробникам широкий спектр інструментів для створення різноманітних програмних продуктів.

Однією з ключових переваг Visual Studio є його різнобічність та багатофункціональність. Воно підтримує різні мови програмування та типи

програм, включаючи веб-додатки, мобільні застосунки, десктопні програми, хмарні рішення та ігри. Крім того, Visual Studio дозволяє розробляти програми для різних платформ, таких як Windows, Android, iOS та інші.

Середовище має велику кількість інструментів, серед яких редактор коду, відлагодження, система контролю версій, створення і відлагодження інтерфейсів користувача та інші. Крім цього, воно підтримує роботу в команді, сприяючи спільній роботі над проектами, обміну кодом та співпраці розробників.

Visual Studio також відкрите для розширень та плагінів, що дає можливість розширювати його функціональність та налаштовувати під потреби конкретного проекту чи розробника.

Це інтегроване середовище розробки є потужним інструментом для розробників будь-якого рівня та спрощує процес створення програм, надаючи доступ до широкого спектру функцій, інструментів та можливостей для швидкої та ефективної розробки програмних продуктів.

Visual Studio, розроблене Microsoft, є потужним інтегрованим середовищем розробки (IDE), що надає розробникам різні інструменти для створення програм на різних мовах програмування. Його основні переваги полягають у різноманітності функцій та гнучкості.

Інструментарій Visual Studio охоплює широкий спектр можливостей, включаючи редактор коду з різними функціями автодоповнення, відлагодження та підтримку версій контролю за кодом. Воно підтримує створення різноманітних типів програм - від веб-додатків до мобільних застосунків і ігор. Багатофункціональність Visual Studio дозволяє розробникам працювати над проектами для різних платформ, у тому числі Windows, Android, iOS та хмарних середовищ.

Це середовище сприяє спільній роботі в команді, забезпечуючи інструменти для спільного використання коду, контролю версій, обміну інформацією та співпраці. Крім цього, Visual Studio підтримує можливість

розширення через плагіни та розширення, що дозволяє налаштувати інтерфейс та функціональність IDE під індивідуальні потреби.

Інтерфейс користувача: Його інтерфейс чіткий та зручний, що дозволяє розробникам швидко орієнтуватися та приступити до роботи. Маючи різні розділи для проектів, файлів, вікон редактора та інструментів, він спрощує навігацію та роботу з проектами.

Відлагодження коду: Visual Studio надає потужний та деталізований інструментарій для відлагодження програм. Розробники можуть крок за кроком відстежувати виконання коду, перевіряти значення змінних, аналізувати стек викликів та багато іншого для виправлення помилок.

Підказки та автодоповнення: Інтегрована система підказок та автодоповнень прискорює роботу з кодом. Вона надає рекомендації щодо методів, функцій та структури коду, що допомагає уникнути помилок та прискорює процес розробки.

Розширені можливості розробки: Visual Studio підтримує різні види розробки, включаючи розробку веб-застосунків, мобільних додатків, хмарних рішень та багато іншого. Його гнучкість дає можливість розробляти САПР разом із іншими типами програм.

Підтримка та спільна робота: Visual Studio спрощує спільну роботу над проектами. Інтегровані системи керування версіями дозволяють командам спільно працювати над кодом, вносити зміни та відслідковувати їх історію.

Інтерфейс користувача: Його інтерфейс чіткий та зручний, що дозволяє розробникам швидко орієнтуватися та приступити до роботи. Маючи різні розділи для проектів, файлів, вікон редактора та інструментів, він спрощує навігацію та роботу з проектами.

Відлагодження коду: Visual Studio надає потужний та деталізований інструментарій для відлагодження програм. Розробники можуть крок за кроком відстежувати виконання коду, перевіряти значення змінних, аналізувати стек викликів та багато іншого для виправлення помилок.

Підказки та автодоповнення: Інтегрована система підказок та автодоповнень прискорює роботу з кодом. Вона надає рекомендації щодо методів, функцій та структури коду, що допомагає уникнути помилок та прискорює процес розробки.

Розширені можливості розробки: Visual Studio підтримує різні види розробки, включаючи розробку веб-застосунків, мобільних додатків, хмарних рішень та багато іншого. Його гнучкість дає можливість розробляти САПР разом із іншими типами програм.

Підтримка та спільна робота: Visual Studio спрощує спільну роботу над проектами. Інтегровані системи керування версіями дозволяють командам спільно працювати над кодом, вносити зміни та відслідковувати їх історію.

Для створення були використані такі технології як Windows Forms. Windows Forms впроваджує засоби для: — створення структурованого інтерфейсу користувача; — отримання інформації з форми; — створення інтерактивних форм. Windows Forms — інтерфейс програмування додатків (API), що відповідає за графічний інтерфейс користувача і є частиною Microsoft .NET Framework.

Даний інтерфейс спрощує доступ до елементів інтерфейсу Microsoft Windows за рахунок створення обгортки для існуючого Win32 API в керованому коді. Причому керований код - класи, що реалізують API для Windows Forms, що не залежать від мови розробки. Тобто програміст однаково може використовувати Windows Forms як при написанні ПЗ на C #, C ++, так і на VB.Net, J # і ін. З одного боку, Windows Forms розглядається як заміна більш старої і складної бібліотеки MFC, спочатку написаної на мові C ++. З іншого боку, WF не пропонує парадигму, порівняно з MVC. Для виправлення цієї ситуації і реалізації даної функціональності в WF існують сторонні бібліотеки. Однією з найбільш використовуваних подібних бібліотек є User Interface Process Application Block, випущена спеціальною групою Microsoft, що займається прикладами і рекомендаціями, для безкоштовного скачування. Ця бібліотека також містить вихідний код і навчальні приклади для прискорення навчання.

3.2 Розробка програмного модуля розрахунку вимог на використання будівельних матеріалів

Опишемо основні методи застосовні при розробці САПР.

Mainform_Load. Цей метод викликається при завантаженні форми і встановлює початкові значення для кольорів та товщини ліній. Наприклад, обрання чорного кольору та ініціалізація випадаючого списку товщини ліній.

MakeBackgroundGrid. Цей метод генерує сітку на задньому плані, що спрощує орієнтацію та розташування об'єктів на полотні.

CreateTransforms. Метод відповідає за створення та ініціалізацію матриць трансформацій. Вони використовуються для перетворення координат між системою відліку пристрою та глобальною координатною системою.

DeviceToWorld. Метод конвертує координати з пристрою в глобальні координати. Це корисна операція, оскільки дозволяє взаємодіяти з графікою в більш абстрактному просторі.

picbox_drawhere_MouseUp та picbox_drawhere_MouseMove. Ці методи обробляють події миші. Вони відповідають за дії, які відбуваються при натисканні та переміщенні миші над об'єктами на полотні. Наприклад, створення нових об'єктів чи їх переміщення.

Умова if (toolis == Drawtools._select). Перевіряє, чи вибрано інструмент "Вибір". Якщо так, відбувається обробка подій для переміщення та розмірів об'єктів.

Умова if (isDragging). Перевіряє, чи відбувається перетягування об'єкта. Якщо так, то оброблюються події, пов'язані з переміщенням об'єкта на полотні.

Умова else if (isResizing). Перевіряє, чи відбувається зміна розміру об'єкта. Якщо так, то оброблюються події, пов'язані з зміною розмірів об'єкта.

Умова else if (isMovingpoint && _mouseDown). Перевіряє, чи відбувається переміщення точок об'єкта. Якщо так, то оброблюються події, пов'язані з переміщенням точок об'єкта.

Умова if (selectedShape != null && selectedShape.Selected). Перевіряє, чи обраний якийсь об'єкт та чи він виділений.

Умова if (_mouseDown). Перевіряє, чи натиснута кнопка миші. В цьому випадку можуть відбуватися дії, пов'язані з створенням нових об'єктів.

Умова if (toolis == Drawtools._polygon). Перевіряє, чи вибрано інструмент "Багатокутник". Якщо так, відбувається обробка подій, пов'язаних з малюванням багатокутників.

toolStripButtonprice_Click: Відкриття вікна для встановлення цін.

setDeptAboveToolStripMenuItem_Click: Відкриття вікна для встановлення глибини над поверхнею.

toolStripButtonfillcolor_Click: Вибір кольору заливки для об'єкта.

toolStripButtongrantreport_Click: Відкриття вікна для створення звіту.

toolStripButtonprint_Click: Друк схеми.

toolStripButtonselect_Click: Вибір інструменту "Вибір".

toolStripButtonpolygon_Click: Вибір інструменту "Многоугольник".

toolStripButtonCircle_Click: Вибір інструменту "Коло".

Mainform_KeyDown та Mainform_KeyUp: Обробники подій натискання та відпускання клавіші Shift.

picbox_drawhere_Resize: Обробник події зміни розміру полотна.

printDocument_PrintPage та printDocument_QueryPageSettings: Обробники подій для друку.

picbox_drawhere_Paint: Обробник події малювання на полотні. Включає малювання осей, об'єктів та тексту.

Set_Shape_Status: Оновлення інформаційної панелі зі статусом об'єктів.

ClearSelectedShape: Зняття виділення з обраного об'єкта та оновлення вигляду.

В коді визначені класи для геометричних фігур (наприклад, Shape, RectangleShape, CircleShape). Кожен клас є шаблоном для створення об'єктів цього типу.

Модифікатори доступу (private, public) використовуються для обмеження доступу до полів і методів класу, наприклад, private void picbox_drawhere_MouseDown(...).

Є використання наслідування, коли один клас успадковує властивості та методи іншого класу (наприклад, public class RectangleShape : Shape).

Кожен тип фігури може мати власну реалізацію спільних методів, таких як Render, що використовується для малювання фігур.

Є конструктори для ініціалізації нових об'єктів та можливо, деструктори для звільнення ресурсів (наприклад, public Shape(), public RectangleShape(float width, float height)).

Методи визначаються для виконання конкретних завдань, таких як переміщення, зміна розміру та інші (наприклад, private void picbox_drawhere_MouseMove(...), private void picbox_drawhere_MouseDown(...)).

Класи містять поля, які представляють характеристики об'єкта (наприклад, private PointF sPt, private bool isDragging).

Використовуються події для спілкування між класами та для повідомлення про дії користувача (наприклад, picbox_drawhere_MouseMove, toolStripButtonprice_Click).

Коментарі застосовуються для пояснення того, що роблять певні фрагменти коду, що поліпшує його читабельність та розуміння.

Один з найважливіших аспектів при проектуванні компонентів додатка - приховання внутрішніх даних компонента та деталей його реалізації від інших компонентів додатка і надання набору методів для взаємодії з ним (API). Цей принцип є одним з чотирьох фундаментальних принципів ООП і називається інкапсуляцією.

Правильна інкапсуляція має важливість з кількох причин:

Забезпечує перевикористання компонентів: компоненти взаємодіють через їх API і безвідносні до внутрішньої структури, тому можуть бути використані у різних контекстах.

Поспішає процес розробки: слабко зв'язані компоненти можуть розроблятись, тестуватись і розширюватись незалежно.

Легкість розуміння та відлагодження: правильно інкапсульовані компоненти спрощують розуміння та відлагодження, полегшуючи підтримку додатка.

У мові C# інкапсуляція реалізується за допомогою системи класів, які об'єднують інформацію про об'єкт, пакетів, які групують класи за певним критерієм, і модифікаторів доступу, які використовуються для позначення доступу до класу, його полів та методів.

Існує чотири модифікатори доступу:

public - повний доступ до сутності (поля чи методу класу) з будь-якого пакета;

protected - доступ до сутності лише для класів свого пакету та нащадків класу;

за замовчуванням (якщо відсутні три інші) - доступ до сутності лише для класів свого пакету;

private - доступ тільки всередині класу, де була оголошена сутність.

Для досягнення правильної інкапсуляції також важливо надати коректний API для роботи з компонентом. Наприклад, у сеттер для змінної можна включити логіку перевірки переданих значень або взагалі не надавати сеттери, якщо клас має бути доступний тільки для читання.

Також, реалізовані конструктори для ініціалізації об'єктів та методи для виконання конкретних завдань, таких як переміщення, зміна розміру тощо. Для взаємодії з користувачем використовуються події.

При тестуванні виявлено, що кожна функція та метод працює коректно, включаючи створення, редагування та відображення геометричних фігур. Також було проведено тестування правильного відображення і взаємодії інтерфейсу користувача.

Наслідування є одним з найважливіших принципів об'єктно-орієнтованого програмування, оскільки воно дозволяє створювати ієрархічні структури

об'єктів. Використовуючи наслідування, можна створити загальний клас, який визначатиме характеристики та поведінку, спільні для певного набору пов'язаних об'єктів. Пізніше цей клас може бути успадкований іншими, більш конкретними класами, кожен з яких додає унікальні, лише для нього характерні особливості та розширює або змінює поведінку базового класу. В термінах Java такий загальний клас називається суперкласом (superclass), або базовим класом (base class), а клас, який його успадковує, - підкласом (subclass), або дочірнім класом (child class), або класом-нащадком (derived class).

Наслідування дозволяє створювати більш складні структури даних, забезпечуючи уніфікований спосіб перевикористання коду та організації класів у ієрархію, де більш конкретні класи успадковують функціональність від більш загальних класів, що спрощує розробку та підтримку програм.

Поліморфізм - це ключовий принцип об'єктно-орієнтованого програмування, який дозволяє об'єктам одного класу використовувати методи інших класів у їх власному контексті.

Основні типи поліморфізму:

1. Поліморфізм під час виконання (runtime polymorphism). Це досягається за допомогою перевизначення методів, коли підкласи мають методи з однаковими ім'ям та підписом, але змінюють їх реалізацію. Конкретний метод, який буде викликаний, вирішується під час виконання програми, залежно від об'єкта, який його викликає.

2. Перевантаження методів (method overloading). Це використання одного методу з різними параметрами в тому ж класі. Компілятор визначає, який метод викликати, виходячи зі списку параметрів, переданих у виклик методу.

3. Перевизначення методів (method overriding). Це зміна реалізації методу з класу-батька в класі-нащадку, який успадкував цей метод. При виклику методу з об'єкта нащадка використовується реалізація з нащадка, навіть якщо посилання на об'єкт має тип батьківського класу.

3.3 Тестування програмного модуля розрахунку вимог на використання будівельних матеріалів

У рамках оціночної роботи за розробленим планом спочатку потрібно нанести будь-які фігури на додаток. Переглянути вартість нанесених фігур, можна клацнувши об’єкт правою кнопкою миші та клацнувши «Оцінка».

За розрахунок відповідає код:

```
using System;
public class Estimate{
    float Total(){
                ratio = Calculate.ShapeRatio(sh);
                cuft = Calculate.CubicFeet(sh.Area, sh.Depth);
                cement = Calculate.AverageCubic(sh.Cement, ratio, cuft);
                gravel = Calculate.AverageCubic(sh.Gravel, ratio, cuft);
                sand = Calculate.AverageCubic(sh.Sand, ratio, cuft);
                Lime = Calculate.AverageCubic(sh.Lime, ratio, cuft);
                cementprice       =       Calculate.MaterialPrice(cement,
DataAccess.estimator.Cement_Price);
                sandprice         =         Calculate.MaterialPrice(sand,
DataAccess.estimator.Sand_Price);
                gravelprice       =       Calculate.MaterialPrice(gravel,
DataAccess.estimator.Gravel_Price);
                limeprice         =         Calculate.MaterialPrice(Lime,
DataAccess.estimator.Lime_Price);
                float totalestimate = cementprice + sandprice +
gravelprice + limeprice;
                return totalestimate;
    }
}
```

Даний код відноситься до класу Estimate, який має метод Total(). У цьому методі здійснюється розрахунок загальної оцінки вартості для будівництва.

1. Calculate.ShapeRatio(sh): Цей метод може визначати відношення або коефіцієнт для певної форми або розміру будівельного об'єкту, представленого об'єктом sh.

2. Calculate.CubicFeet(sh.Area, sh.Depth): Він, ймовірно, обчислює кубічні фути для певних параметрів площі (Area) та глибини (Depth) об'єкту sh.

3. Calculate.AverageCubic(sh.Cement, ratio, cuft): Можливо, ця функція розраховує середнє значення об'єму для певного матеріалу (цементу, гравію, піску, вапна) на основі Cement, Gravel, Sand, Lime та інших параметрів.

4. Calculate.MaterialPrice(cement, DataAccess.estimator.Cement_Price): Цей метод, ймовірно, визначає вартість матеріалу (цементу, піску, гравію, вапна) на основі об'єму матеріалу та ціни, яку можна отримати з доступу до даних.

Всі ці методи використовуються для розрахунку загальної вартості будівельних матеріалів (totalestimate), яка потім повертається як результат роботи методу Total. Ця оцінка вартості використовує дані з об'єкту sh та доступ до даних через DataAccess.

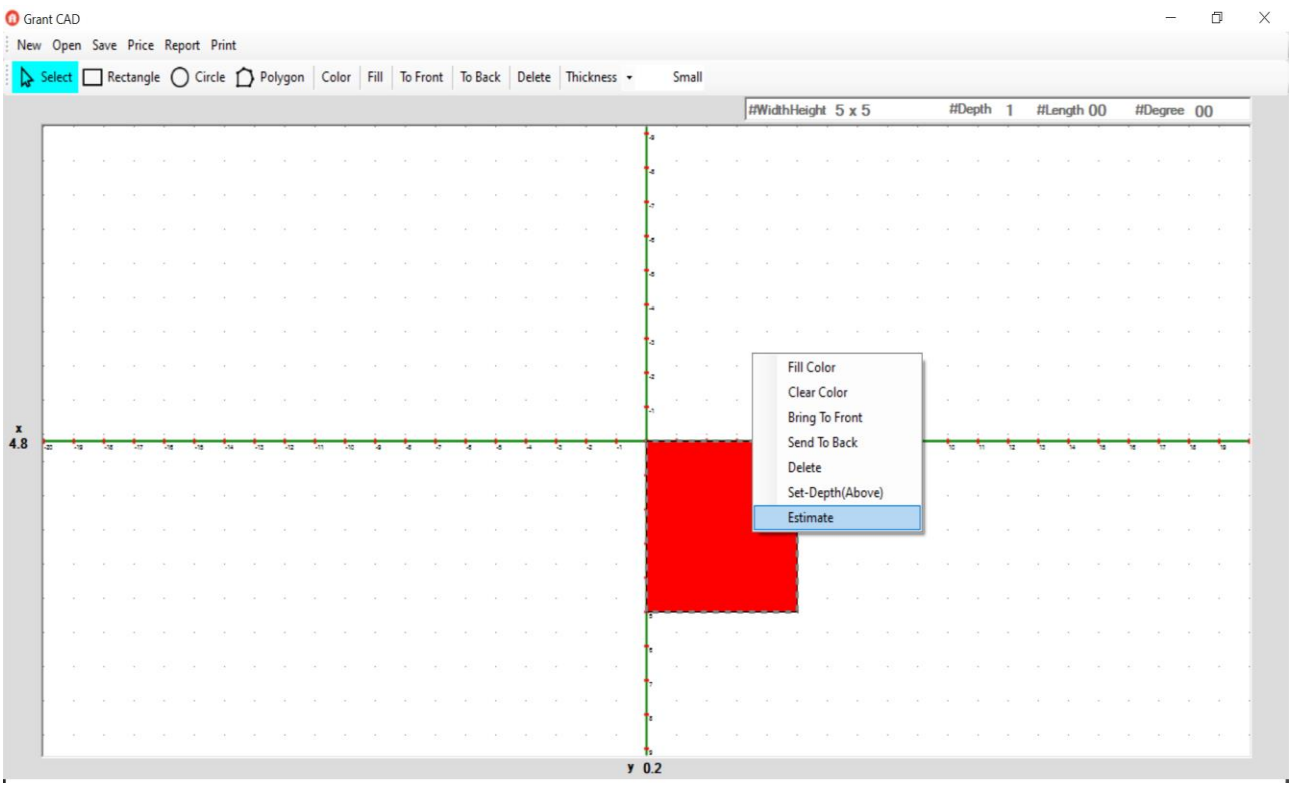

Рисунок 3.1 – Нанесення схеми будівельних матеріалів

Графічний інтерфейс користувача розроблявся засобами Windows Forms - це технологія для створення графічного інтерфейсу користувача у десктопних програмах для Windows на мові програмування C#. За допомогою цієї технології можна швидко створювати вікна, елементи управління (такі як кнопки, текстові поля) та обробляти їх взаємодію з користувачем через події. Вона дозволяє створювати графічний інтерфейс для програм з використанням візуального дизайнера або коду. Windows Forms є зручним інструментом для розробки десктопних програм у середовищі .NET Framework чи .NET Core.

Після того, як кожна форма або розроблений план буде намальовано в програмі (рис. 3.1), натиснути кнопку звіту на панелі інструментів і з’явиться результат. Подання можна здійснити шляхом отримання файлу Excel із форми звіту (рис. 3.2).

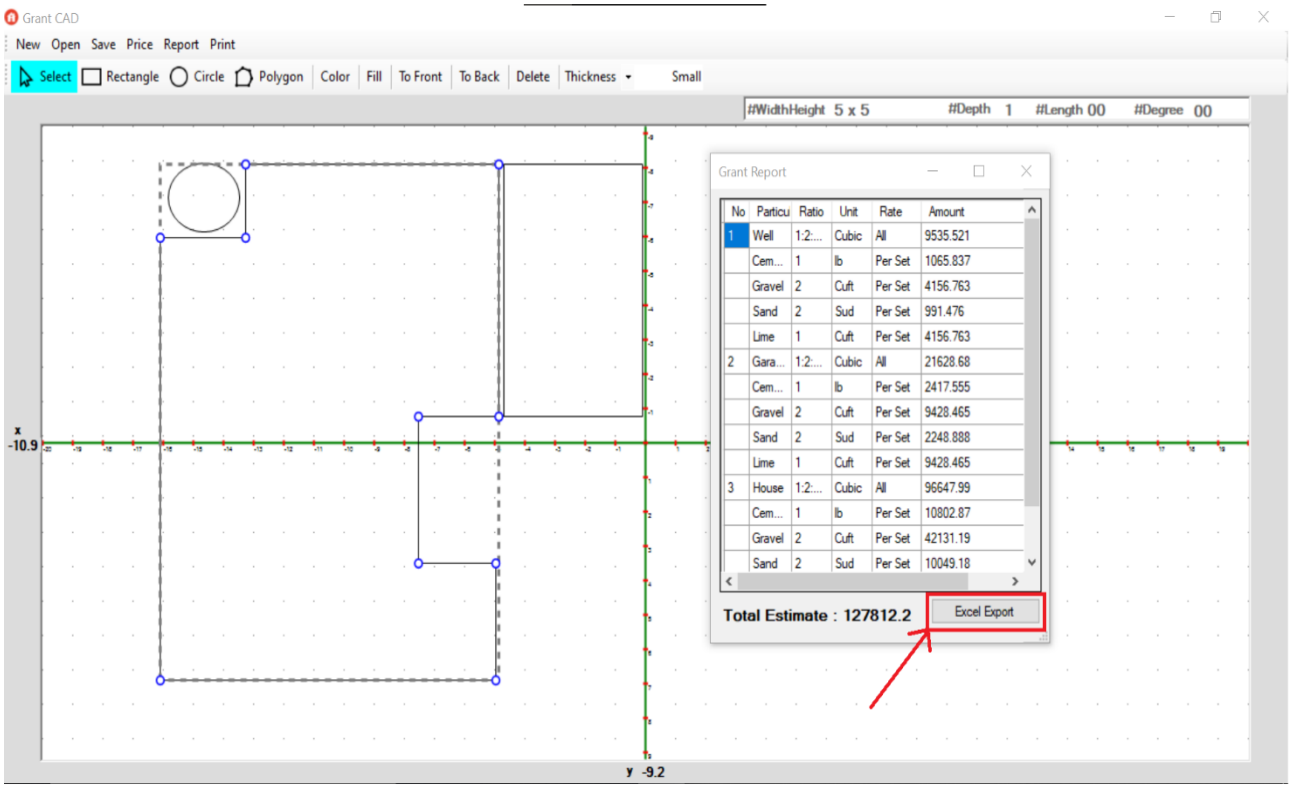

Рисунок 3.2 – Отримання розрахунків

Проведене тестування повністю підтвердило працездатність і коректність додатку.

3.4 Висновок до розділу 3

Розроблена САПР (система автоматизованого проектування) представляє собою програмне забезпечення для розрахунку витрат на будівельні матеріали. Основні елементи включають класи, що описують різні типи фігур (наприклад, прямокутники, кола, полігони) і визначають їхні характеристики. Кожен клас є шаблоном для створення відповідних об'єктів.

У коді застосовані принципи ООП, такі як інкапсуляція (використання модифікаторів доступу), наслідування (один клас успадковує властивості і методи іншого), поліморфізм (використання однаково названих методів з різними реалізаціями).

4 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічні розробки залишаються актуальними лише в тому випадку, якщо вони відповідають потребам сучасності як з точки зору науково-технічного прогресу, так і економічних аспектів. Тому важливо оцінювати ефективність результатів досліджень щодо їхньої потенційної економічної вигідності.

Магістерська робота "Інформаційна технологія розрахунку вимог на використання будівельних матеріалів" відноситься до технічних проектів, спрямованих на майбутнє введення на ринок (або вирішення щодо його можливого введення під час виконання роботи). Це передбачає комерційне використання науково-технічних розробок. Цей підхід є перспективним, оскільки результати досліджень можуть зацікавити інших користувачів, що приносить певний економічний виграш. Проте для цього необхідно знайти інвестора, який був би зацікавлений у втіленні такого проекту і переконати його в економічній доцільності цього кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково-технічної розробки;

3) розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1 [13].

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів

Продовження таблиці 4.1

Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці 4.2.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	3	4	3
2. Ринкові переваги (наявність аналогів)	3	3	3
3. Ринкові переваги (ціна продукту)	3	3	4
4. Ринкові переваги (технічні властивості)	5	4	4
5. Ринкові переваги (експлуатаційні витрати)	3	3	4
6. Ринкові перспективи (розмір ринку)	5	4	4
7. Ринкові перспективи (конкуренція)	4	3	2
8. Практична здійсненність (наявність фахівців)	5	4	5
9. Практична здійсненність (наявність фінансів)	2	5	4
10. Практична здійсненність (необхідність нових матеріалів)	4	4	2
11. Практична здійсненність (термін реалізації)	3	5	3
12. Практична здійсненність (розробка документів)	4	4	4
Сума балів	44	46	42
Середньоарифметична сума балів $СБ_с$	44		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3 [13].

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ ,розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» становить 44 балів, що, відповідно до таблиці 4.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки високий).

4.2 Розрахунок узагальненого коефіцієнта якості розробки

Окрім комерційного аудиту розробки доцільно також розглянути технічний рівень якості розробки, розглянувши її основні технічні показники. Ці показники по-різному впливають на загальну якість проектної розробки.

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення розрахуємо за формулою [14]:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i , \qquad (4.1)$$

де k – кількість найбільш важливих технічних показників, які впливають на якість нового технічного рішення;

α_i – коефіцієнт, який враховує питому вагу *і-го* технічного показника в загальній якості розробки. Коефіцієнт α_i визначається експертним шляхом і при цьому має виконуватись умова $\sum_{i=1}^{k} \alpha_i = 1$;

β_i – відносне значення *і-го* технічного показника якості нової розробки.

Відносні значення β_i для різних випадків розраховуємо за такими формулами:

- для показників, зростання яких вказує на підвищення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{ні}}{I_{аі}}, \tag{4.2}$$

де $I_{ні}$ та $I_{на}$ – чисельні значення конкретного i-го технічного показника якості відповідно для нової розробки та аналога;

для показників, зростання яких вказує на погіршення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{аі}}{I_{ні}}; \tag{4.3}$$

Використовуючи наведені залежності можемо проаналізувати та порівняти техніко-економічні характеристики аналогу та розробки на основі отриманих наявних та проектних показників, а результати порівняння зведемо до таблиці 4.4.

Таблиця 4.4 – Порівняння основних параметрів розробки та аналога.

Показники (параметри)	Одиниця вимірю-вання	Аналог	Проектований пристрій	Відношення параметрів нової розробки до аналога	Питома вага показника
Доступність (дружність) інтерфейсу	бал	5	7	1,4	0,3
Розмір на диску	Мб	80	4	20	0,1
Потреба в оперативній пам'яті	Гб	12	3	4	0,25
Швидкодія взаємозв'язку	с	1,2	0,6	2	0,1
Кількість підтримуваних баз даних	од	3	6	2	0,25

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення складе:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i = 1{,}4 \cdot 0{,}3 + 20 \cdot 0{,}1 + 4 \cdot 0{,}25 + 2 \cdot 0{,}1 + 2 * 0{,}25 = 4{,}12.$$

Отже за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 4,12 рази.

4.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

4.3.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [13]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (4.4)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=22 дні.

$$З_o = 16000{,}00 \cdot 22 / 22 = 16000{,}00 \; грн.$$

Проведені розрахунки зведемо до таблиці 4.5.

Таблиця 4.5 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	16000,00	727.27	22	16000,00
Програміст	22000,00	1000	22	22000,00
Консультант (прораб на будівництві)	10000,00	1000	10	10000,00
Лаборант	8000	363,63	22	8000,00
Всього				56/*000,00

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_p$) за відповідними найменуваннями робіт НДР на тему «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» розраховуємо за формулою:

$$З_p = \sum_{i=1}^{n} C_i \cdot t_i , \qquad (4.5)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}}, \quad (4.6)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо *Мм*=6700,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [13];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p – середнє число робочих днів в місяці, приблизно T_p = 22 дн;

$t_{зм}$ – тривалість зміни, год.

$C_1 = 6700{,}00 \cdot 1{,}10 \cdot 1{,}35 / (22 \cdot 8) = 56{,}53$ *грн.* $З_{р1} = 56{,}53 \cdot 4 = 226{,}12$ *грн.*

Проведені розрахунки зведемо до таблиці 4.6.

Таблиця 4.6 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Інсталяція програмного забезпечення розробки програмного забезпечення	4,00	3	1,10	56,53	226,12
Встановлення цифрових обчислювальних систем та мережевого доступу	2,50	3	1,35	72,35	180,875
Відлагодження програмних модулів аналітичного дослідження	3,00	5	1,60	85,35	256,05
Всього					663,045

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.7)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 10%.

$З_{дод}$ = (56000,00 + 663,045)· *10 / 100% = 5666,30грн.*

4.3.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%}, \quad (4.8)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (56000,00 + 663,045+ *5666,30) · 22 / 100% = 13712.46 грн.*

4.3.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів».

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{в\,j}, \quad (4.9)$$

де $Н_j$ – норма витрат матеріалу *j*-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$М_1 = 3,0 \cdot 225,00 \cdot 1 - 0 \cdot 0 = 675$ *грн.*

Проведені розрахунки зведемо до таблиці 4.7.

Таблиця 4.7 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір канцелярський офісний (А4) XEROX ULTRA PLUS	225,00	3,0	0	0	675
Папір для заміток (А5) Light	150,00	4,0	0	0	600
Начиння канцелярське Premium	250,00	3,0	0	0	750
Органайзер офісний	210,00	4,0	0	0	840
Картридж для принтера	2100,00	2,0	0	0	4200
Диск оптичний	25,00	5,0	0	0	125
USB-пам'ять Microtech 32 GB	135,00	2,0	0	0	270
Всього					6920

4.3.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \,, \qquad (4.10)$$

де $Н_j$ – кількість комплектуючих *j*-го виду, шт.;

$Ц_j$ – покупна ціна комплектуючих *j*-го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ *= 1,1 ... 1,15*).

$К_в = 1 \cdot 3500{,}00 \cdot 1{,}05 =$ 3675,00 *грн.*

Проведені розрахунки зведемо до таблиці 4.8.

Таблиця 4.8 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Маршрутизатор AX1800	1	3500,00	3675,00
Всього			3675,00

4.3.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \ , \qquad (4.11)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, (K_i = *1,10…1,12);*

k – кількість найменувань устаткування.

$B_{спец}$ = *7650,00 · 1 · 1,05 =* 7980*,00 грн.*

Отримані результати зведемо до таблиці 4.9:

Таблиця 4.9 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Мережеве обладнання передачі цифрових даних	1	7600,00	7980
Всього			7980

4.3.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.12)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.і}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

$K_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_і$ = *1,10…1,12);*

k – кількість найменувань програмних засобів.

$B_{прг}$ = *5000,00 · 1 · 1,05 = 5250 грн.*

Отримані результати зведемо до таблиці 4.10:

Таблиця 4.10 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Середовище програмування Visual Studio Professional	1	5000,00	5250
Всього			5250

4.3.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.13)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (30000,00 \cdot 1) / (4 \cdot 12) = 625$ *грн.*

Проведені розрахунки зведемо до таблиці 4.11.

Таблиця 4.11 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний комп'ютер Asus	30000,00	4	1	625
Робоче місце інженера-розробника (дослідника програмного забезпечення)	7000,00	5	1	116,66
Графічні пристрої виводу інформації	6600,00	5	1	110
Офісна оргтехніка	8020,00	4	1	167,71
Приміщення лабораторії розробки та дослідження	320000,00	25	1	1066
ОС Windows 10	5630,00	2	1	234,58
Прикладний пакет Microsoft Office 2022	4800,00	4	1	100
Всього				2419,95

4.3.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.14)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,05 \cdot 150,0 \cdot 7,50 \cdot 0,95 / 0,97 = 55.$

Проведені розрахунки зведемо до таблиці 4.12.

Таблиця 4.12 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Персональний комп'ютер Asus	0,05	150,0	55
Робоче місце інженера-розробника (дослідника програмного забезпечення)	0,08	150,0	90,00
Графічні пристрої виводу інформації	0,12	2,0	1,80
Офісна оргтехніка	0,32	2,0	4,80
Всього			151.6

4.3.9 Службові відрядження

Витрати за статтею «Службові відрядження» відсутні.

4.3.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» відсутні.

4.3.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.15)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів} = 55\%$.

4.3.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.16)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$В_{нзв}$ = (56000,00 + 663,045) · 100 / 100% = 56663,45 грн.

Витрати на проведення науково-дослідної роботи на тему «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.17)$$

$В_{заг}$= 56000,00 + 663.45 + 5666,30 + 13712.46 + 6920 + 3675 + 7980 + 5250 + 2419,95 + 151,6 + 31 164,67 = 133 605.18 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (4.18)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,95.

ЗВ = 133 605.18 / 0,95 = 140637.03 грн.

4.4 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» передбачають комерціалізацію протягом 4-х років реалізації на ринку (таблиця 4.13). Робота ідентифікується як «*Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем».*

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

ΔN – збільшення кількості споживачів продукту, у періоди часу, що аналізуються, від покращення його певних характеристик;

Таблиця 4.13 – Комерціалізацію протягом 4-х років

Показник	1-й рік	2-й рік	3-й рік	4-й рік
Збільшення кількості споживачів, осіб	1000	1300	1400	1050

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 1000 осіб;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 350,00 грн;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 50 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [13]:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.19)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2023 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо ρ =50%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році $\vartheta = 18\%$

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (50 \cdot 1000,00 + 350 \cdot 1000) \cdot 0,83 \cdot 0,5 \cdot (1\text{-}0,18/100\%) = 119\ 105.00$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (50 \cdot 1000,00 + 350 \cdot 2300) \cdot 0,83 \cdot 0,5 \cdot (1\text{-}0,18/100\%) = 290\ 959.50$ грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (50 \cdot 1000,00 + 350 \cdot 3700) \cdot 0,83 \cdot 0,5 \cdot (1\text{-}0,18/100\%) = 457\ 703.50$ грн.

Збільшення чистого прибутку 4-го року:

$\Delta П_4 = (50 \cdot 1000,00 + 350 \cdot 4750) \cdot 0,83 \cdot 0,5 \cdot (1\text{-}0,18/100\%) = 582\ 763.75$ грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.20)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,22;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 119105.00/(1+0,22)^1+290959.50/(1+0,22)^2+457703.50/(1+0,22)^3+ 582763.75/ /(1+0,22)^4=97627,05 + 195484,40 + 252060,20 + 263058.30=$$

808230,30 грн.

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \quad (4.21)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$ =1.5;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо *140637.03 грн.*

$PV = k_{інв} \cdot ЗВ = 1.5 \cdot 140637.03 = 210\,955{,}55$ грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV, \tag{4.22}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 808230,30 грн;

PV – теперішня вартість початкових інвестицій, 210 955,55 грн.

$E_{абс} = ПП - PV = 808230{,}30 - 210\,955{,}55 = 597\,274{,}75$ грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{4.23}$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 808230,30;

PV – теперішня вартість початкових інвестицій, 210 955,55 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1 = (1 + 597\,274{,}75 / 210\,955{,}55)^{1/4} - 1 = 0.399.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \tag{4.24}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,2.

$\tau_{мін}$ = 0,1+0,2 = 0,3 < 0,399 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{4.25}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 0,399 = 2,506 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

4.5 Висновок до розділу 4

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів» становить 44 бала, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки високий).

При оцінюванні за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 4,12 рази.

Також термін окупності становить 2,506 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційна технологія розрахунку вимог на використання будівельних матеріалів».

ВИСНОВКИ

В ході аналізу предметної області та вивчення існуючих реалізацій САПР для розрахунку будівельних матеріалів виявлено ключові вимоги та функціональність, які повинен надавати розроблюваний додаток. Окреслено основні принципи організації та взаємодії з користувачем.

Було розроблено алгоритм роботи модуля розрахунку вимог на використання будівельних матеріалів. Представлено UML-діаграму, що ілюструє взаємодію компонентів модуля.

Також було розглянуто процес створення модулю розрахунку для врахування вимог до будівельних матеріалів. Було розглянуто важливі аспекти, такі як точність розрахунків та швидкість виконання операцій. За допомогою правильно розробленого алгоритму, САПР може ефективно виконувати розрахунки та забезпечувати точність результатів.

Проведено перевірку розробленої САПР за визначеними критеріями. Були враховані такі аспекти, як функціональність, надійність, ефективність та інші.

Проведено дослідження, в якому комерційний потенціал розробки оцінюється на високому рівні, досягаючи 44 бала. Технічно розробка виявилася значно переважною над існуючими аналогами, маючи узагальнений коефіцієнт якості приблизно в 4,12 рази. Термін окупності цієї розробки становить 2,506 р., що відповідає вимогам комерційної привабливості і може зацікавити потенційних інвесторів.

Всі завдання магістерської кваліфікаційної роботи виконані, мети магістерської кваліфікаційної роботи виконані повністю досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Інформаційна технологія розрахунку вимог на використання будівельних матеріалів / А.С. Сіваєв, С.І. Петришин, – Тези LII науково-технічної конференції підрозділів ВНТУ (НТКП ВНТУ-2023). [Електронний ресурс]. Режим доступу – https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2023/paper/view/18930/15693.

2. Bidarra, R.; Bronsvoort, W. F.: Семантичне моделювання фігур, Комп'ютерне проектування, 32(3), 2000, 201– 25. http://doi.org/10.1016/S0010-4485(99)00090-1

3. Bonacich, P.: Влада і центральність: родина вимірів, Американський соціологічний журнал, 92(5), 1987, 1170–1182. http://doi.org/10.1086/228631

4. Bonacich, P.; Lloyd, P.: Міри центральності типу власного вектора для асиметричних відносин, Соціальні мережі, 23 (3), 2001, 191–201. http://doi.org/10.1016/S0378-8733(01)00038-7

5. Borgatti, S. P.; Everett, M. G.: Граф-теоретична перспектива на центральність, Соціальні мережі, 28(4), 2006, 466–84. http://doi.org/10.1016/j.socnet.2005.11.005

6. Camba, J. D.; Contero, M.; Company, P.: Параметричне моделювання САПР: аналіз стратегій для повторного використання дизайну, Комп'ютерне проектування, 74, 2016, 18–31. http://doi.org/10.1016/j.cad.2016.01.003

7. Cheng, H.; Chu, X.: Підхід до оцінки впливу змін на складний продукт на основі мережі, Журнал інтелектуального виробництва, 23, 2012, 1419–1431. http://doi.org/10.1007/s10845-010- 0454-8

8. Cheng, Z.-R.; Ma, Y.-S.: Мережева оцінка залежності дизайну у розробці продукту, У зб. 2014 Міжнар. конф. Інновації в дизайні та виробництві, 2014, 137–142. Монреаль, Квебек, Канада. http://doi.org/10.1109/IDAM.2014.6912684

9. Cheng, Z.-R.; Ma, Y.-S.: Метод моделювання функціональних ознак, Розширена інженерна інформатика, 33, 2017, 1–15. https://doi.org/10.1016/j.aei.2017.04.003

10. Easley, D.; Kleinberg, J.: Мережі, Толпи та Ринки: Мислення про високозв'язаний світ, Кембриджський університетський прес, Нью-Йорк, США, 2010

11. Freeman, L. C.: Набір вимірів центральності на основі посередництва, Соціометрія, 40(1), 1977, 35– 41. http://doi.org/10.2307/3033543

12. He, K.-J.; Chen, Z.-M.; Jiang, J.-F.; Wang, L.: Створення користувацьких вільних форм з поверхневих моделей на основі характерних кривих, Комп'ютерна промисловість 65(4), 2014, 598–609. https://doi.org/10.1016/j.compind.2014.01.011

13. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

14. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

15. Hoffmann, C.- M.; Robert, J.-A.: Про користувацькі визначені ознаки, Комп'ютерне проектування, 30(5), 1998, 321– 32. doi:10.1016/S0010-4485(97)00048-1.

16. Jaiswal, P.; Huang, J.; Rai, R.: Концептуальне 3D моделювання на основі збору з ймовірнісним графом, Комп'ютерне проектування, 74, 2014, 45–54. http://doi.org/10.1016/j.cad.2015.10.002

17. Katz, L.: Новий індекс статусу, отриманий з соціометричного індексу, Психометрика, 18(1), 1953, 39-43. http://doi.org/10.1007/BF02289026

18. Le, Q.; Sha, Z.; Panchal, J.-H: Генеративна модель мережі для еволюції продукту, Ж. обчислення та інформаційних наук. інж., 14(1), 2014, 11003. http://doi.org/10.1115/1.4025856

19. Li, L.; Lange, C.-F.; Ma, Y.-M.: Асоціація дизайну та обчислювальної інтентування в оптимізації продукту керування потоками, Звіти Інституту механічних інженерів, Частина B: Журнал інженерної виробництва. доступно онлайн, 2017, doi:10.1177/0954405417697352.

20. Li, M.; Zhang, Y.-F.; Fuh, J. Y. H.; Qiu, Z. M.: До ефективного повторного використання механічного дизайну: пошук CAD-моделей на основі загальних та часткових форм, Журнал механічного дизайну, 131(12), 2009, 124501. http://doi.org/10.1115/1.4000253

21. Ma, Y.-S.; Tong, T.: Асоціативне моделювання функціональних ознак для інтеграції конкурентного інжинірингу, Комп'ютери в промисловості, 51(1), 2003, 51–71. http://doi.org/10.1016/S0166-3615(03)00025-3

22. MacCormack, A.; Rusnak, J.; Baldwin, C.-Y.: Вивчення структури складних програмних дизайнів: емпіричне дослідження відкритого та пропрієтарного коду, Управлінська наука, 52(7), 2006, 1015–1030.

Додаток А (обов'язковий)

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія розрахунку вимог на використання будівельних матеріалів

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 96,77% Схожість 3,23%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Сіваєв А.С.

Керівник роботи _____ Петришин С.І.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку _____ Озеранський В.С.

Додаток Б (обов'язковий)

Лістинг програми

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace GrantCalculator
{
    public static class Calculate
    {

        // Calculate the distance squared between two points.
        public static float FindDistanceToPointSquared(PointF pt1,
PointF pt2)
        {
            float dx = pt1.X - pt2.X;
            float dy = pt1.Y - pt2.Y;
            return dx * dx + dy * dy;
        }
        public static double Differentiate(float p1, float p2)
        {
            double diff = p2 - p1;
            return diff;
        }
        public static float LengthBetweenTwopoints(PointF pt, PointF
pts)
        {
            float num = pt.X - pts.X;
            float num2 = pt.Y - pts.Y;
            float value = num * num + num2 * num2;
```

```
            double num3 = Math.Sqrt(value);
            return (float)num3;
        }
        public static float AngelBetweenTwopoints(PointF pt1, PointF
pt2)
        {
            double dx = Differentiate(pt1.X , pt2.X);
            double dy = Differentiate(pt1.Y , pt2.Y);
            double num3 = Math.Atan2(dy, dx) * 57.295779513082323;
            return (float)num3;
        }
        public static float RectangleArea(float width, float height)
        {
            return width * height;
        }
        // Return the polygon's area in "square units."
        // The value will be negative if the polygon is
        // oriented clockwise.
        public static float PolygonArea(PointF[] ptfs)
        {
            // Add the first point to the end.
            int num_points = ptfs.Length;
            PointF[] pts = new PointF[num_points + 1];
            ptfs.CopyTo(pts, 0);
            pts[num_points] = ptfs[0];

            // Get the areas.
            float area = 0;
            for (int i = 0; i < num_points; i++)
            {
                area +=
                    (pts[i + 1].X - pts[i].X) *
                    (pts[i + 1].Y + pts[i].Y) / 2;
            }
```

```
            // Return the result.
            return area;
        }
        public static float EllipseArea(float width, float height)
        {
            //width - semi major axis
            //height - semi minor axis
            float Area;

            Area = (float)3.142 * width * height;

            return Area;

        }
        public static int ShapeRatio(Shape shape)
        {
            return shape.Cement + shape.Gravel + shape.Sand +
shape.Lime;
        }
        public static float CubicFeet(float area,float depth)
        {
            return area * depth;
        }
        public static float AverageCubic(int particalRatio,int
ratio,float cubicfeet)
        {
            return (cubicfeet / ratio) * particalRatio;
        }
        public static float MaterialPrice(float material,float
price)
        {
            return material * price;
        }
```

```
      }
    }
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Printing;
using System.IO;
using System.Linq;
using System.Runtime.Serialization.Formatters.Binary;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GrantCalculator
{
    public partial class Mainform : Form
    {
        public enum Drawtools
        {
            _select, _rectangle, _circle, _polygon

        }
        Drawtools toolis;
        readonly string RECTSHAPE = "REC";
        readonly string POLSHAPE = "POL";
        readonly string CIRSHAPE = "CIR";
        bool _mouseDown = false;
        PointF sPt, ePt;
        Pen curPen;

        // Store the selected shape.
```

```
private Shape selectedShape;
// Keep track of when drag or resize mode is enabled.
private bool isDragging = false;
private bool isResizing = false;
private bool isMovingpoint = false;
private Shape.HitSpot resizingMode;

// Store the location where the user clicked on the control.
private float clickOffsetX, clickOffsetY;
Color myColor, ourColor;
Shape drawShape = null;
public float lineThick = .01F;
RectangleF myRect, myCirc;
List<PointF> myPolygons;
private bool reqStraightLine;
RectanglePlotter plotRect;
private ShapesCollection shapes = new ShapesCollection();
float abovedepth = 1;
public Mainform()
{
    InitializeComponent();

    MakeBackgroundGrid();
    CreateTransforms();

    if (!File.Exists(DataAccess.fPath))
    {
        DataAccess.CreateEstimationXML();
    }
    DataAccess.ParseEstimationXML();

}
private void Mainform_Load(object sender, EventArgs e)
{
```

```
                myColor = Color.Black;
                ourColor = Color.Empty;
                toolStripComboBoxlinethickness.SelectedIndex = 0;
            }
            #region xyPlaneVari
            // The transformations.
            private Matrix Transform = null, InverseTransform = null;
            /*The Transform and InverseTransform variables are the matrices used to transform the drawing and to find
            * the inverse points for mouse coordinates.*/
            private const float DrawingScale = 30F;
            // The world coordinate bounds.
            private float Wxmin, Wxmax, Wymin, Wymax;
            /* Wxmin, Wxmax, Wymin, and Wymax store the world coordinates used to draw the ellipses.*/
            private const int GridGap = 30;
            #endregion
            #region xyPlaneFunc
            private void MakeBackgroundGrid()
            {
                Bitmap bm = new Bitmap(picbox_drawhere.ClientSize.Width, picbox_drawhere.ClientSize.Height);
                for (int x = 0; x < picbox_drawhere.ClientSize.Width; x += GridGap)
                {
                    for (int y = 0; y < picbox_drawhere.ClientSize.Height; y += GridGap)
                    {
                        bm.SetPixel(x, y, Color.Gray);
                    }
                }

                picbox_drawhere.BackgroundImage = bm;
            }
```

```
private void CreateTransforms()
{
    // Make the draw transformation. (World --> Device)
    Transform = new Matrix();
    Transform.Scale(DrawingScale, DrawingScale);
    float cx = picbox_drawhere.ClientSize.Width / 2;
    float cy = picbox_drawhere.ClientSize.Height / 2;
    Transform.Translate(cx, cy, MatrixOrder.Append);

    // Make the inverse transformation. (Device --> World)
    InverseTransform = Transform.Clone();
    InverseTransform.Invert();

    // Calculate the world coordinate bounds.
    Wxmin = -cx / DrawingScale;
    Wxmax = cx / DrawingScale;
    Wymin = -cy / DrawingScale;
    Wymax = cy / DrawingScale;
}



// Convert from device coordinates to world coordinates.
private PointF DeviceToWorld(PointF point)
{
    PointF[] points = { point };
    InverseTransform.TransformPoints(points);
    return points[0];
}
private void Form1_Resize(object sender, EventArgs e)
{

}
#endregion
```

```
            #region PIC_MOUSE_EVE
            int hit_point;
            List<PointF> HitPolygon;
            private    void    picbox_drawhere_MouseUp(object    sender,
MouseEventArgs e)
            {
                ePt = DeviceToWorld(e.Location);
                _mouseDown = false;
                if (toolis == Drawtools._select)
                {
                    picbox_drawhere.Cursor = Cursors.Arrow;
                    isDragging = false;
                    isResizing = false;
                    isMovingpoint = false;

                }
                if (!myRect.IsEmpty && toolis == Drawtools._rectangle)
                {
                    myRect = plotRect.ShapeRect(sPt, ePt);
                    drawShape = new RectangleShape
                    {
                        ShapeType = RECTSHAPE,
                        Location = myRect.Location,
                        Size = myRect.Size,
                        ForeColor = myColor,
                        BackColor = ourColor,
                        PenThickness = lineThick,
                        Name = "NoName",
                        Cement = DataAccess.estimator.Cement,
                        Gravel = DataAccess.estimator.Gravel,
                        Sand = DataAccess.estimator.Sand,
                        Lime = DataAccess.estimator.Lime,
                        Depth = abovedepth,
```

```
                        Area   =   Calculate.RectangleArea(myRect.Width,
myRect.Height)
                    };
                    shapes.Add(drawShape);
                    myRect = RectangleF.Empty;
                    picbox_drawhere.Refresh();

                }
                if (!myCirc.IsEmpty && toolis == Drawtools._circle)
                {
                    myCirc = plotRect.ShapeRect(sPt, ePt);
                    drawShape = new CircleShape
                    {
                        ShapeType = CIRSHAPE,
                        Location = myCirc.Location,
                        Size = myCirc.Size,
                        ForeColor = myColor,
                        BackColor = ourColor,
                        PenThickness = lineThick,
                        Name = "NoName",
                        Cement = DataAccess.estimator.Cement,
                        Gravel = DataAccess.estimator.Gravel,
                        Sand = DataAccess.estimator.Sand,
                        Lime = DataAccess.estimator.Lime,
                        Depth = abovedepth,
                        Area    =    Calculate.EllipseArea(myCirc.Width,
myCirc.Height)
                    };
                    shapes.Add(drawShape);
                    myCirc = RectangleF.Empty;
                    picbox_drawhere.Refresh();


                }
```

```
            }
            private   void   picbox_drawhere_MouseMove(object   sender,
MouseEventArgs e)
            {
                ePt = DeviceToWorld(e.Location);

                if (toolis == Drawtools._select)
                {
                    if (isDragging)
                    {
                        picbox_drawhere.Cursor = Cursors.SizeAll;
                        if  (selectedShape.ShapeType  ==  RECTSHAPE  ||
selectedShape.ShapeType == CIRSHAPE)
                            selectedShape.Location = new PointF(ePt.X -
clickOffsetX, ePt.Y - clickOffsetY);

                        else if (selectedShape.ShapeType == POLSHAPE)
                        {
                            // See how far the first point will move.
                            float new_x1 = ePt.X + clickOffsetX;
                            float new_y1 = ePt.Y + clickOffsetY;

                            float dx = new_x1 - HitPolygon[0].X;
                            float dy = new_y1 - HitPolygon[0].Y;
                            if (dx == 0 && dy == 0) return;
                            for (int i = 0; i < HitPolygon.Count; i++)
                            {
                                HitPolygon[i] = new PointF(
                                    HitPolygon[i].X + dx,
                                    HitPolygon[i].Y + dy);
                            }
```

```
                    selectedShape.Locations = HitPolygon;

                }

                picbox_drawhere.Invalidate();

            }
            else if (isResizing)
            {
                int minSize = 1;

                // Keep track of the old size. Useful
                // for invalidating when NOT double-buffering.
                //RectangleF            oldPosition            =
selectedShape.GetLargestPossibleRegion();

                // Resize the control, according to the resize
mode.
                switch (resizingMode)
                {
                    case Shape.HitSpot.Top:
                    case Shape.HitSpot.TopRightCorner:
                        if (ePt.Y < (selectedShape.Location.Y +
selectedShape.Size.Height - minSize))
                        {
                            selectedShape.Size       =       new
SizeF(selectedShape.Size.Width,
                                selectedShape.Location.Y      +
selectedShape.Size.Height - (ePt.Y - clickOffsetY));
                            selectedShape.Location     =     new
PointF(selectedShape.Location.X, ePt.Y - clickOffsetY);
                        }
                        break;
                    case Shape.HitSpot.Bottom:
```

```
                                    if (ePt.Y > (selectedShape.Location.Y +
minSize))
                                    {
                                        selectedShape.Size       =       new
SizeF(selectedShape.Size.Width, ePt.Y - selectedShape.Location.Y);
                                    }
                                    break;
                                case Shape.HitSpot.Left:
                                case Shape.HitSpot.BottomLeftCorner:
                                case Shape.HitSpot.TopLeftCorner:
                                    if (ePt.X < (selectedShape.Location.X +
selectedShape.Size.Width - minSize))
                                    {
                                        selectedShape.Size       =       new
SizeF((selectedShape.Location.X  +  selectedShape.Size.Width)  -  (ePt.X  -
clickOffsetX), selectedShape.Size.Height);
                                        selectedShape.Location     =     new
PointF(ePt.X - clickOffsetX, selectedShape.Location.Y);
                                    }
                                    break;
                                case Shape.HitSpot.Right:
                                    if (ePt.X > (selectedShape.Location.X +
minSize))
                                    {
                                        selectedShape.Size = new SizeF(ePt.X
- selectedShape.Location.X, selectedShape.Size.Height);
                                    }
                                    break;
                                case Shape.HitSpot.BottomRightCorner:
                                    if (ePt.Y > (selectedShape.Location.Y +
minSize))
                                    {
                                        selectedShape.Size       =       new
SizeF(selectedShape.Size.Width, ePt.Y - selectedShape.Location.Y);
```

```
                                }
                                if (ePt.X > (selectedShape.Location.X +
minSize))
                                {
                                    selectedShape.Size = new SizeF(ePt.X
- selectedShape.Location.X, selectedShape.Size.Height);
                                }
                                break;
                        }
                        if (selectedShape.ShapeType == RECTSHAPE)
                            selectedShape.Area                          =
Calculate.RectangleArea(selectedShape.Size.Width,
selectedShape.Size.Height);
                        else if (selectedShape.ShapeType == CIRSHAPE)
                            selectedShape.Area                          =
Calculate.EllipseArea(selectedShape.Size.Width,
selectedShape.Size.Height);
                        picbox_drawhere.Refresh();
                    }
                    else if (isMovingpoint && _mouseDown)
                    {
                        if (selectedShape != null)
                        {
                            HitPolygon[hit_point] = ePt;
                            selectedShape.Locations = HitPolygon;
                            selectedShape.Area                          =
Math.Abs(Calculate.PolygonArea(HitPolygon.ToArray()));

                        }
                        picbox_drawhere.Refresh();

                    }
                    else
                    {
```

```
                        if     ((selectedShape     !=     null)     &&
(selectedShape.Selected))
                        {
                            if (selectedShape.ShapeType == RECTSHAPE ||
selectedShape.ShapeType == CIRSHAPE)
                            {
                                if
(selectedShape.HitTestFocusBorder(ePt, out resizingMode))
                                {
                                    switch (resizingMode)
                                    {
                                        case Shape.HitSpot.Top:
                                        case Shape.HitSpot.Bottom:
                                        case
Shape.HitSpot.TopRightCorner:
                                            picbox_drawhere.Cursor    =
Cursors.SizeNS;
                                            break;
                                        case Shape.HitSpot.Left:
                                        case Shape.HitSpot.Right:
                                        case
Shape.HitSpot.BottomLeftCorner:
                                        case
Shape.HitSpot.TopLeftCorner:
                                            picbox_drawhere.Cursor    =
Cursors.SizeWE;
                                            break;
                                        case
Shape.HitSpot.BottomRightCorner:
                                            picbox_drawhere.Cursor    =
Cursors.SizeNWSE;
                                            break;
                                        default:
```

```
                                        picbox_drawhere.Cursor     =
Cursors.Arrow;
                                        break;
                                    }
                                }
                                else
                                {
                                    picbox_drawhere.Cursor             =
Cursors.Arrow;
                                }

                            }
                            else    if    (selectedShape.ShapeType    ==
POLSHAPE)
                            {
                                if
(selectedShape.MouseIsOverCornerPoint(ePt,  selectedShape.Locations,  out
hit_point))
                                {
                                    picbox_drawhere.Cursor             =
Cursors.Hand;
                                    isMovingpoint = true;

                                }
                                else
                                {
                                    picbox_drawhere.Cursor             =
Cursors.Arrow;
                                    isMovingpoint = false;
                                }
                            }
                        }
                        else
                        {
```

```
                    picbox_drawhere.Cursor = Cursors.Arrow;
                }

            }


        }
        if (_mouseDown)
        {

            if (toolis == Drawtools._rectangle)
            {
                ClearSelectedShape();
                myRect = plotRect.ShapeRect(sPt, ePt);
                picbox_drawhere.Refresh();
            }

            if (toolis == Drawtools._circle)
            {
                ClearSelectedShape();
                myCirc = plotRect.ShapeRect(sPt, ePt);
                picbox_drawhere.Refresh();
            }
        }
        if (toolis == Drawtools._polygon)
        {

            if (reqStraightLine)
            {
                float                   deg                   =
Math.Abs(Calculate.AngelBetweenTwopoints(ePt, sPt));

                if (deg >= 0 && deg <= 45) //Hoz
                {
```

```
                    ePt = new PointF(ePt.X, sPt.Y);
                }
                else if (deg >= 45 && deg <= 90) //Ver
                {
                    ePt = new PointF(sPt.X, ePt.Y);
                }
                else if (deg >= 90 && deg <= 135)//Hoz
                {
                    ePt = new PointF(sPt.X, ePt.Y);
                }
                else if (deg >= 135 && deg <= 180)//Ver
                {
                    ePt = new PointF(ePt.X, sPt.Y);
                }


            }

        }
        if (myPolygons != null && myPolygons.Count > 0)
        {
            picbox_drawhere.Refresh();
        }
        Set_Shape_Status();
    }
    private   void   picbox_drawhere_MouseDown(object   sender,
MouseEventArgs e)
    {
        sPt = DeviceToWorld(e.Location);
        _mouseDown = true;

        if (toolis == Drawtools._select)
        {
            Shape.HitSpot hitSpot;
```

```
                    if      ((selectedShape      !=      null)      &&
(selectedShape.Selected) &&
                     (selectedShape.HitTestFocusBorder(new
PointF(sPt.X, sPt.Y), out hitSpot)))
                    {
                        // The border was clicked. Turn on resize mode.
                        clickOffsetX = sPt.X - selectedShape.Location.X;
                        clickOffsetY = sPt.Y - selectedShape.Location.Y;
                        isResizing = true;
                    }
                    else
                    {
                        // Retrieve a reference to the selected shape
                        // using hit testing.
                        ClearSelectedShape();
                        selectedShape = shapes.HitTest(sPt);
                        if (selectedShape != null)
                        {
                            selectedShape.Selected = true;
                            picbox_drawhere.Refresh();

                            if (e.Button == MouseButtons.Left)
                            {

                                // Start dragging mode.
                                if (selectedShape.ShapeType == POLSHAPE)
                                {
                                    HitPolygon                          =
selectedShape.Locations;
                                    clickOffsetX  =  HitPolygon[0].X  -
sPt.X;
                                    clickOffsetY  =  HitPolygon[0].Y  -
sPt.Y;
                                }
```

```
                                else
                                {
                                    clickOffsetX = sPt.X -
selectedShape.Location.X;
                                    clickOffsetY = sPt.Y -
selectedShape.Location.Y;
                                }
                                if (!isMovingpoint)
                                {
                                    picbox_drawhere.Cursor =
Cursors.SizeAll;
                                    isDragging = true;
                                }

                            }
                        }

                    }


                }
                if (toolis == Drawtools._rectangle)
                {
                    ClearSelectedShape();
                    curPen = new Pen(myColor, lineThick);
                    plotRect = new RectanglePlotter();
                }
                if (toolis == Drawtools._polygon)
                {
                    ClearSelectedShape();
                    if (myPolygons == null)
                        myPolygons = new List<PointF>();
                    curPen = new Pen(myColor, lineThick);
                    if (reqStraightLine)
```

```
                myPolygons.Add(ePt);
            else
                myPolygons.Add(sPt);
            picbox_drawhere.Refresh();

        }
        if (toolis == Drawtools._circle)
        {
            ClearSelectedShape();
            curPen = new Pen(myColor, lineThick);
            plotRect = new RectanglePlotter();
        }
        if (e.Button == MouseButtons.Right)
        {
            if (toolis == Drawtools._select)
            {
                if (selectedShape != null)
                {
                    mnuselectshape.Show(this,   new   Point(e.X,
e.Y));
                }
            }
            if (toolis == Drawtools._polygon)
            {
                if (myPolygons.Count > 2)
                {
                    drawShape = new PolygonShape()
                    {

                        Locations = myPolygons,
                        ShapeType = POLSHAPE,
                        BackColor = ourColor,
                        ForeColor = myColor,
                        PenThickness = lineThick,
```

```
                                Name = "NoName",
                                Cement = DataAccess.estimator.Cement,
                                Gravel = DataAccess.estimator.Gravel,
                                Sand = DataAccess.estimator.Sand,
                                Lime = DataAccess.estimator.Lime,
                                Depth = abovedepth,
                                Area                                    =
Math.Abs(Calculate.PolygonArea(myPolygons.ToArray()))
                            };
                            shapes.Add(drawShape);
                            myPolygons = null;
                        }

                    }

                }
                Set_Shape_Status();
            }
            #endregion
            #region ToolStrips
            void Color_ToolsButtonHighLight(ToolStripButton setbutton)
            {
                toolStripButtonselect.BackColor = Color.Empty;
                toolStripButtonCircle.BackColor = Color.Empty;
                toolStripButtonpolygon.BackColor = Color.Empty;
                toolStripButtonRect.BackColor = Color.Empty;
                setbutton.BackColor = Color.Aqua;
            }


            private   void   toolStripButtonRect_Click(object   sender,
EventArgs e)
            {
                toolis = Drawtools._rectangle;
```

```
            picbox_drawhere.Cursor = Cursors.Cross;
            Color_ToolsButtonHighLight(toolStripButtonRect);
        }

        private void toolStripButtonColor_Click(object sender,
EventArgs e)
        {
            ColorDialog dlgColor = new ColorDialog();
            if (dlgColor.ShowDialog() == DialogResult.OK)
            {

                myColor = dlgColor.Color;
                if (selectedShape != null)
                    selectedShape.ForeColor = myColor;
                picbox_drawhere.Refresh();
            }
        }

        private void toolStripButtondelete_Click(object sender,
EventArgs e)
        {
            if (selectedShape == null) return;
            shapes.Remove(selectedShape);
            ClearSelectedShape();
        }

        private void toolStripButtontoback_Click(object sender,
EventArgs e)
        {
            if (selectedShape == null) return;
            shapes.SendShapeToBack(selectedShape);
            picbox_drawhere.Refresh();
        }
```

```
            private void toolStripButtontofront_Click(object sender,
EventArgs e)
            {
                if (selectedShape == null) return;
                shapes.BringShapeToFront(selectedShape);
                picbox_drawhere.Refresh();
            }

            private void toolStripButtonSave_Click(object sender,
EventArgs e)
            {
                saveFileDialog.Filter = "Grant files (*.gcc)|*.gcc|All
files (*.*)|*.*";
                if (saveFileDialog.ShowDialog() == DialogResult.OK)
                {
                    try
                    {
                        using (FileStream fs =
                            File.Create(saveFileDialog.FileName))
                        {
                            BinaryFormatter f = new BinaryFormatter();
                            f.Serialize(fs, shapes);
                        }
                    }
                    catch (Exception err)
                    {
                        MessageBox.Show("Error while saving. " +
err.Message);
                    }
                }
            }

            private void toolStripButtonOpen_Click(object sender,
EventArgs e)
```

```
            {
                openFileDialog.Filter = "Grant files (*.gcc)|*.gcc";
                if (openFileDialog.ShowDialog() == DialogResult.OK)
                {
                    ShapesCollection newShapes = null;
                    try
                    {
                        using (FileStream fs =
                            File.Open(openFileDialog.FileName,
FileMode.Open))
                        {
                            BinaryFormatter f = new BinaryFormatter();
                            newShapes                                    =
(ShapesCollection)f.Deserialize(fs, null);
                        }
                    }
                    catch (Exception err)
                    {
                        MessageBox.Show("Error   while   loading.   "   +
err.Message);
                    }

                    // Trigger a refresh.
                    shapes = newShapes;
                    picbox_drawhere.Refresh();
                }
            }

            private   void   toolStripButtonNew_Click(object   sender,
EventArgs e)
            {
                if (shapes.Count > 0)
                {
```

```
                    var confirm = MessageBox.Show("Are you sure to clear
all?", "Caution!", MessageBoxButtons.YesNo);
                    if (confirm == DialogResult.Yes)
                    {
                        shapes.Clear();
                        picbox_drawhere.Refresh();
                    }

                }

            }

            private void fillColorToolStripMenuItem_Click(object sender,
EventArgs e)
            {
                if (selectedShape == null)
                    return;
                ColorDialog dlgColor = new ColorDialog();
                if (dlgColor.ShowDialog() == DialogResult.OK)
                {
                    selectedShape.BackColor = dlgColor.Color;
                    picbox_drawhere.Refresh();
                }
            }

            private    void    clearColorToolStripMenuItem_Click(object
sender, EventArgs e)
            {
                if (selectedShape == null)
                    return;
                selectedShape.BackColor = Color.Empty;
                picbox_drawhere.Refresh();
            }
```

```
            private   void   bringToFrontToolStripMenuItem_Click(object
sender, EventArgs e)
            {
                if (selectedShape == null) return;
                shapes.BringShapeToFront(selectedShape);
                picbox_drawhere.Refresh();
            }

            private    void    sendToBackToolStripMenuItem_Click(object
sender, EventArgs e)
            {
                if (selectedShape == null) return;
                shapes.SendShapeToBack(selectedShape);
                picbox_drawhere.Refresh();
            }

            private  void  deleteToolStripMenuItem_Click(object  sender,
EventArgs e)
            {
                if (selectedShape == null) return;
                shapes.Remove(selectedShape);
                ClearSelectedShape();
            }

            private void estimateToolStripMenuItem_Click(object sender,
EventArgs e)
            {
                if (selectedShape != null)
                {

                    Estimationform          frm          =          new
Estimationform(selectedShape);
                    frm.ShowDialog();
                }
```

```
        }

        private                                                 void
toolStripComboBoxlinethickness_SelectedIndexChanged(object      sender,
EventArgs e)
        {
            if (toolStripComboBoxlinethickness.SelectedIndex == 0)
            {
                lineThick = 0.01F;
            }
            else if (toolStripComboBoxlinethickness.SelectedIndex ==
1)
            {
                lineThick = 0.1F;
            }
            else if (toolStripComboBoxlinethickness.SelectedIndex ==
2)
            {
                lineThick = 1F;
            }
            SendKeys.Send("{ESC}");
            this.ActiveControl = null;



        }
        private   void   toolStripButtonprice_Click(object   sender,
EventArgs e)
        {
            PricesetForm psf = new PricesetForm();
            psf.ShowDialog();
        }
```

```
            private   void   setDeptAboveToolStripMenuItem_Click(object
sender, EventArgs e)
            {
                DepthForm df = new DepthForm(abovedepth);
                if (df.ShowDialog() == DialogResult.OK)
                {
                    abovedepth = (float)df.numericUpDownZDepth.Value;
                    if (selectedShape != null)
                        selectedShape.Depth = abovedepth;
                }
            }

            private  void  toolStripButtonfillcolor_Click(object  sender,
EventArgs e)
            {
                ColorDialog dlgColor = new ColorDialog();
                if (dlgColor.ShowDialog() == DialogResult.OK)
                {

                    ourColor = dlgColor.Color;
                    if (selectedShape != null)
                        selectedShape.BackColor = ourColor;
                    picbox_drawhere.Refresh();
                }
            }

            private void toolStripButtongrantreport_Click(object sender,
EventArgs e)
            {
                if (shapes.Count > 0)
                {
                    ReportForm reportForm = new ReportForm(shapes);
                    reportForm.ShowDialog();
                }
```

```
        }

        private void toolStripButtonprint_Click(object sender,
EventArgs e)
        {
            if (shapes.Count > 0)
            {
                ClearSelectedShape();
                printPreview.ShowDialog();
            }
        }
        private void toolStripButtonselect_Click(object sender,
EventArgs e)
        {
            toolis = Drawtools._select;
            picbox_drawhere.Cursor = Cursors.Arrow;
            Color_ToolsButtonHighLight(toolStripButtonselect);
        }

        private void toolStripButtonpolygon_Click(object sender,
EventArgs e)
        {
            toolis = Drawtools._polygon;
            picbox_drawhere.Cursor = Cursors.Cross;
            Color_ToolsButtonHighLight(toolStripButtonpolygon);
            myPolygons = null;
        }
        private void toolStripButtonCircle_Click(object sender,
EventArgs e)
        {
            toolis = Drawtools._circle;
            picbox_drawhere.Cursor = Cursors.Cross;
            Color_ToolsButtonHighLight(toolStripButtonCircle);
```

```
            }
            #endregion
            #region KEYBOARD
            private void Mainform_KeyDown(object sender, KeyEventArgs e)
            {
                if (e.KeyCode == Keys.ShiftKey)
                {

                    reqStraightLine = true;
                }
            }

            private void Mainform_KeyUp(object sender, KeyEventArgs e)
            {
                if (e.KeyCode == Keys.ShiftKey)
                {

                    reqStraightLine = false;
                }
            }
            #endregion
            #region Drawing

            private void picbox_drawhere_Resize(object sender, EventArgs
e)
            {
                CreateTransforms();
                picbox_drawhere.Refresh();
            }
            private    void    printDocument_PrintPage(object    sender,
System.Drawing.Printing.PrintPageEventArgs e)
            {
                Graphics graphics = e.Graphics;
                graphics.SmoothingMode = SmoothingMode.AntiAlias;
```

```
                graphics.Transform = Transform;
                foreach (Shape item in shapes)
                    item.Render(graphics);

            }

            private void printDocument_QueryPageSettings(object sender,
System.Drawing.Printing.QueryPageSettingsEventArgs e)
            {
                e.PageSettings.PaperSize = new PaperSize("Construction",
picbox_drawhere.Width, picbox_drawhere.Height);
            }




            private    void    picbox_drawhere_Paint(object    sender,
PaintEventArgs e)
            {
                // If we don't have the transforms yet, get them.
                if (Transform == null) CreateTransforms();
                e.Graphics.SmoothingMode = SmoothingMode.AntiAlias;
                e.Graphics.Transform = Transform;

                Font font = new Font(FontFamily.GenericSansSerif, 0.15F,
FontStyle.Regular);
                // Use a pen that isn't scaled.
                using (Pen thin_pen = new Pen(Color.Green, 0.1F))
                {
                    // Draw the axes.
                    float tic = 0.08f;
                    thin_pen.Width = 2 / DrawingScale;
                    //Draw Horizontal Line
```

```
                    e.Graphics.DrawLine(thin_pen, Wxmin, 0, Wxmax, 0);

                    //Draw vertical tic
                    for (int x = (int)Wxmin; x <= Wxmax; x++)
                    {
                        if (x != 0)
                            e.Graphics.DrawString(x.ToString(),    font,
Brushes.Black, new PointF(x, tic));
                        e.Graphics.DrawLine(new           Pen(Color.Red,
thin_pen.Width), x, -tic, x, tic);

                    }

                    e.Graphics.DrawLine(thin_pen, 0, Wymin, 0, Wymax);
                    //Draw Horizontal tic
                    for (int y = (int)Wymin; y <= Wymax; y++)
                    {
                        if (y != 0)
                            e.Graphics.DrawString(y.ToString(),    font,
Brushes.Black, new PointF(tic, y));
                        e.Graphics.DrawLine(new           Pen(Color.Red,
thin_pen.Width), -tic, y, tic, y);
                    }

                }
                shapes.ReverseSort();
                foreach (Shape shape in shapes)
                    shape.Render(e.Graphics);

                if (!myRect.IsEmpty)
                {
                    e.Graphics.DrawRectangle(curPen, myRect.X, myRect.Y,
myRect.Width, myRect.Height);
```

```
                    e.Graphics.FillRectangle(new  SolidBrush(ourColor),
myRect.X, myRect.Y, myRect.Width, myRect.Height);
                }

                if (!myCirc.IsEmpty)
                {
                    e.Graphics.DrawEllipse(curPen,  myCirc.X,  myCirc.Y,
myCirc.Width, myCirc.Height);
                    e.Graphics.FillEllipse(new    SolidBrush(ourColor),
myCirc.X, myCirc.Y, myCirc.Width, myCirc.Height);
                    e.Graphics.DrawRectangle(curPen, myCirc.X, myCirc.Y,
myCirc.Width, myCirc.Height);
                }

                if (myPolygons != null)
                {
                    e.Graphics.DrawLine(curPen,
myPolygons[myPolygons.Count - 1], ePt);
                    e.Graphics.DrawLine(curPen, myPolygons[0], ePt);
                    if (myPolygons.Count > 1)
                    {
                        e.Graphics.DrawLines(curPen,
myPolygons.ToArray());
                        e.Graphics.FillPolygon(new
SolidBrush(ourColor), myPolygons.ToArray());
                    }

                }

            }

            void Set_Shape_Status()
            {
                labelxpointvalue.Text = Math.Round(ePt.X, 1).ToString();
```

```
                labelypointvalue.Text = Math.Round(ePt.Y, 1).ToString();
                labeldepthvalue.Text = abovedepth.ToString();
                if (toolis == Drawtools._select)
                {
                    labeldegreevalue.Text = "00";
                    labellengthvalue.Text = "00";

                    if (selectedShape != null)
                    {
                        labeldepthvalue.Text                          =
selectedShape.Depth.ToString();
                        if (selectedShape.ShapeType == RECTSHAPE)
                            labelwdithheightvalue.Text                =
Math.Round(selectedShape.Size.Width,   2).ToString()   +   "   x   "   +
Math.Round(selectedShape.Size.Height, 2).ToString();
                        if (selectedShape.ShapeType == CIRSHAPE)
                            labelwdithheightvalue.Text                =
Math.Round(selectedShape.Size.Width,   2).ToString()   +   "   x   "   +
Math.Round(selectedShape.Size.Height, 2).ToString();
                    }
                }
                if (toolis == Drawtools._polygon && myPolygons != null)
                {
                    float                    degree                   =
Math.Abs(Calculate.AngelBetweenTwopoints(sPt, ePt));
                    labeldegreevalue.Text      =      Math.Round(degree,
2).ToString();
                    float length = Calculate.LengthBetweenTwopoints(sPt,
ePt);
                    labellengthvalue.Text      =      Math.Round(length,
2).ToString();
                }
                if (toolis == Drawtools._rectangle && !myRect.IsEmpty)
                {
```

```
                    labelwdithheightvalue.Text                          =
Math.Round(myRect.Width, 2).ToString() + " x " + Math.Round(myRect.Height,
2).ToString();
                }
                if (toolis == Drawtools._circle && !myCirc.IsEmpty)
                {
                    labelwdithheightvalue.Text                          =
Math.Round(myCirc.Width, 2).ToString() + " x " + Math.Round(myCirc.Height,
2).ToString();
                }
            }
            private void ClearSelectedShape()
            {
                if (selectedShape != null)
                {
                    selectedShape.Selected = false;
                    picbox_drawhere.Refresh();
                }
                selectedShape = null;
                Set_Shape_Status();
            }
            #endregion


        }
    }
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ РОЗРАХУНКУ ВИМОГ НА ВИКОРИСТАННЯ БУДІВЕЛЬНИХ МАТЕРІАЛІВ

Виконав: студент 2-го курсу,
групи ЗКН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Сіваєв А.С.
(прізвище та ініціали)

Керівник: к.т.н., ст.в. каф. КН
_____ Петришин С.І.
(прізвище та ініціали)
«07» 12 2023 р.

Вінниця ВНТУ - 2023 рік

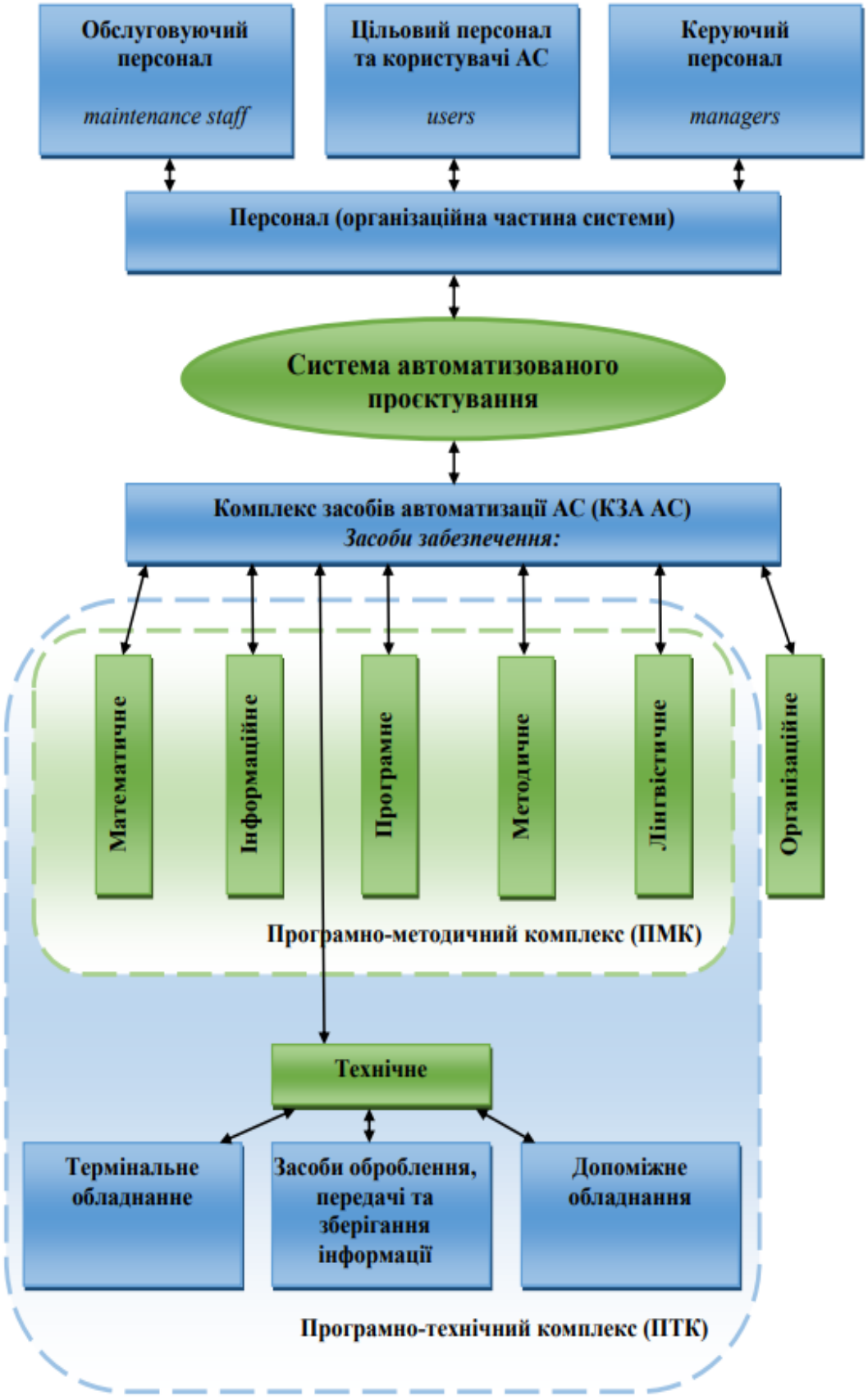

Рисунок В.1 – Загальна структурна схема роботи САПР

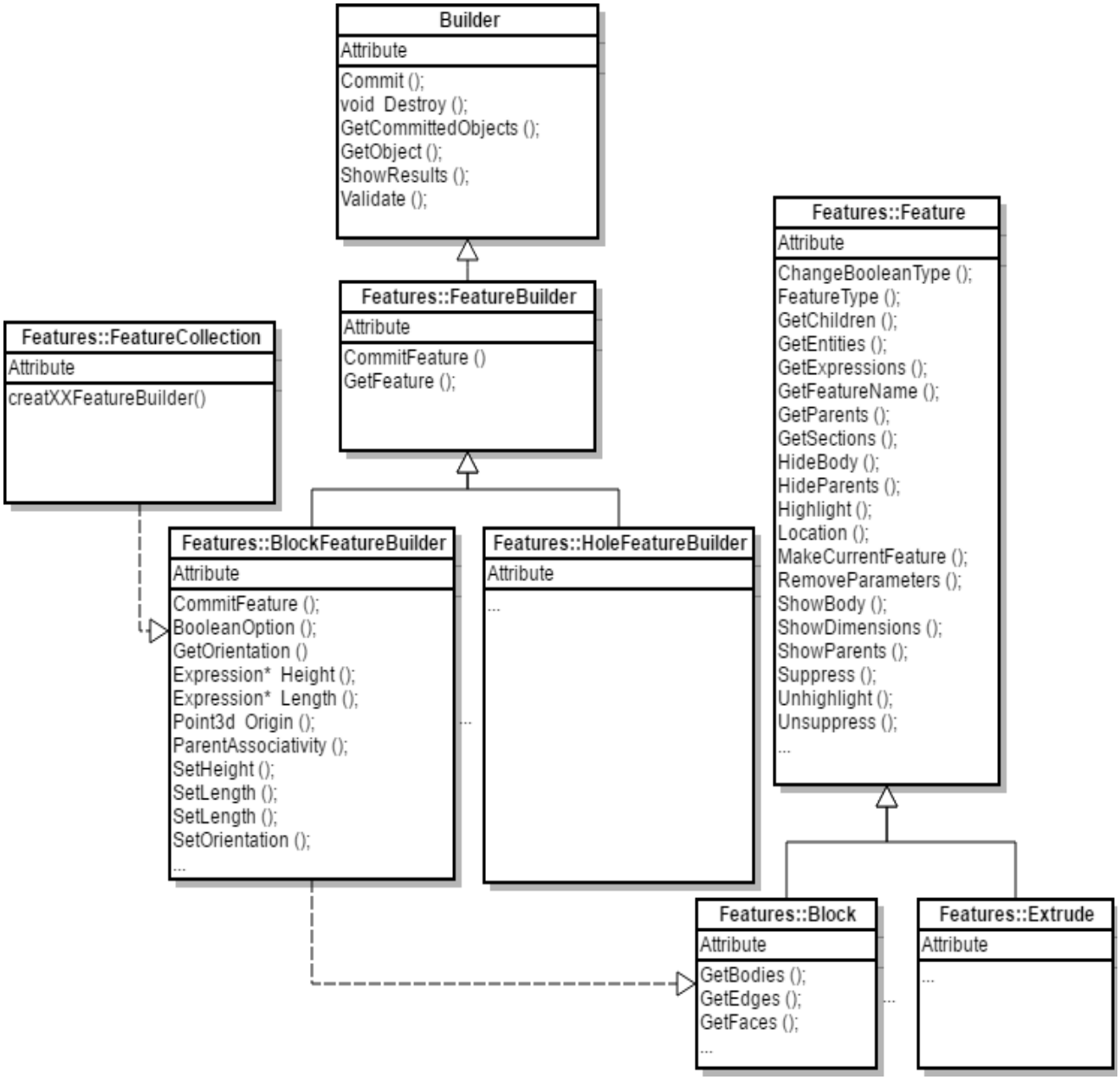

Рисунок В.2 – Юмл-діаграма класів САПР

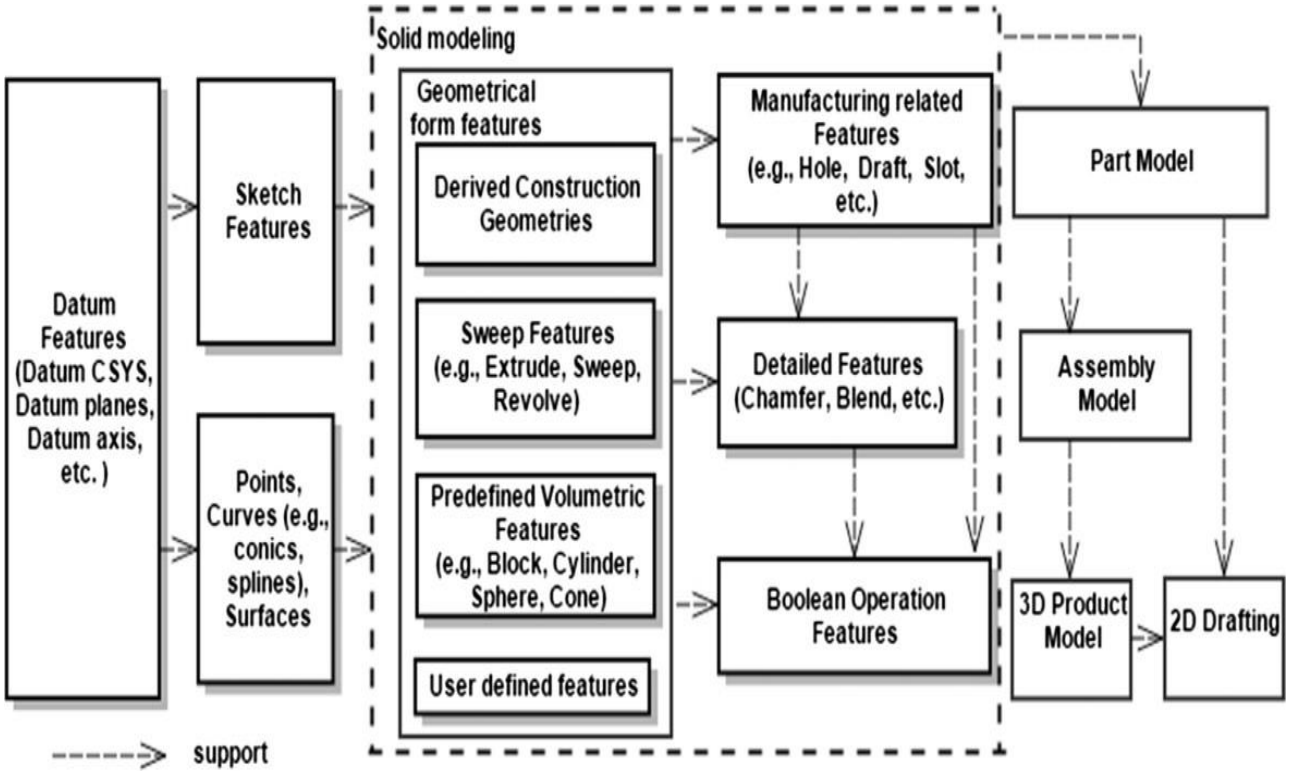

Рисунок В. 3 – Діаграма компонентів САПР

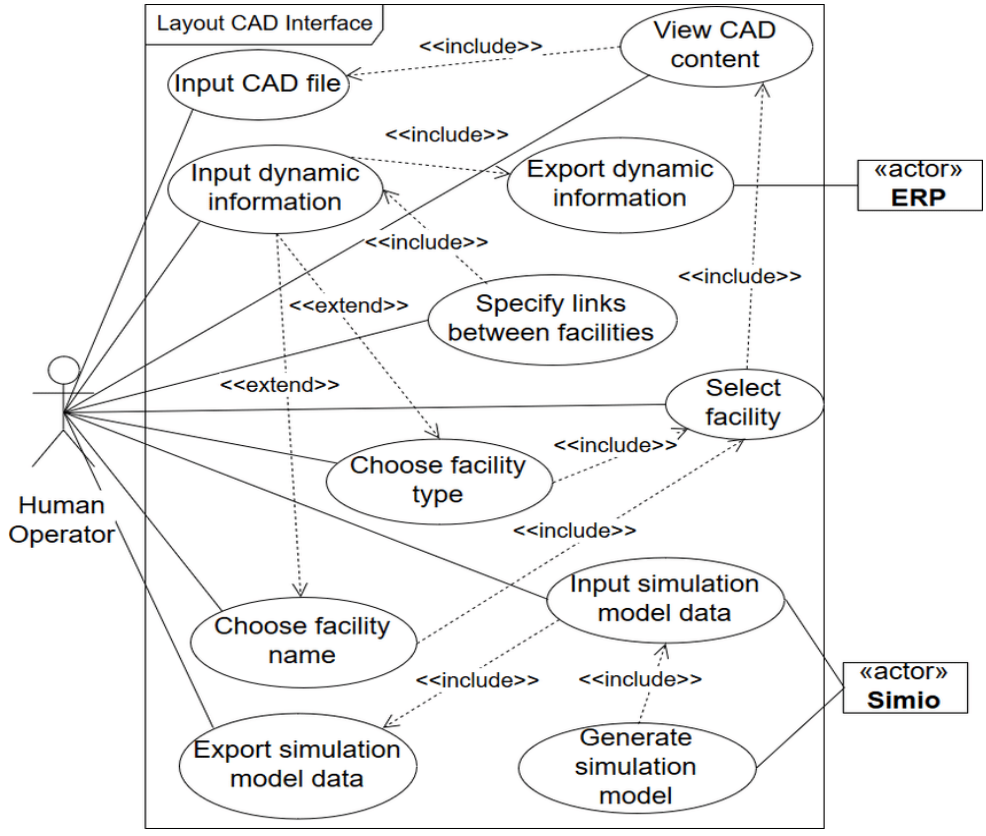

Рисунок В.4 – Діаграма прецедентів САПР

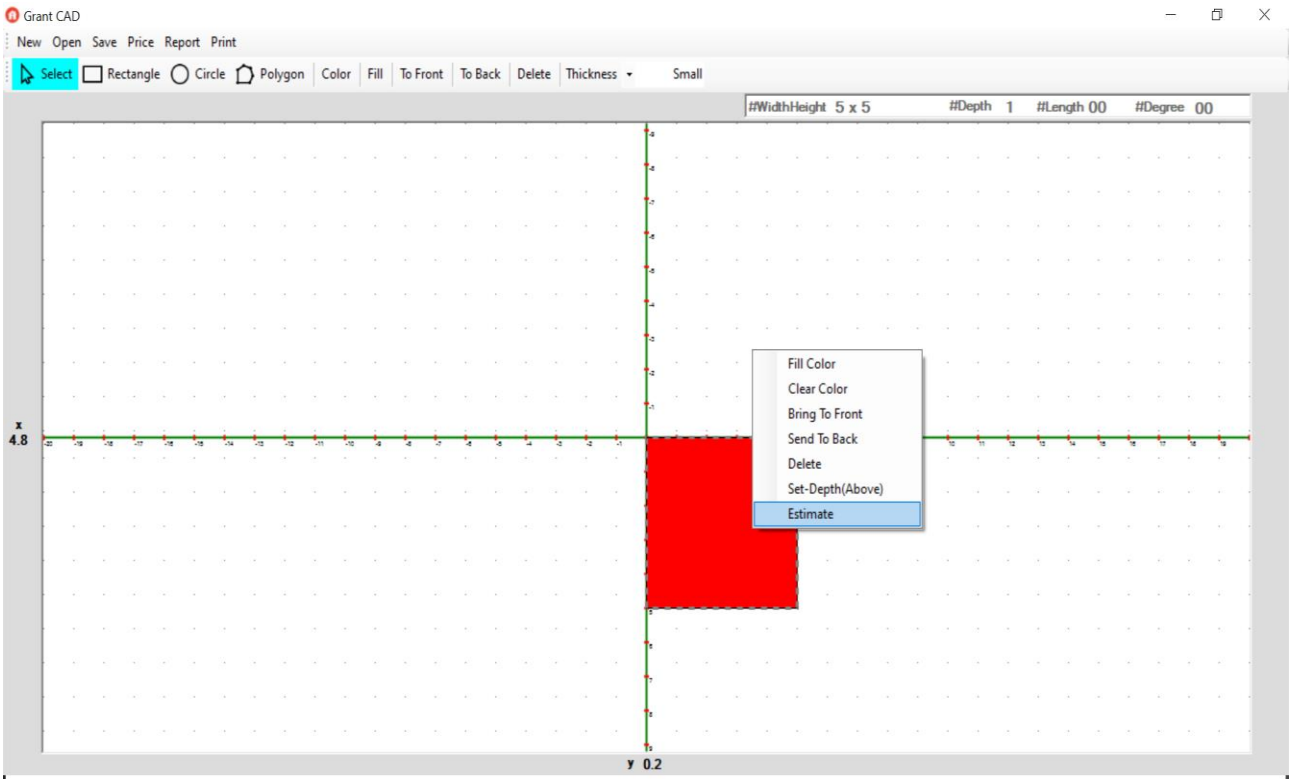

Рисунок В.5 – Нанесення схеми будівельних матеріалів

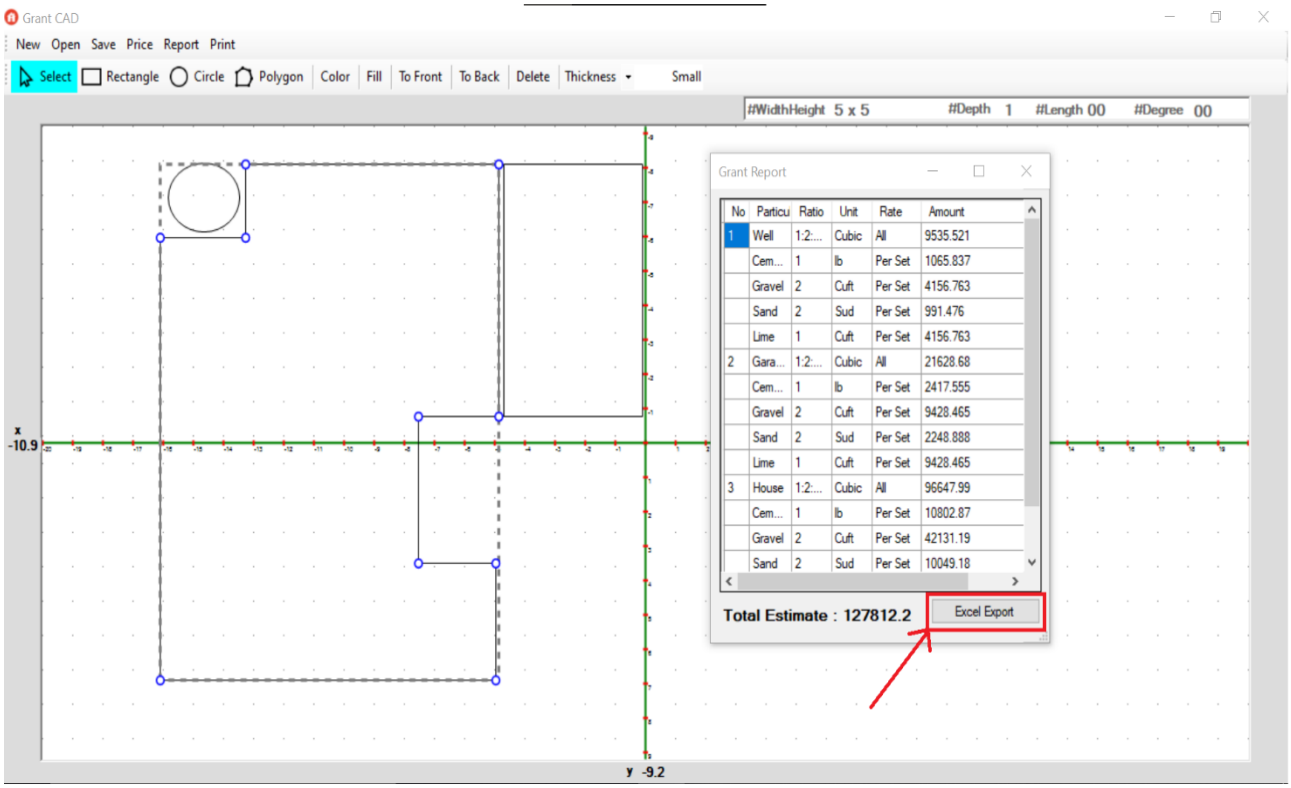

Рисунок В.6 – Отримання розрахунків

Додаток Г (довідниковий)

Інструкція користувача

Вікдриємо додаток GrantCAD.exe та почнемо малювати на ньому будівельні схеми за допомогою графічних примітивів.

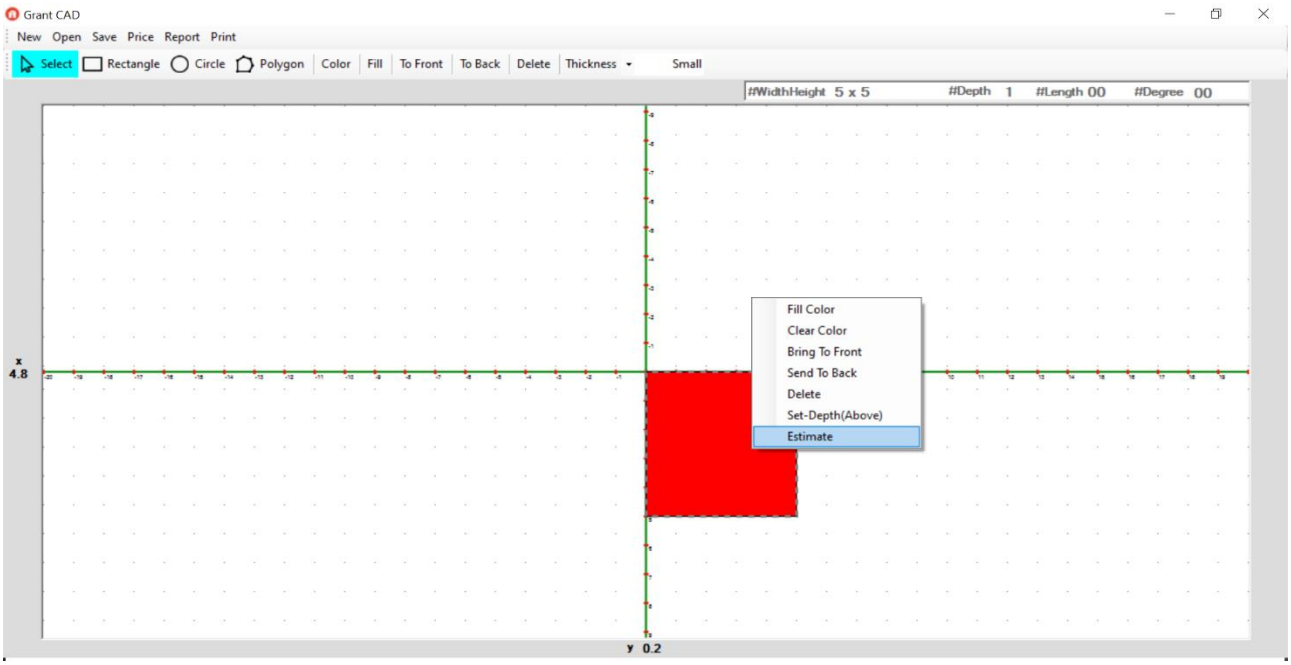

Рисунок Г.1 – Нанесення схеми будівельних матеріалів

Тоді натиснемо кнопку розрахувати витрати на матеріали.

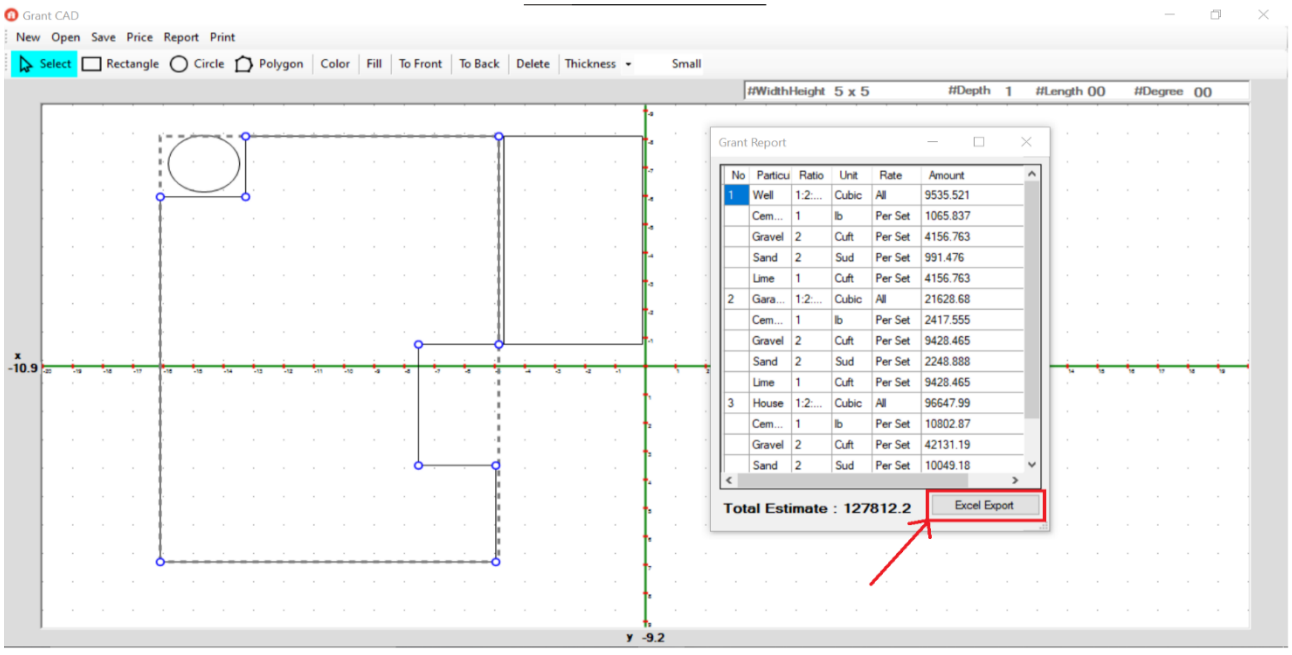

Рисунок Г.2 – Отримання розрахунків