

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Використання API інтерфейсу в програмному комплексі SolidWorks»

Виконав студент групи ЗПЗм-22-1
КОВАЛЕНКО Яна Павлівна

Керівник роботи:
ЄФРЕМОВ Юрій Миколайович

Рецензент:
САВІЦЬКИЙ Роман Святославович

Житомир – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

зав. кафедри інженерії

програмного забезпечення

Тетяна ВАКАЛЮК

«18» вересня 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КОВАЛЕНКО Яна Павлівна**

Керівник роботи: **ЄФРЕМОВ Юрій Миколайович**

Тема роботи: **«Використання API інтерфейсу в програмному комплексі SolidWorks»**, затверджена Наказом закладу вищої освіти від **«18» вересня 2023 р. № 481/с**

Вихідні дані для роботи: **предметом дослідження є засоби розробки програмного забезпечення з використанням API інтерфейсу. Об'єктом дослідження є програмний модуль для автоматичного редагування розмірів деталей машин у програмному комплексі SolidWorks.**

Консультанти з магістерської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Єфремов Ю.М.	22.09.2023р.	22.09.2023р.
2	Єфремов Ю.М.	22.09.2023р.	22.09.2023р.
3	Єфремов Ю.М.	22.09.2023р.	22.09.2023р.

РЕФЕРАТ

Індивідуальне завдання кваліфікаційної магістерської роботи полягає в обґрунтуванні актуальності теми, аналізі предметної галузі, у підвищенні продуктивності роботи конструкторів-проектувальників, а саме в модифікації деталей виробів в 3D збірках за допомогою розробленого модуля при застосуванні API інтерфейсу в SolidWorks для зменшення витрат часу на проектування технічної документації, та на її редагування.

Кваліфікаційна магістерська робота на тему «Використання API інтерфейсу в програмному комплексі SolidWorks» складається з теоретичного і технічного комплексу, та пояснювальної записки. Пояснювальна записка складається зі вступу, 3 розділів, списку літератури з 29 найменувань, 19 рисунків та з додатків. Повний обсяг роботи становить 65 сторінок.

КЛЮЧОВІ СЛОВА: SOLIDWORKS, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, БІБЛІОТЕКА КЛАСІВ, API, САПР, CAD-СИСТЕМИ.

ABSTRACT

The individual task of the qualifying master's thesis is to substantiate the relevance of the topic, analyze the subject area, increase the productivity of designers, namely, to modify product parts in 3-D assemblies using the developed module when using the API interface in SolidWorks to reduce the time spent on designing technical documentation and editing it.

The qualification master's thesis on the topic "Using the API interface in the SolidWorks software package" consists of a theoretical and technical complex, and an explanatory note. The explanatory note consists of an introduction, 3 chapters, a list of references of 29 titles, 19 figures, and appendices. The total volume of the paper is 65 pages.

KEYWORDS: SOLIDWORKS, SOFTWARE, CLASS LIBRARY, API, SAPR, CAD SYSTEMS.

ЗМІСТ

РЕФЕРАТ.....	3
ABSTRACT.....	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ.....	10
1.1. Загальні відомості з предметної галузі.....	10
1.2. Аналіз прикладних програм із застосуванням API в SolidWork.....	12
1.3. Аналіз існуючих автоматизованих систем SolidWorks.....	17
1.4. Призначення та галузь застосування CAD-систем: постановка завдання для подальшої розробки.....	26
Висновки до розділу 1.....	27
РОЗДІЛ 2. РОЗРОБКА ТА ПРОЕКТУВАННЯ САПР НА БАЗІ API.....	28
2.1. Функціональне призначення програми.....	30
2.2. Архітектура проектування API SolidWorks.....	34
2.3. Інтерфейс програмування застосунків (API) SolidWorks.....	35
2.4. Вибір інструментальних засобів для розробки API.....	36
2.5. Застосування API інтерфейсу для програмного комплексу SolidWorks....	40
Висновки до розділу 2.....	46
РОЗДІЛ 3. ЕКОНОМІЧНА СКЛАДОВА.....	47
3.1. Розрахунок трудомісткості та вартості розробки програмного продукту..	47
3.2. Рахунок витрат на створення програми.....	49
Висновки до розділу 3.....	50
ВИСНОВКИ.....	51
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	52
ДОДАТКИ.....	56

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface

САПР – Система Автоматизованого Проектування

CAD – Computer Aided Design

COM – Component Object Model

PDM – Product Data Management

ВСТУП

Актуальність теми. На сьогоднішній день програмний комплекс SolidWorks є однією із найпопулярніших автоматизованих систем проектування, як в Україні, так і в закордонних країнах. І все частіше користування цією програмою розглядається не тільки в контексті 3D-моделювання, а й при застосуванні прикладного програмного забезпечення.

Розробка програмних продуктів, застосування різних Application Programming Interface (API-інтерфейсів), та поєднання Product Data Management (PDM) з системами автоматизованого проектування (САПР) дає змогу автоматизувати різні етапи інженерно-конструкторських проектів, що значно спростить роботу конструктора-проектувальника, та звільнить його від повторів «рутинних» операцій.

Актуальність розвитку САПР, зокрема SolidWorks, визначається великим попитом в машинобудуванні, та зростає з кожним роком із-за застосування програмного комплексу в різних галузях [1, 5, 16, 22, 23].

Впровадження сучасних технологій в промисловій галузі дозволяє досягати високих результатів на підприємствах з виробництва деталей машин, та бути конкурентоспроможним на ринку машинобудівної індустрії.

Подальше впровадження сучасних комп'ютерних технологій на промислових підприємствах потребує вирішення задач із впровадження різних САПР, що дозволять автоматизувати весь виробничий цикл на підприємстві: від розробки виробів до проектування технології їх виготовлення.

На багатьох підприємствах широко розвинувся процес впровадження різних підсистем САПР (CAD/CAM-систем).

Автоматизація підготовки промислового виробництва дає можливість іншим підприємствам швидко реагувати на зміну попиту, у короткий термін випускати нові види виробів, швидко модернізувати продукцію, що випускається, та відслідковувати життєвий цикл виготовлення деталей машин, що дає змогу ефективно підвищувати якість.

САПР залишаються надзвичайно актуальними у сучасній машинобудівній галузі не тільки в нашій країні, а й зарубіжних. А також вони є необхідним інструментом для розвитку та вдосконалення різних сфер, зокрема інженерії, архітектури, технічного виробництва та інших галузей.

Ключові аспекти включають в себе:

- **Ефективність та точність:** Вони дозволяють створювати проекти з високою точністю і швидкістю, зменшуючи час на розробку та виробництво продукції.

- **Витрати:** САПР можуть значно зменшити витрати на проектування, оскільки дозволяють уникнути помилок та сприяють ефективному використанню ресурсів.

- **Колаборація:** Вони полегшують спільну роботу між різними командами та спеціалістами, що працюють над проектом, навіть якщо вони знаходяться в різних частинах світу.

- **Інновації:** САПР постійно розвиваються, дозволяючи впроваджувати нові технології та методи проектування, що сприяє інноваціям у виробництві.

- **Стандартизація:** Вони дозволяють дотримуватися стандартів і вимог якості у проектуванні, що є надзвичайно важливим у багатьох галузях.

Загалом, САПР є невід'ємною частиною сучасної промисловості, вони не лише полегшують процес проектування та розробки, але й сприяють інноваціям, зменшенню витрат та покращенню якості продукції [1, 2, 3].

Мета і завдання роботи. Метою роботи є розробка програмного модуля за допомогою застосування API інтерфейсу для автоматичного редагування розмірів деталей в 3D збірках технологічних виробів машин в САПР SolidWorks.

Досягнення поставленої мети передбачає розв'язання таких **завдань:**

- обґрунтувати актуальність теми роботи та напрямку досліджень;
- здійснити аналіз предметної галузі для виявлення проблем даної тематики та визначити загальний стан сфери автоматизованого проектування;

- здійснити аналіз CAD-систем, які використовуються на промислових підприємствах;
- скласти алгоритм роботи програмного модуля в SolidWorks для модифікації розмірів деталей в збірках 3D за допомогою API інтерфейсу;
- розробити програмний модуль за допомогою API інтерфейсу в програмному комплексі SolidWorks для можливості автоматично редагувати розміри деталей в 3D збірках;
- запропонувати інструкцію для використання.

Об’єкт дослідження є програмний модуль для автоматичного редагування розмірів деталей машин у SolidWorks.

Предметом дослідження є засоби розробки програмного забезпечення з використанням API інтерфейсу.

Методи дослідження. В кваліфікаційній роботі було застосовано програмний пакет C# для побудови програмного модуля, та програмний комплекс SolidWorks для створення 3D-моделі.

Наукова новизна отриманих результатів полягає в такому:

- сформовано алгоритм роботи програмного модуля САПР;
- розроблено програмний модуль на базі API в SolidWorks для модифікації розмірів деталей в збірках 3D.

Практичне значення роботи полягає у такому:

- підвищення продуктивності праці конструктора-проектувальника за рахунок зменшення повторів «рутинних» операцій;
- зменшення часу на розробку технічної документації, що дозволить скоротити терміни її виготовлення;

Апробація результатів роботи. Матеріали кваліфікаційної роботи доповідалися на VI Всеукраїнської науково-технічної конференції: “Комп’ютерні технології: інновації, проблеми, рішення – 2023 (КТІПР-2023)” 29-30 листопада 2023 р.

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Загальні відомості з предметної галузі

Безперервний розвиток машинобудівної галузі диктує нові запити для вдосконалення науково-технічного прогресу. Збільшились вимоги до максимального скорочення часу, що витрачається на весь цикл розробки виробів, зокрема креслеників деталей машин.

Складність інженерних рішень обумовлює розробку нових автоматизованих методів для забезпечення високої якості виготовлення деталей машин, технологічності створення складальних креслеників та конкурентоспроможності на ринку праці.

При створенні таких автоматизованих систем проектування все частіше додається прикладне програмне забезпечення для оптимізації проектно-конструкторських робіт.

САПР використовуються в різних галузях, таких як машинобудування, будівництво, електроніка, авіація, автомобільна промисловість та інші. Ці системи дозволяють інженерам створювати 2D та 3D моделі, виконувати аналізи, тести та оптимізації продуктів перед фізичним виробництвом [4, 6, 8].

В теперішній час, одним із найпопулярніших систем проектування, як в Україні так і в світі, є програмний комплекс SolidWorks. Він пропонує широкий спектр функціональних можливостей, які дозволяють створювати як 2D, так і 3D геометрію при проектуванні виробів деталей машин, а також працює на платформі Microsoft Windows, що в свій час було вперше для САПР при моделюванні твердотільних деталей.

Зараз інженера-конструктори використовують SolidWorks на всіх етапах технологічної та конструкторської підготовки, що забезпечує розробку виробів різних видів складності. А в останній час все частіше при 3D-моделюванні застосовуються розроблені прикладні програмні забезпечення, які допомагають автоматизувати різні задачі.

У промисловості самі застосовувані CAD – системи можуть бути основою для побудови САПР, які здатні вирішувати задачі з розрахунків та проектування конкретного класу виробів. Одними із самих відомих CAD-систем, що дозволяють використовувати API-технології є SolidWorks [5].

API в SolidWorks відіграє важливу роль у розширенні функціональності цієї CAD-платформи та в її інтеграції з іншими програмами та системами. Параметри API у SolidWorks є важливим:

- Розширення можливостей: API надає доступ до внутрішніх функцій SolidWorks, що дозволяє розширювати та налаштовувати його функціональність, створювати нові інструменти та функції відповідно до потреб користувача.
- Автоматизація: API дозволяє автоматизувати рутинні завдання та операції в SolidWorks. Це може значно збільшити продуктивність та зменшити час, необхідний для виконання операцій з моделями.
- Інтеграція з іншими системами: За допомогою API SolidWorks можна легко інтегрувати цю CAD-платформу з іншими програмами та системами, такими як системи управління даними, ERP-системи, системи виробництва тощо.
- Розробка власних додатків: Користувачі можуть створювати власні програми, скрипти або додатки на основі API SolidWorks для вирішення специфічних задач або для вдосконалення процесів роботи з моделями.
- Створення спеціалізованих рішень: API дозволяє розробникам створювати спеціалізовані рішення для різних галузей, наприклад, програмне забезпечення для аналізу, виробництва, документації тощо.
- Доступ до функціональності: API надає доступ до широкого спектру функцій SolidWorks, включаючи створення та редагування моделей, керування кресленнями, обробкою даних та багато іншого.

API SolidWorks є потужним інструментом для розробників та інженерів, які працюють з цією CAD-платформою. Він відкриває широкі можливості для автоматизації процесів, розширення функціональності та інтеграції з іншими системами для оптимізації робочого процесу.

Взагалі, програмний комплекс SolidWorks є найпопулярнішим серед відомих застосувань, та складає 19,6%, що є самим високим % серед інших продуктів всього ринку станом на 2020 рік (Рис.1).[3]

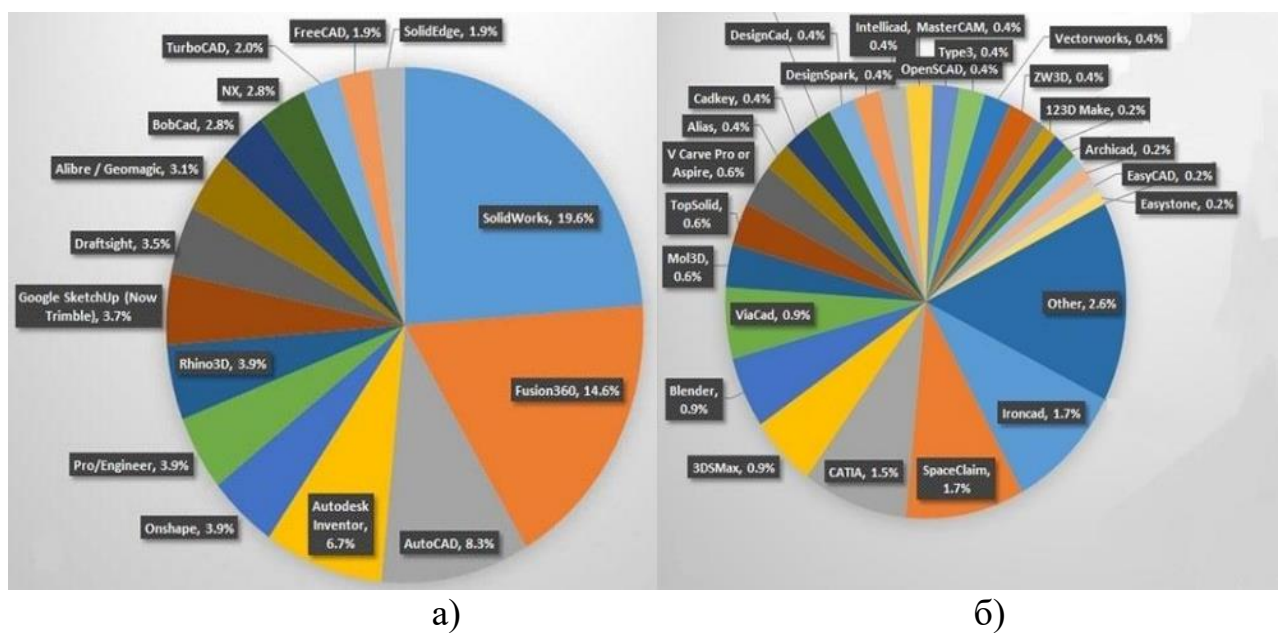


Рис.1. Розподіл ринку CAD-систем на період 2020 року:

а) розподіл сектору у 82.6% всього ринку;

б) розподіл залишкових 17.4%

Можна зробити висновок, що цей напрямок роботи дуже актуальний в різних галузях, зокрема промислово-виробничій, будівельній, конструкторсько-технологічній, технічно-навчальній та інших напрямках.

1.2. Аналіз прикладних програм із застосуванням API в SolidWorks

API - інтерфейс прикладного програмування. В тому числі дозволяє розробляти різні програми на платформі САПР SolidWorks.

API передбачає забезпечення можливостей зручного підключення зовнішніх модулів, які пишуться на мовах високого рівня, а також покращує роботу в мережі при застосуванні універсальних систем в специфічних предметних галузях.

API визначає функціональність, що надає, наприклад, бібліотека чи модуль, та при цьому абстрагується від того, як ця функціональність була реалізована. Також містить в собі сотні функцій, які можна визивати із програм Microsoft Visual Basic, VBA (Microsoft Excel, Word, Access), Microsoft Visual C, C++, .NET, а також файли-макроси SolidWorks.

Існує багато прикладних програм, які використовують API SolidWorks для різних цілей, від автоматизації операцій до розширення функціональності. Ось деякі з них:

CAD Automation Tools: Багато програм використовують API SolidWorks для автоматизації процесів створення моделей та креслень. Це може бути автоматичне створення шаблонів деталей або складань, генерація звітів, автоматичне додавання міток тощо.

ERP/PLM інтеграція: Деякі програми використовують API SolidWorks для інтеграції з системами управління даними (PDM/PLM) або ERP-системами для автоматизації обміну даними, витягування та оновлення інформації про моделі.

Simulation Software: Деякі пакети програмного забезпечення для чисельного моделювання (наприклад, ANSYS, Abaqus) можуть використовувати API SolidWorks для імпорту та редагування геометрії з моделей SolidWorks для подальшого аналізу.

CAM Software: Пакети програмного забезпечення для комп'ютерного підготовки виробництва (наприклад, Mastercam, CAMWorks) можуть використовувати API SolidWorks для автоматичного створення обробочних операцій на основі моделей SolidWorks.

Product Customization Tools: Деякі програми дозволяють користувачам налаштовувати продукти на основі моделей SolidWorks, змінюючи параметри або вибираючи різні опції.

Documentation and Visualization Tools: Додатки для створення документації, візуалізації, анімацій або інструкцій з виготовлення на основі моделей SolidWorks використовують API для доступу до геометрії та інших даних.

Design Automation Solutions: Програми для автоматизації процесу проектування, що базуються на API SolidWorks, дозволяють автоматично створювати або адаптувати дизайни відповідно до введених параметрів [11, 13].

Це лише кілька прикладів того, які програми використовують API SolidWorks для різних цілей. Використання API відкриває безліч можливостей для розробників та компаній, щоб оптимізувати та розширювати функціональність SolidWorks у різних сферах виробництва та дизайну.

Для SolidWorks можна використовувати API для вирішення різних задач. Найбільш поширеними є:

- Розробка додаткових модулів для розширення функціональності;
- Інтеграція SolidWorks з іншими CAD-, CAM- застосунками для збільшення базових можливостей SolidWorks.

На сьогоднішній день існує чимало прикладних програм, що вирішують цілий ряд специфічних задач в SolidWorks. Прикладами таких програм, які використовують API наведені нижче:

1) Модуль автоматичної генерації SWR-специфікації (Рис.2);

The screenshot shows the SolidWorks SWR specification generation interface. On the left is a tree view with the following structure:

- KБ20.01.00.00_Мотор
 - Примечания
 - Спецификация8
 - 1
 - Формат листа1
 - Чертежный вид1
 - Чертежный вид2
 - Чертежный вид3
 - Чертежный вид4
 - C-1
 - Формат листа2
 - C-2
 - Формат листа8

The main table on the right is a specification table with the following data:

Формат	Зона	Лист	Обозначение	Наименование	Кол.	Примечание
				Документация		
A4			КБ20.01.00.00СБ	Сборочный чертеж		
				Сборочные единицы		
A4	1		КБ.01.01.02.01	Шкив мотора	1	
A4	2		КБ20.01.02.00-02	Вал	1	
A4	3		КБ20.01.03.00	Верхняя пластина	1	
				Детали		
A4	4		КБ20.01.00.01	Пластина	1	
A4	5		КБ20.01.00.02	Входной кронштейн R8	1	
A4	6		КБ20.01.00.03	Сторонная плита R5	1	
A4	7		КБ20.01.00.04	Втулка	1	*) A4, A3
A4	8		КБ20.01.00.05	Подшипник (16mm x 8mm x 5mm)	1	
A4	9		КБ20.01.00.06	Втулка бронзовая	2	
A4	10		КБ20.01.00.07	Винт потайной	6	
				Стандартные изделия		
	11			Винт DIN 965 4-40 x 0.25	5	
	12			Винт DIN 7985 8-32 x 0.625	3	
	13			Кольцо стопорное DIN 471 6	1	

Below the table, there is a summary section for the assembly:

КБ20.01.00.00

Мотор

Лист 1 из 2

КБ 2.0

Формат A4

Рис.2. Приклад вигляду SWR-специфікації в SolidWorks

Модуль автоматичної генерації специфікацій забезпечує повне автоматизоване заповнення переліку деталей та їх кількості. Є можливість двохстороннього асоціативного зв'язку креслення з моделями SolidWorks.

Також є вбудовані бібліотеки матеріалів, та добре налаштований експорт даних, генерація інших конструкторсько-технологічних документів.

2) Бібліотека стандартних виробів та SWR-бібліотеки (Рис.3);

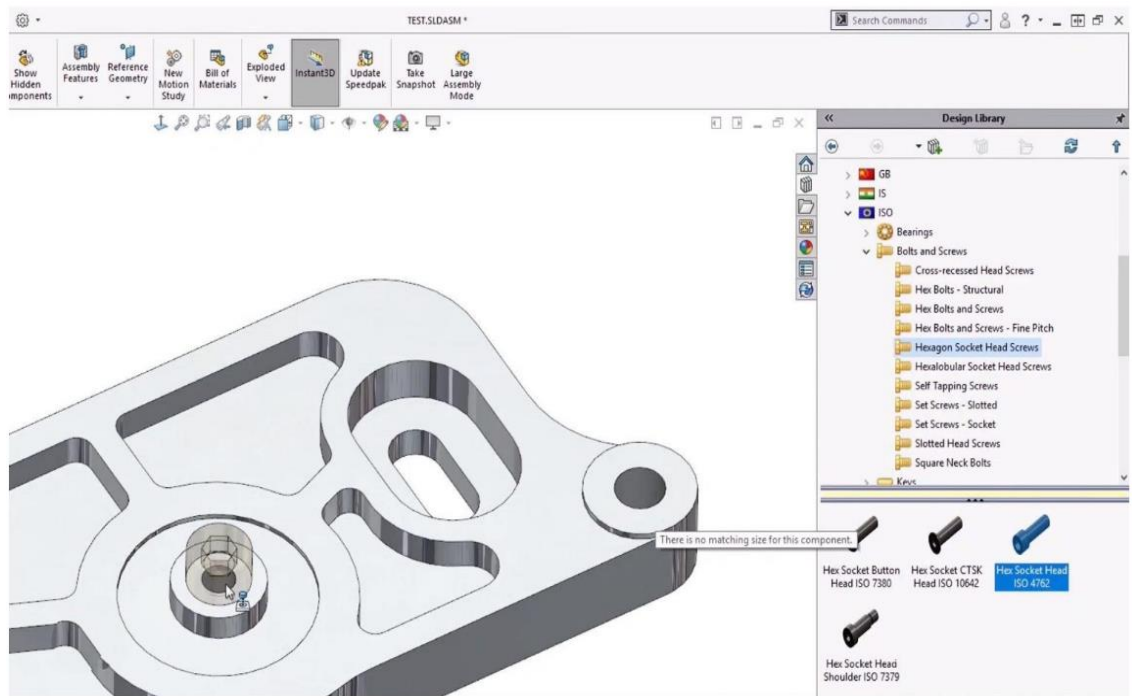


Рис.3. Приклад бібліотеки стандартних виробів

Бібліотеки стандартних виробів є доповнюваними параметризованими моделями, та є можливість розділення доступу адміністратора та користувача. Також одним із переваг є можливість доповнення елементів із бібліотек прямо у збірку з накладенням спряжень, та інтеграція з модулем автоматичної генерації специфікацій.

3) Система проставлення допусків та посадок TolerAction (Рис.4);

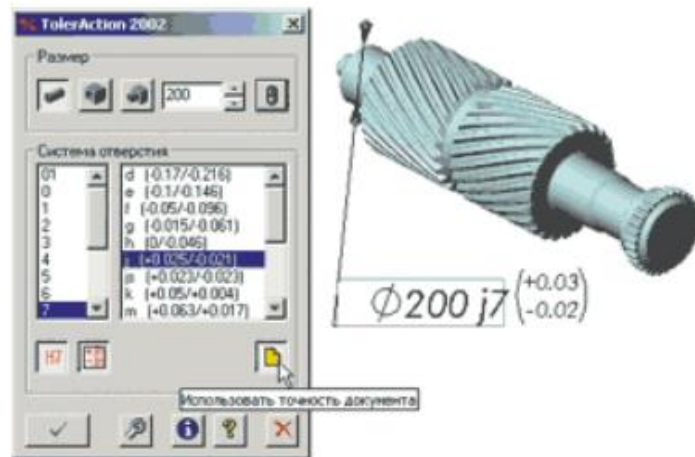


Рис.4. Приклад системи проставлення допусків та посадок

Система проставлення допусків та посадок має широкі можливості у роботі з системами отворів та валів при виборі стандартних значень квалітету та полів допуску, який відповідає розміру. Є можливість автоматичного реагування на зміну номінальних значень розмірів, яким було призначено допуски, та їх оновлення, а також у роботі в контексті деталі, збірки та креслень [15, 18].

Вище перераховані модулі, які зроблені за допомогою API інтерфейсів, є дуже важливою складовою при проектуванні різних виробів в SolidWorks з регулярним оновленням версій, високою якістю виконання та з кваліфікаційною технічною підтримкою.

1.3. Аналіз існуючих автоматизованих систем SolidWorks

Аналіз існуючих автоматизованих систем SolidWorks включає в себе огляд різноманітних рішень, що використовуються для підтримки різних аспектів роботи з програмою SolidWorks. Ось деякі з найпопулярніших систем, які можна розглянути:

SolidWorks PDM (Product Data Management):

Це система керування даними, яка дозволяє зберігати та керувати CAD-даними та іншими відомостями, пов'язаними з проектами. PDM допомагає управляти версіями файлів, забезпечує збереження даних та їх доступність для команди.

SolidWorks CAM (Computer-Aided Manufacturing):

Це рішення, яке інтегрується з SolidWorks для автоматизації процесу виготовлення деталей та складання CNC-програм.

SolidWorks Simulation:

Це пакет програмного забезпечення для вирішення завдань чисельного моделювання, таких як аналіз механічних властивостей, розрахунків міцності тощо.

SolidWorks Electrical:

Ця система спеціалізується на проектуванні електричних схем та систем.

SolidWorks Composer:

Це програмне забезпечення для створення документації, технічних мануалів, анімацій та інструкцій з виготовлення на основі моделей SolidWorks.

SolidWorks Inspection:

Це інструмент для автоматизації процесу створення інспекційних звітів для деталей та зборок.

SolidWorks Visualize:

Це програмне забезпечення для створення реалістичних візуалізацій на основі моделей SolidWorks для презентацій та маркетингу.

Такі системи допомагають різним відділам інженерних компаній ефективно використовувати потенціал SolidWorks у своїх проектах, автоматизуючи багато аспектів процесу розробки та виробництва. Кожна з них спеціалізується на певних аспектах життєвого циклу виробу і надає інструменти для розв'язання конкретних завдань.

Програмний комплекс SolidWorks включає:

- Проектування виробів з врахуванням специфіки виготовлення: прес-форми, листовий матеріал, штампи металоконструкцій, деталі із пластмас;
- Проектування збірок. Робота зі складними зборками: SpeedPak – управління продуктивністю системи, управління відображеннями, конфігураціями, режим скорочених збірок та креслень.
- Гібридне параметричне моделювання: твердотільне моделювання, моделювання поверхонь, каркасне моделювання та їх комбінація без обмежень ступені складності.
- Бібліотеки проектування: єдина бібліотека фізичних властивостей матеріалів, текстур та штриховок. Типові конструктивні елементи, стандартні деталі та вузли, елементи листових деталей, профілі прокатного асортименту.
- Пряме редагування геометрії: технології Instant3D.
- Проектування на основі без знань: технології DriveWorksXpress.
- Експертні системи.

А також можна розглядати такі складові:

- SketchXpert – аналіз конфліктів в ескізах, пошук оптимального рішення.
- FeatureXpert, FilletXpert, DraftXpert – автоматичне управління елементами скруглень та уклонів, оптимізація порядку створення моделі.
- Instant3D – динамічне пряме редагування 3D моделей та збірок, стандартних компонентів.
- DimXpert – автоматична постановка розмірів та допусків в 3D моделях деталей, а також розмірів в креслениках, можливість роботи з імпортованою геометрією.
- AssemblyXpert – аналіз продуктивності великих збірок, підготовка варіантів рішень по покращенню швидкодії.
- MateXpert – аналіз спряжень збірок, пошук оптимального рішення.

- Інженерний аналіз: експрес-розрахунки масово-інерційних характеристик, кінематики та динаміки механізмів, міцності та гідродинаміки.
- Аналіз технологічності моделі: металообробка, лиття, заповнення прес-форм, обробка листів.
- Екологічна експертиза проекту: технології SustainabilityXpress.
- Оформлення креслень по ЄСКД.
- STEP AP203/AP214.
- Parasolid,
- ACIS.
- IGES.
- VDAFS.
- STL.
- VRML.
- Pro/ENGINEER.
- NX.
- Solid Edge.
- Inventor.
- AutoCAD.
- CATIA Graphics.
- P-CAD.
- Altium Designer.
- Mentor Graphics.
- CADENCE та інші.
- Анімація: виготовлення мультиплікації на основі 3D моделей.
- API SDK: підтримка програмування на мовах Visual Basic, Visual C++ та інших, записи та редагування макросів (VBA).

Практика показала, що із всіх завдань автоматизованої системи управління найбільш типізованими є завдання автоматизації проектування в

питаннях підготовки виробництва та його управління. Тому була потреба у нових автоматизованих системах: CAD/CAM/CAE і MRP (MRP II).

CAD/CAM/CAE - це комплекс програмних засобів комп'ютерного проектування, підготовки виробництва й інженерних розрахунків.

MRP (Materials Requirement Planning) - призначена для планування потреб у матеріалах, а вдосконалений варіант цієї системи - *MRP II (Manufacturing Resource Planning)* - став загальноприйнятим засобом рішення комплексу завдань управління фінансово-господарською діяльністю підприємства: планування виробництва, матеріально-технічного постачання, управління фінансовими ресурсами й ін.

Оптимальним рішенням проблеми інформаційної взаємодії САПР на виробництві є започатковування базової САПР, що буде виступати з'єднувальним компонентом, який поєднує різні результати інженерної промисловості в одну сформовану систему.

Проаналізувавши SolidWorks як інформаційну систему, можна відокремити деякі властивості:

1. Розширення;
2. Схильність до інтеграції;
3. Інтероперабельність;
4. Масштабування;

Розширення: інформаційна система повинна бути легкою в редагуванні, щоб можна було вносити зміни та не деформувати цілісність системи.

При виборі САПР одним із важливих факторів є питання розширення. Тому програмний комплекс SolidWorks пропонує широкий перелік різних доповнень. Розробники SolidWorks використали доповнення MSC.visualNastran, Sigmund1D и EmbassyWorks. На це впливало два фактори: перший дозволяє вирішувати поставлені задачі з мінімальними затратами часу та коштів, а другий показує, що ці зміни є важливими для користувача, який

використовує SolidWorks, і виключає труднощі з передачею геометрії, та повністю зберігає параметризацію, спрощує роботу з даними комплексу. Все перераховане значно збільшує функціонал всієї системи.

Маштабування: властивість інформаційної системи справлятися зі збільшенням робочого навантаження. Розробники приділяють більше уваги роботі з комплексними збірками, де кількість компонентів може досягати десятки та сотні деталей. При таких збірках доцільно використовувати спеціальний режим у SolidWorks, який називається «Режим роботи з великими збірками». Цей режим дозволяє оптимально розкидати програмні та апаратні ресурси для збереження часу завантаження та перебудови збірки.

Схильність до інтеграції: ця властивість системи має можливість спільної роботи різних підсистем в одній системі так, щоб окремі функції підсистеми отримали нові властивості як єдиної системи. Інтеграція на рівні застосунків має багато переваг: забезпечення наскрізного циклу інформаційної взаємодії застосунків в межах інформаційних систем; автоматизоване виконання сервісних застосунків по команді із базових складових інформаційної системи. Тому одною із основних вимог є можливість програмного керування набором функціональних речей, що реалізовані в цих компонентах.

Інтероперабельність: це можливість продукту або системи, в яких інтерфейси повністю відкриті, щоб взаємодіяти та функціонувати з іншими продуктами та системами без обмежень доступу та реалізації. Наприклад, для застосування SolidWorks разом з AutoCAD, то розробники пропонують додаток XchangeWorks. При встановленні SolidWorks програма інсталяції автоматично визначає чи встановлений AutoCAD на вашому комп'ютері, та пропонує встановити систему взаємодії цих двох програмних комплексів.

Автоматизовані системи підтримки життєвого циклу виробів (PLM) - це комплексні програмні засоби, які допомагають управляти життєвим циклом

виробу від його створення та розробки до використання, удосконалення та відновлення.

Основні аспекти систем PLM включають:

Управління даними: Зберігання, організація та керування даними про виробу, включаючи специфікації, CAD-моделі, технічні креслення, вимоги клієнтів, інструкції з виготовлення тощо.

Керування змінами: Відстеження та управління змінами в конструкції виробу протягом всього життєвого циклу, забезпечення контролю версій та історії змін.

Управління процесами: Планування та виконання процесів, пов'язаних з розробкою, виробництвом, контролем якості, постачанням, обслуговуванням тощо.

Співпраця та комунікація: Забезпечення спільної роботи між командами та стейкхолдерами, обмін даними, комунікація та спілкування через систему PLM.

Управління конфігурацією: Контроль конфігурацій виробів, їх варіантів, залежностей та відслідковування змін в різних версіях.

Аналіз та звітність: Надання аналітичних звітів та статистики щодо ефективності процесів, витрат ресурсів, використання часу тощо.

Інтеграція з іншими системами: Можливість взаємодіяти з ERP-системами (системами планування ресурсів підприємства), CAD/CAM системами, системами управління виробництвом та іншими внутрішніми та зовнішніми додатками.

PLM-системи дозволяють підвищити продуктивність, зменшити час розробки, забезпечити якість виробів, скоротити витрати та сприяють більш ефективній співпраці між командами та підрозділами виробничого підприємства.

Аналіз розвитку інформаційних технологій у виробничій сфері показує, що одним з напрямків такого розвитку є всебічний охват цими технологіями

різних етапів і стадій життєвого циклу (ЖЦ) виробів. Гнучкі виробничі системи вирішували завдання, що стосувалися винятково виробництва виробів. Основні етапи ЖЦ промислових виробів і типи автоматизованих систем, що використовуються у їх ЖЦ, представлені на рис. 5.

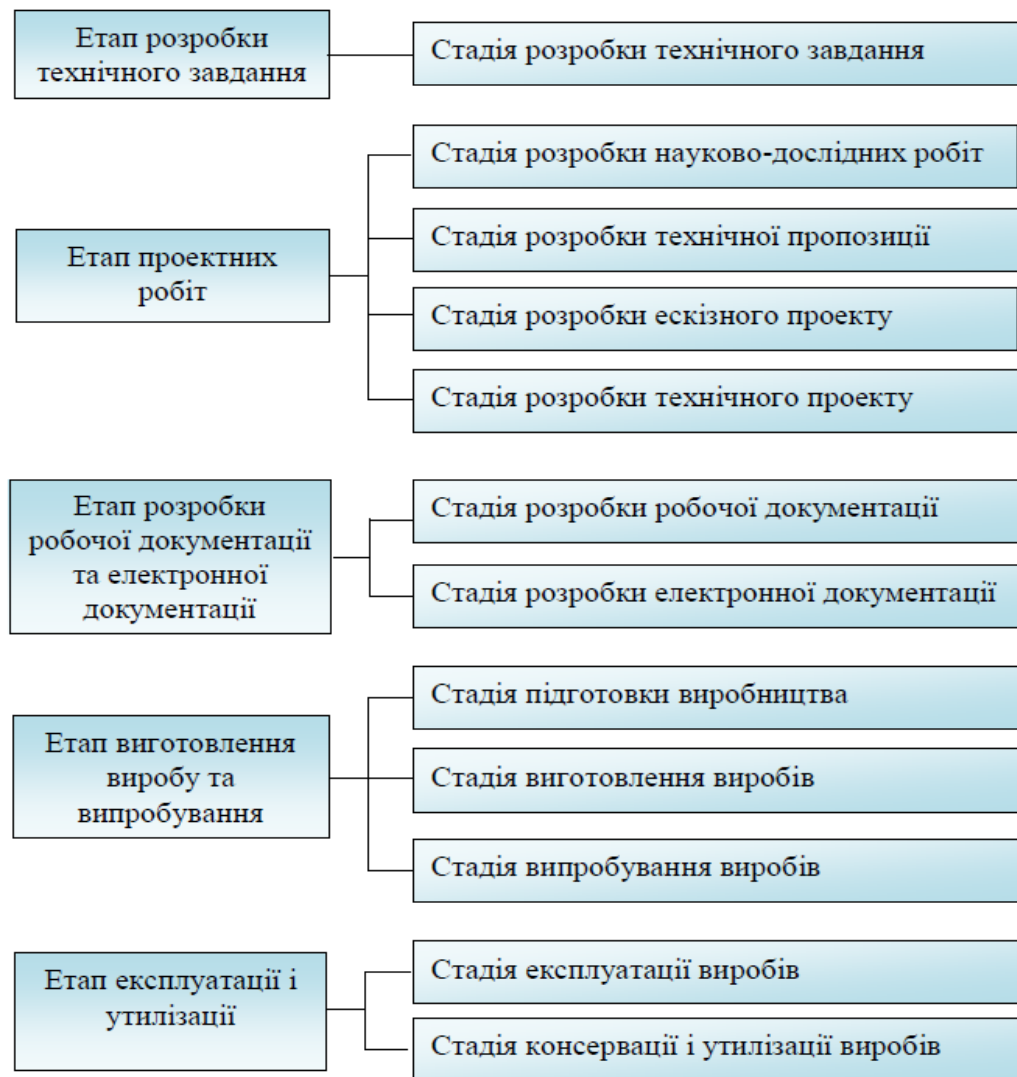


Рис.5. Життєвий цикл виробу

CAE (Computer Aided Engineering) - автоматизовані розрахунки й аналіз;
 CAD (Computer Aided Design) - автоматизоване проектування;
 CAM (Computer Aided Manufacturing) - автоматизована технологічна підготовка виробництва;
 PDM (Product Data Management) - управління проектними даними;

ERP (Enterprise Resource Planning) - планування й управління підприємством;

MRP II (Manufacturing Requirement Planning) - планування виробництва;

MES (Manufacturing Execution System) - виробнича виконавча система;

SCM (Supply Chain Management) - управління ланцюжками поставок;

CRM (Customer Relationship Management) - управління взаєминами із замовниками;

SCADA (Supervisory Control And Data Acquisition) - диспетчерське управління виробничими процесами;

CNC (Computer Numerical Control) - комп'ютерне числове управління;

S&SM (Sales and Service Management) - управління продажами й обслуговуванням;

PLM (Product Lifecycle Management) - систему управління життєвим циклом продукції.

Автоматизовані системи підтримують певні етапи та процедури (Рис.6).

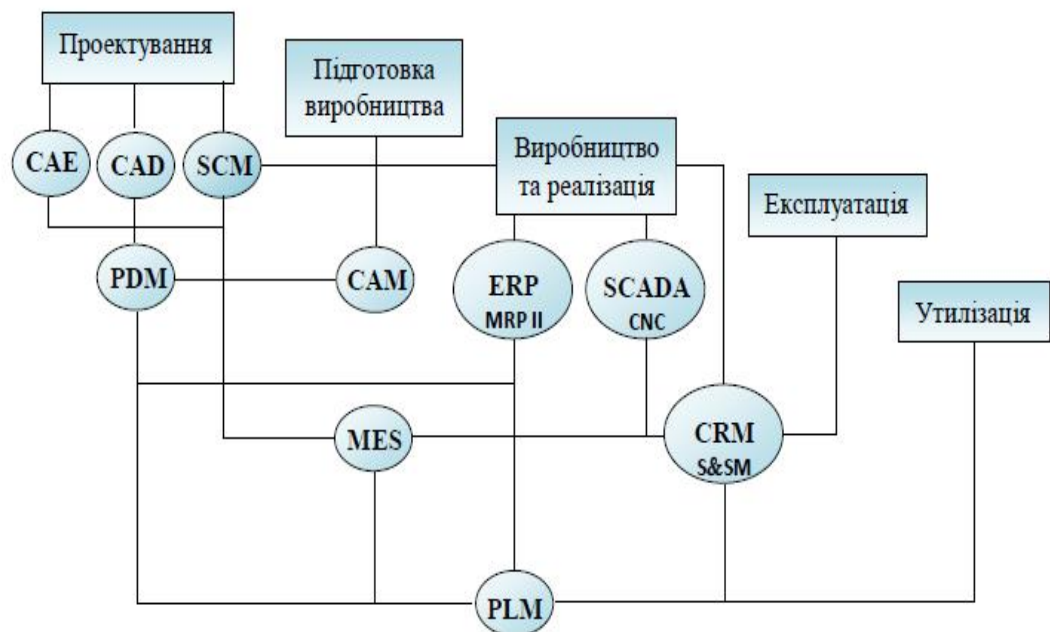


Рис.6. Етапи життєвого циклу виробів в системи їх автоматизації

Автоматизація підготовки виробництва дає можливість підприємствам швидко реагувати на зміну попиту, скорочувати терміни проектування і виробництва нової продукції, швидко модернізувати продукцію, що випускається, відслідковувати життєвий цикл виробів, ефективно підвищувати якість, знижувати собівартість продукції. На рис.7 зображено функціональну схему автоматизованого проектування, що дозволяє підвищити продуктивність праці проектувальників.



Рис.7. Функціональна схема автоматизованого проектування

1.4. Призначення та галузь застосування CAD-систем: постановка завдання для подальшої розробки

Запропоновано використовувати API інтерфейси для програмного комплексу SolidWorks, щоб підвищити продуктивність роботи конструкторів-проектувальників та скоротити терміни проектування.

Запропонований програмний модуль дозволить:

- Створити параметризований об'єкт;
- Підключити до програмного середовища SolidWorks, в якому необхідно здійснити модифікацію існуючої моделі;

- Завантажити модель в програмний модуль, при цьому у вікно програмного модуля завантажаться параметри моделі;
- Зробити корегування геометрії об'єкту (згідно нових параметрів креслення), який був завантажений;
- Зробити нове складання об'єкту та зберегти новий результат.

ВИСНОВКИ ДО РОЗДІЛУ 1

Аналіз предметної галузі показав, що найбільш прогресивною умовою удосконалення процесу проектування є впровадження в практику систем автоматизованого проектування, які забезпечені сучасним прикладним програмним забезпеченням. Основними критеріями ефективності впровадження програмного продукту є недопустимість виникнення помилок та збоїв при роботі. Також продукт має бути орієнтований на користувача і задовольняти його потреби, що пов'язані зі сферою роботи програми та вимогами зручності.

РОЗДІЛ 2. РОЗРОБКА ТА ПРОЕКТУВАННЯ САПР НА БАЗІ API

Application Programming Interface SolidWorks – це інтерфейс, який дозволяє програмувати додатки для системи SolidWorks. Даний інтерфейс містить в собі сотні функцій, які можна викликати із програм Microsoft Visual Basic, VBA (Microsoft Excel, Word, Access и т.д.), Microsoft Visual C, C++ або файлів-макросів SolidWorks [14, 19].

Програмний інтерфейс SolidWorks застосовує об'єктно-орієнтовну модель. Всі функції SolidWorks API – це методи або властивості, які використовують до об'єктів SolidWorks. Діаграма об'єктів SolidWorks представляє собою ієрархічну структуру, яка відображає зв'язки між об'єктами SolidWorks та показує яким чином можна отримати доступ до того чи іншого об'єкта SolidWorks API. Діаграма об'єктів SolidWorks зображена на рисунку 8.

Головним елементом дерева об'єктів є SLDWorks. Цей об'єкт може бути отриманим за допомогою методу CreateObject.

```
Set SldWorks = CreateObject("SldWorks.Application")
```

Метод CreateObject запускає програму SolidWorks, якщо вона не була запущена, або відтворює об'єкт SldWorks останньої сесії. Коли створюється об'єкт SldWorks можна отримати доступ до всіх об'єктів SolidWorks для поточної сесії. Одним із ключових об'єктів SolidWorks API є ModelDoc. Об'єкт дає розробнику інтерфейс до модифікації 3D моделей та креслеників SolidWorks. Об'єкт ModelDoc відповідає за функціональність деталей, збірок, креслеників.

Різновидами ModelDoc є об'єкти PartDoc, AssemblyDoc и DrawingDoc. Ці об'єкти забезпечують доступ до документів деталей, збірок відповідно. Всі методи та властивості об'єкта ModelDoc, більш застосовані до об'єктів PartDoc.

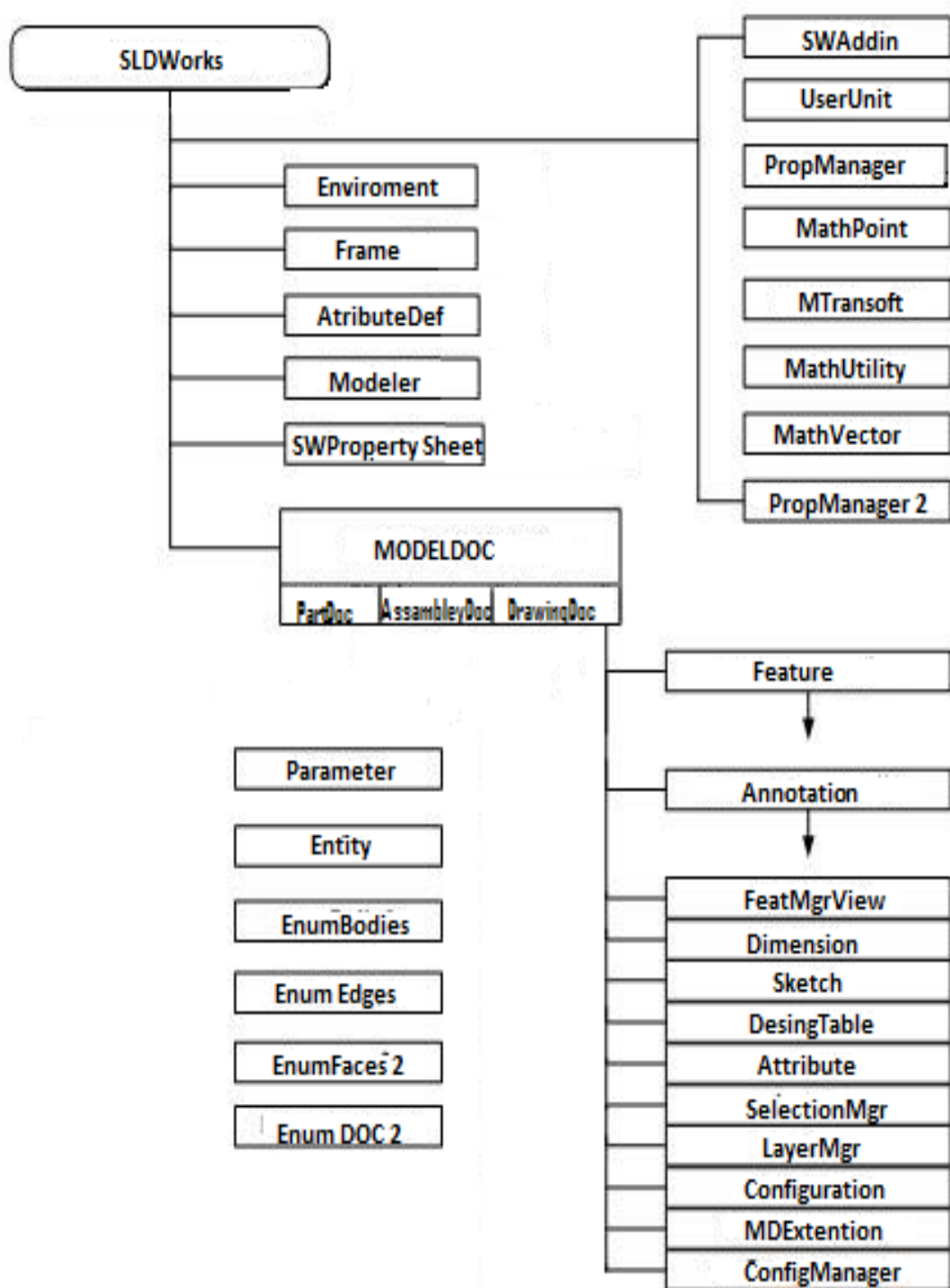


Рис.8. Схема API программного комплексу SolidWorks

Всі функції SolidWorks API підтримують COM інтерфейс. В додатках по API описується синтаксис Dispatch та відмінний від нього синтаксис COM, які можуть бути використані для виклику тій чи інший API функції.

Синтаксис Dispatch описується більш детально, чим синтаксис COM, в тих випадках коли аргументи чи повернені значення ідентичні для обох стилів програмування.

Також при програмуванні в стилі COM, у відмінності від Dispatch, функції SolidWorks API будуть повертати HRESULT і цілий ряд додаткових значень аргументів [6, 21].

2.1. Функціональне призначення програми

Як об'єкт впровадження розроблюваної САПР розглядається універсальне програмне забезпечення на базі мови програмування C# з використанням API для розширення бібліотек нових виробів в найбільш поширених системах автоматизованого проектування SolidWorks.

Призначення розробленого модулю:

- Збільшити бібліотеку нових виробів та вирішити завдання розрахунку і проектування конкретного класу виробів враховуючи специфіку підприємства;
- Підвищення продуктивності праці проектування, скорочення термінів проектування та зниження витрат на розробку технічної документації;
- Збільшення якості проектування за рахунок підвищення точності виготовлення виробів.

Структура розробленого програмного модуля:

- Клас CADet - головний клас, являє собою Windows Form, містить користувацький інтерфейс, виконує функцію менеджера функцій (підключитися до програми, створити об'єкт ProgramObject).
- Клас ProgramObject - управляючий батьківський загальний клас для SolidObject класу. Містить віртуальні методи, призначені для наслідування:

- Клас SolidObject (успадкований від ProgramObject, призначений для управління SolidWorks COM об'єктом).

SolidWorks використовує ядро Parasolid - це найшвидше ядро, доступне для ліцензування, розроблено UGS. Parasolid забезпечує технологію для твердотілого моделювання, узагальненого пористого моделювання, інтегровані поверхні вільної форми і листове моделювання [27].

Parasolid дозволяє розробникам швидко створювати конкурентоспроможні продукти використовуючи ці технології. На цьому ядрі розроблено багато CAD / CAM / CAE систем високого і середнього рівня. Parasolid підтримує SMP (багатопроесорне апаратне забезпечення), що дозволяє збільшити продуктивність. Parasolid включає більш ніж 600 об'єктно-орієнтованих функцій для додатків під управлінням Windows NT, UNIX, і LINUX.

Загальний алгоритм роботи модулю:

- Використовуючи системний метод CreateInstance () створюється COM об'єкт. В якості параметру виступає тип CAD системи, об'єкт якої створюється.
- Відкривається файл моделі, яку необхідно модифікувати відповідно до нових даних (креслення деталі).
- Отримуємо список параметрів за допомогою API функцій відповідної CAD-системи та записуємо їх у користувацький інтерфейс.
- Здійснюємо модифікацію тривимірної моделі встановлюючи нові параметри.
- Перебудовуємо модель натисненням кнопки «Перебудувати» та зберігаємо результат перебудови.

Ієрархія класів реалізованого програмного модуля «CADet» представлена на рисунку 9. Клас CADet - головний клас, являє собою Windows Form, містить користувацький інтерфейс, виконує функцію менеджера функцій - підключитися до програми, створити об'єкт ProgramObject.

Клас ProgramObject - управляючий батьківський загальний клас для SolidObject класу. Містить віртуальні методи, призначені для наслідування:

- Booleaninit () - ініціалізація COM об'єкта CAD-системи;
- VoidCloseAllInstance () - закрити всі існуючі процеси відповідної CAD-системи;
- VoidReBuild () - перебудувати поточну модель, з урахуванням змінених параметрів;
- VoidSetSettings () - встановити поточні значення параметрів в користувацький інтерфейс.

Клас SolidObject - успадкований від ProgramObject, призначений для управління SolidWorks COM об'єктом;



Рис.9. Ієрархія класів програмного модуля "CADet"

Архітектура проектування

Програмний комплекс SolidWorks використовують для робіт машинобудівної галузі підприємств на етапах конструкторсько-технологічної підготовки виробництва виробів різної ступені складності та призначення.

Спеціалізовані модулі програмного комплексу вирішують задачі на етапі виробництва та експлуатації. Складається із компонентів:

- Бібліотека проектування: єдина бібліотека фізичних властивостей матеріалів, текстур, штриховок. Типові конструктивні елементи, стандартні деталі та вузли, елементи листових деталей. Бібліотека стандартних отворів.
- Експрес-аналіз: динаміки механізмів, міцності деталей – SimulationXpress, гідродинаміки – FloXpress, технологічності виготовлення деталі – DFMXpress, лиття пластмас - e-SimpoeWorks. Розрахунок масово-інерційних та геометричних характеристик моделі, моделювання роботи кулачків.
- Експертні системи: SketchXpert - аналіз конфліктів в ескізах, пошук оптимального рішення. FeatureXpert, FilletXpert, DraftXpert – автоматичне керування елементами скруглень, оптимізація порядку створення моделі.
- Instant3D – динамічне пряме редагування 3D моделей деталей та збірок, стандартних компонентів. DimXpert – автоматизована постановка розмірів та допусків з 3D моделі, а також розмірів в кресленнях є можливість роботи з імпортованою геометрією.
- Оформлення креслеників по ЄСКД: Анімація 3D;
- API SDK підтримка програмування на мовах Visual Basic, Visual C++ та інші, запис та редагування макросів (VBA).
- SolidWorks Rx: Утиліта автоматичної діагностики комп'ютера на відповідність вимогам SolidWorks.
- SolidWorks Explorer: Провідник файлів SolidWorks.
- 3D Content Central: Доступ до єдиної бібліотеки стандартних компонентів виробників у форматі SolidWorks.
- SolidWorks eDrawings: Просмотр та висновки на печать документів SolidWorks, Pro/Engineer, DWG, DXF.

- DraftSight: Робота з креслениками у форматі DWG без обмежень номеру версії. Асоціативні зв'язки з моделлю SolidWorks.

2.2. Архітектура проектування API SolidWorks

Архітектура проектування API SolidWorks базується на об'єктному підході, оскільки SolidWorks API надає доступ до об'єктів моделей, складових, функцій та інших елементів програми SolidWorks.

Основні складові архітектури API SolidWorks включають:

Об'єктна модель: API SolidWorks використовує об'єктну модель для представлення різних елементів програми. Кожен елемент (модель, елемент керування, властивість тощо) представлений об'єктом з методами та властивостями для доступу та зміни.

COM (Component Object Model): SolidWorks API базується на технології COM, що дозволяє взаємодіяти з програмою через інтерфейси об'єктів. Це означає, що ви можете створювати, змінювати та управляти об'єктами SolidWorks через інтерфейси, які надаються API.

Модель скриптів та мови програмування: Ви можете використовувати різні мови програмування, такі як C#, VB.NET, VBA тощо для роботи з API SolidWorks. SolidWorks надає доступ до бібліотек для цих мов, що дозволяє створювати скрипти та програми для автоматизації рутинних задач, зміни моделей та ін.

Компоненти API: SolidWorks API містить різні компоненти для керування моделями, функціями, кресленнями, роботою з файлами, додатковими інструментами для маніпуляції та аналізу даних.

Розширення функціональності SolidWorks: API дозволяє розширювати базовий функціонал SolidWorks, додавати власні функції, інструменти та можливості для покращення робочого процесу.

Документація та приклади коду: Для вивчення та розуміння API SolidWorks доступна документація, яка містить опис об'єктів, методів, властивостей, а також приклади коду для різних операцій [7].

Розробка програмного модуля для CAD-систем за допомогою API загалом має типову структуру. Структура модуля, який включає в себе підтримку таких CAD-систем як SolidWorks, і написаний на мові C#, представлена на рис 10.

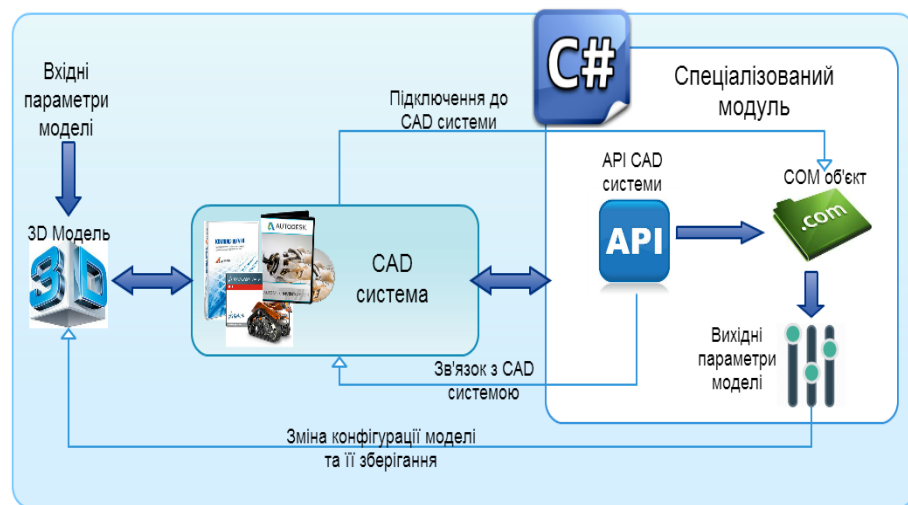


Рис.10. Схема зміни конфігурації 3D моделі

Робота з процесами CAD-систем здійснюється за допомогою COM об'єктів.

COM (Component Object Model - об'єктна модель компонентів) - це технологічний стандарт від компанії Microsoft, призначений для створення програмного забезпечення на основі взаємодіючих компонентів, кожен з яких може використовуватися в багатьох програмах одночасно. Стандарт втілює в собі ідеї поліморфізму і інкапсуляції об'єктно-орієнтованого програмування.

2.3. Інтерфейс програмування застосунків (API) SolidWorks

Інтерфейс програмування застосунків (API) SolidWorks - це набір інструментів, методів та властивостей, які дозволяють програмістам

взаємодіяти з програмою SolidWorks. Цей інтерфейс надає доступ до різних функцій, що дозволяють автоматизувати процеси створення, редагування та аналізу моделей CAD.

Основні складові інтерфейсу програмування API SolidWorks включають:

Об'єкти моделі: API надає доступ до об'єктів SolidWorks, таких як моделі, креслення, елементи керування, властивості, розміри, функції та інші елементи.

Методи та функції: API містить набір методів та функцій для виконання різних операцій над об'єктами SolidWorks, таких як створення/відкриття/закриття моделей, зміна розмірів, додавання функцій, геометрії тощо.

Події: API SolidWorks дозволяє відслідковувати події, що відбуваються в програмі, наприклад, зміни в моделі, активність користувача та інші.

Керування файлами: Інтерфейс API надає можливість працювати з файлами SolidWorks, зчитувати та зберігати їх, виконувати операції імпорту та експорту.

Робота з кресленнями: API має засоби для створення та редагування креслень, додавання вимірів, тексту, керування складовими креслення та інші.

Розширення функціоналу: Інтерфейс API SolidWorks дозволяє розширювати функціональність програми, додавати власні плагіни, макроси та інструменти [11].

2.4. Вибір інструментальних засобів для розробки API

Обираючи інструменти для розробки модуля API для SolidWorks або будь-якого іншого середовища, важливо враховувати кілька ключових факторів:

Мова програмування: Вибір мови програмування, з якої ви збираєтеся розробляти модуль, це перше рішення. C#, VB.NET, VBA або Python - це деякі з популярних варіантів для роботи з SolidWorks API.

Інтегроване середовище розробки (IDE): Використання IDE, яке підтримує вибрану мову програмування, є важливим. Наприклад, Visual Studio для C# або VB.NET, або PyCharm для Python.

Документація та підтримка: Оберіть інструменти з хорошою документацією та активною спільнотою користувачів. Для SolidWorks API це може бути офіційна документація SolidWorks API, форуми спільноти розробників тощо.

Зручність в роботі: Обирайте ті інструменти, з якими ви зручно працюєте та які відповідають потребам вашого проекту.

Щодо конкретних інструментів для розробки модулів API для SolidWorks:

Для C#: Visual Studio є дуже популярним інструментом. Ви можете використовувати SolidWorks API в поєднанні з .NET Framework для розробки модулів.

Для VBA: Вбудоване середовище VBA у SolidWorks може бути використане для простих скриптів та макросів.

Для Python: Є бібліотеки, такі як swiglpk, які надають можливість взаємодії з SolidWorks через Python. Також, бібліотека pywin32 дозволяє взаємодіяти з COM-об'єктами, що використовуються в SolidWorks.

Серед мов програмування найбільш широкий розвиток отримали універсальні мови, за допомогою яких зручно вирішувати як логічні завдання, так і обчислювальні. При цьому універсальна мова програмування повинна бути зручний для обробки текстів, зображень, файлів, а складена програма повинна добре зчитуватися і розумітися однозначно всіма програмістами, щоб її надалі можна було модифікувати.

C# найкращим чином відповідає всім цим потребам. C# – це комбінація кількох найважливіших технологій:

- високопродуктивний компілятор в машинний код;
- об'єктно – орієнтована модель компонентів;

Вибір оптимальної мови програмування, для створення API – інтерфейсу, здійснюємо по методу аналізу ієрархій опираючись на наступні критерії:

- сприйняття коду;
- функціональність;
- простота освоєння;
- вартість придбання та супроводу;
- інтерфейс;
- швидкість розробки рішення;
- швидкість роботи додатків.

Порядок застосування методу аналізу ієрархій:

- побудова якісної моделі проблеми у вигляді ієрархії, що включає мету, альтернативні варіанти досягнення цілі та критерії для оцінки якості альтернатив;
- визначення пріоритетів всіх елементів ієрархії з використанням методу парних порівнянь;
- синтез глобальних пріоритетів альтернатив шляхом лінійної згортки пріоритетів елементів на ієрархії;
- перевірка суджень на узгодженість;
- ухвалення рішення на основі отриманих результатів.

Більшість застосовуваних у промисловості тривимірних САПР можуть бути використані як основа для побудови спеціалізованої САПР, яка вирішує завдання розрахунку і проектування конкретного класу виробів. При цьому необхідно об'єднати розрахунковий модуль, який визначає розмірні та інші

2.5. Застосування API інтерфейсу для програмного комплексу SolidWorks

Для отримання можливості працювати з API SolidWorks, в проект необхідно додати посилання на бібліотеки - SolidWorks.Interop.sldworks.dll, SolidWorks.Interop.swcommands.dll та SolidWorks.Interop.swconst.dll (за замовчуванням для 64-розрядної операційної системи, ці бібліотеки перебувають в кореневій папці «C:\ProgramFiles\SolidWorksCorp\SolidWorks\Toolbox\toolboxconfigure», також їх можна завантажити з офіційного сайту).

Перед тим як створити новий процес SolidWorks, необхідно закрити всі поточні процеси SolidWorks (з метою оптимізації роботи):

```
Process[] processes = Process.GetProcessesByName("SLDWORKS");
foreach (Process proces in processes)
{
    proces.CloseMainWindow();
    proces.Kill();
}
```

Далі створюємо новий процес SolidWorks і наводимо тип COM об'єкта до типу SldWorks:

```
object v_processSW =
System.Activator.CreateInstance(System.Type.GetTypeFromCLSID(v_myGuid));
v_App = (SldWorks)v_processSW;
```

SldWorks - посилання на об'єкт SolidWorks, точка входу для використання API SolidWorks.

Створюємо новий 3D документ типу IModelDoc2 і відкриваємо 3D модель використовуючи метод OpenDoc об'єкта IModelDoc2:

```
IModelDoc2 v_Model;
v_Model = (IModelDoc2)v_App.OpenDoc(FilePath, 1);
```

При натисканні на кнопку «Перебудувати», викликається метод ReBuild (), отримуємо посилання на менеджер параметрів поточної моделі – EquationMgr:

```
EquationMgrswEquationMgr = default(EquationMgr);
swEquationMgr = (EquationMgr)v_Model.GetEquationMgr();
```

Далі, проходячи циклом з цієї колекції, змінюємо значення параметрів. Після зміни значень параметрів, необхідно перебудувати 3D документ:

```
Part.EditRebuild3();
```

Таким чином, за допомогою API SolidWorks, програмний модуль змінює значення параметрів 3D моделі. При необхідності, можна зберегти поточний результат або ще раз змінити значення параметрів і знову перебудувати модель.

Розроблений програмний модуль дозволяє вирішувати завдання розрахунку і проектування конкретного класу виробів за рахунок його об'єднання з вже наявним в САПР тривимірним геометричним ядром. Для цього спочатку створюється параметрична збірка спроектованого механізму, в якій ряд розмірів винесений в змінні моделі. Розрахунковий модуль (це зовнішній exe-файл або dll-бібліотека, що підключається до САПР) може розрахувати необхідні значення змінних моделі і автоматично змінити їх, в результаті чого буде отримано новий варіант 3D збірки. Таким чином, відразу ж після розрахунку буде отримана нова геометрія виробу. Контроль із зовнішньої програми здійснюється за допомогою API-інтерфейсів, які надають доступ до параметрів збірки.

Розроблений програмний модуль дозволяє під'єднатися до однієї з CAD-систем (SolidWorks); обрати необхідну модель, що потребує модифікації; здійснити модифікацію обраної моделі (програмний модуль зчитує поточні значення параметрів моделі та динамічно завантажує їх у інтерфейс, де користувач має змогу змінити їх та перебудувати модель); зберегти поточний

результат (модифіковану модель) при необхідності, або ще раз змінити значення параметрів і знову перебудувати модель.

Вхідні дані: створений параметризований об'єкт в якому ряд розмірів винесений в змінні моделі (3D моделі, що потребує модифікації);

Вихідні дані: модифікована тривимірна модель об'єкту.

Щоб скористатися послугами розроблено програмного модулю, необхідно запустити зовнішній exe-файл CADet.exe (рис.12) та потрапити до користувацького інтерфейсу.

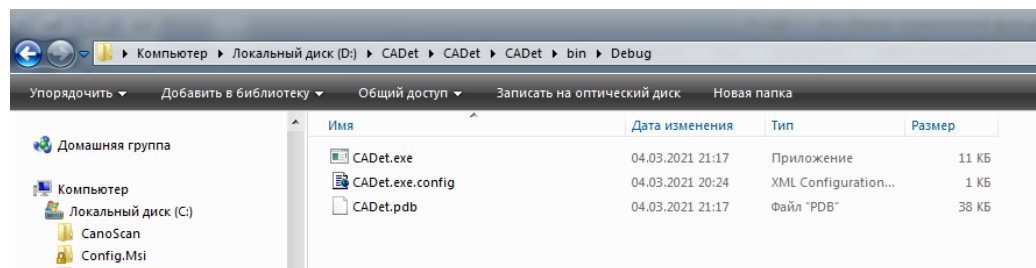


Рис.12. Виклик та завантаження програми

Інтерфейс користувача програмного модуля зображений на рис. 13.

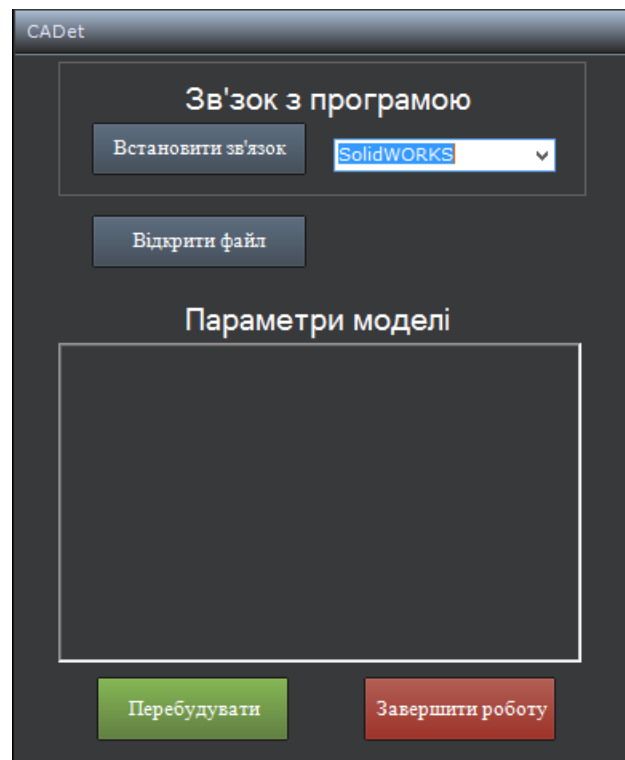


Рис.13. "CADet" - користувацький інтерфейс модуля САПР

Програмний модуль САПР істотно спрощує роботу проектувальника, скорочуючи час роботи над об'єктами і над проектами в цілому. Виходячи з цього, інструкція використання модуля досить проста. Розглянемо її на програмному комплексі SolidWorks:

1. Створення параметризованого об'єкту (побудова тривимірної моделі в одній із CAD-систем).
2. Підключення до програмного середовища, в якому необхідно модифікувати існуючу геометрію об'єкту SolidWorks. При успішному встановленні з'єднання з обраною зі списку CAD-системою з'явиться наступний значок .
3. Завантаження об'єкта в програмний модуль. При натисканні на кнопку "Відкрити файл" з'являється можливість обрання необхідної моделі. При цьому, у вікно програмного модуля динамічно завантажаться параметри моделі (рис.15).

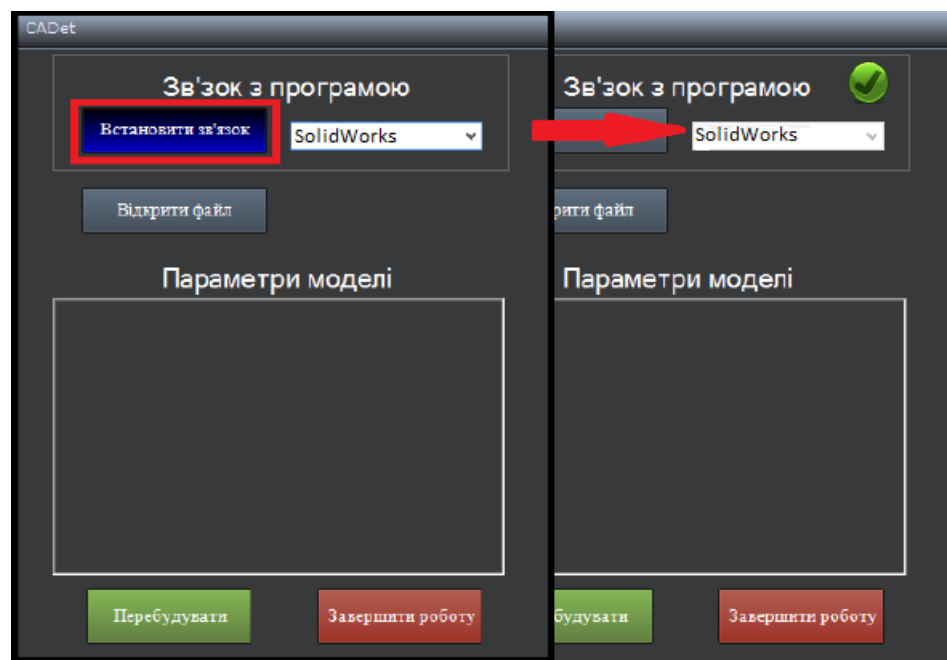


Рис.14. Підключення до CAD-системи

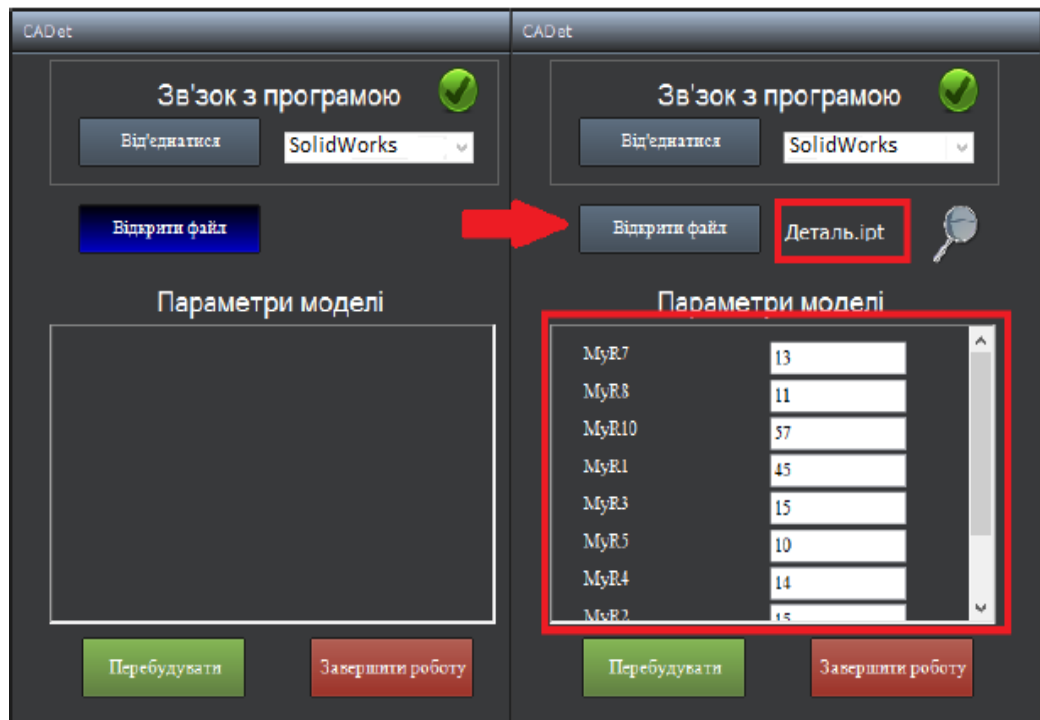


Рис.15. Завантаження об'єкта в програмний модуль

4. Тепер користувач має змогу переглянути модель (Рис.16.)

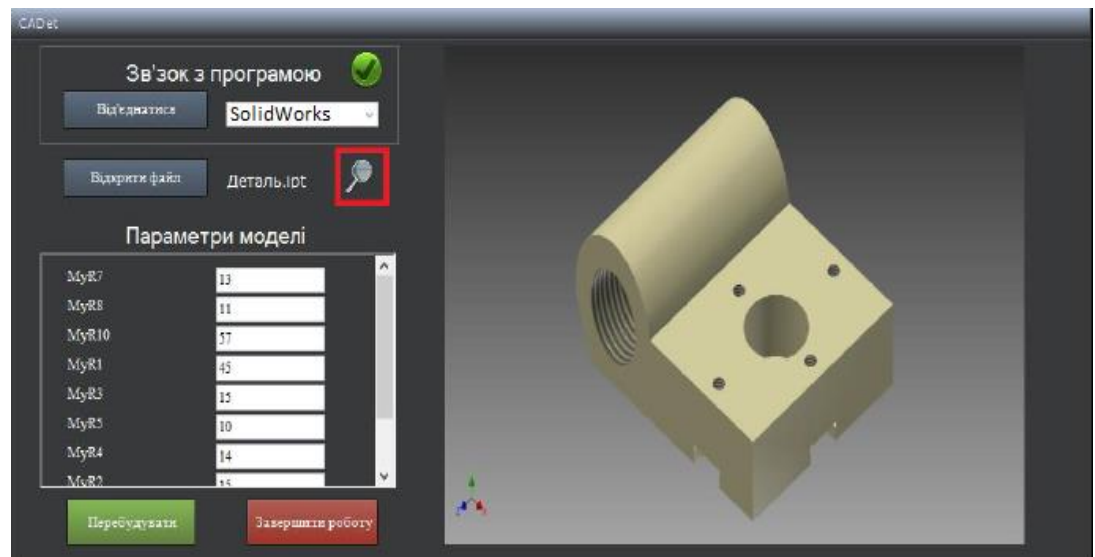


Рис.16. Швидкий перегляд моделі

5. Модифікація існуючої геометрії моделі (зміна параметрів моделі у відповідному вікні «Параметри моделі» згідно до заданого креслення).
6. Перебудова, збереження об'єкта.

Для демонстрації роботи програмного модуля було використано 3D-модель деталі «Корпус». Початковий вид даної деталі представлений на рис.17.

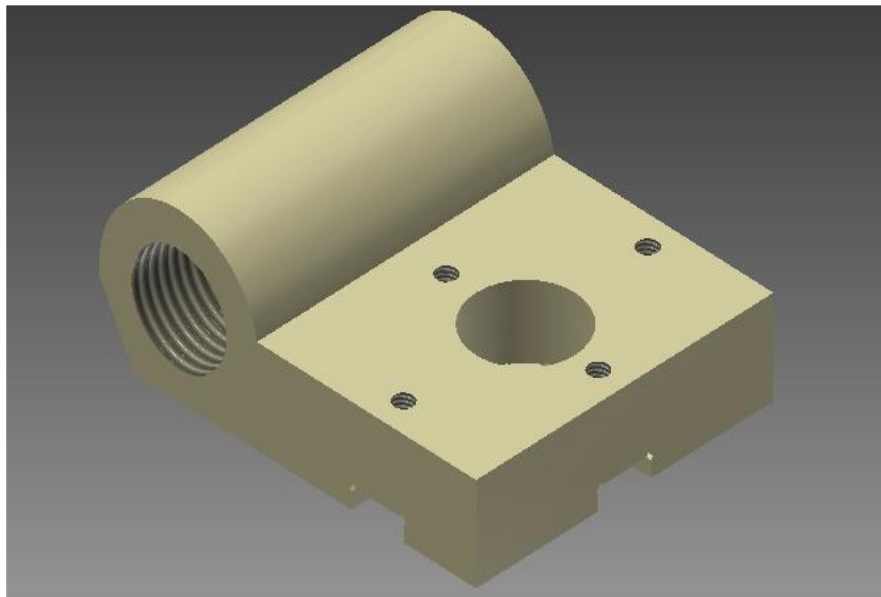


Рис.17. 3D-модель деталі "Корпус"

Після натискання кнопки «Перебудувати», отримуємо результат у відповідному програмному CAD середовищі, зі зміненими параметрами (Рис.18).

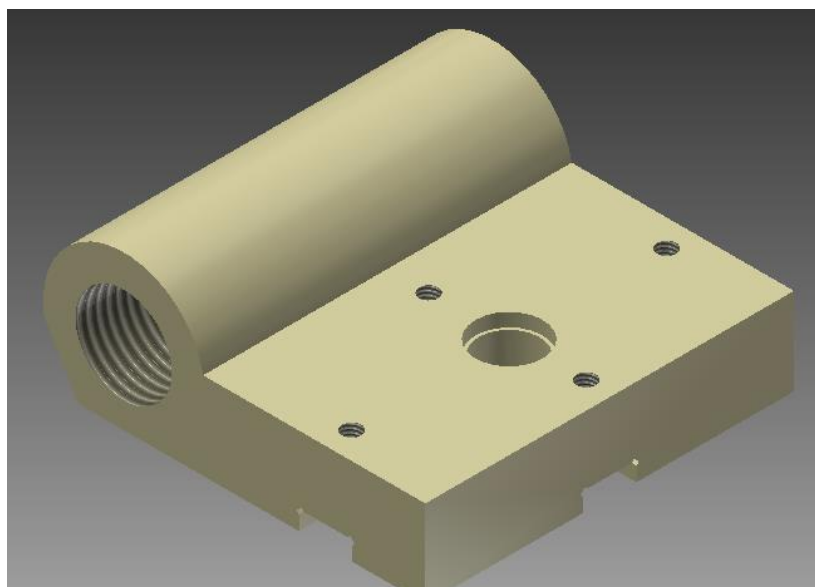


Рис.18. Модифікована 3D-модель деталі «Корпус»

Також, для прикладу, можна розглянути ще одну деталь до та після модифікації. Для цього знову потрібно запустити модуль API, завантажити потрібну деталь, та натиснути кнопку перебудувати (Рис.19).

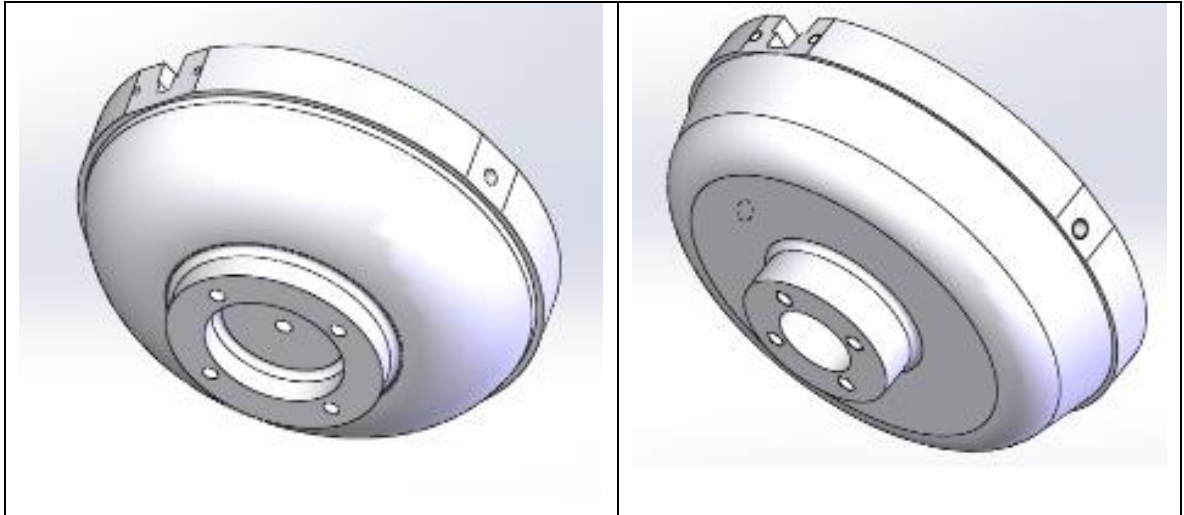


Рис. 19. Деталь до і після модифікації

ВИСНОВКИ ДО РОЗДІЛУ 2

Розроблений програмний модуль дозволяє вирішувати завдання розрахунку і проектування деталей виробів за рахунок його об'єднання з САПР тривимірним геометричним ядром. Розрахунковий модуль може розрахувати необхідні значення змінних моделі і автоматично змінити їх, що дає змогу отримати в результаті нову геометрію виробу. Контроль із зовнішньої програми здійснюється за допомогою API-інтерфейсів, які надають доступ до параметрів збірки. Можна зробити висновок, що автоматизація процесів проектування дозволить інженеру-проектувальнику збільшити швидкість редагування розмірів, що в результаті підвищить продуктивність праці конструктора та покращить якість продукту.

РОЗДІЛ 3 ЕКОНОМІЧНА СКЛАДОВА

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Початкові дані:

1. Передбачуване число операторів програми - 500;
2. Коефіцієнт складності програми (1,25...2,0) - 1,5;
3. Коефіцієнт корекції програми в ході її розробки (0,05...0,1) - 0,05;
4. Годинна заробітна плата програміста - 84грн/год;
5. Коефіцієнт збільшення витрат праці в наслідок недостатнього опису задачі (1,2...1,5) - 1,2;
6. Коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності (від 3-хдо 5 років) – 1,1;
7. Вартість машино-години ЕОМ - 14грн/год.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки. Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_0 + t_u + t_a + t_n + t_{отл} + t_d \quad (1)$$

де t_0 – витрати праці на підготовку та опис поставленої задачі (приймається 50 годин);

t_a – витрати праці на дослідження алгоритму рішення задачі;

t_n – витрати праці на програмування по готовій блок-схемі;

$t_{отл}$ – витрати праці на налагодження програми на ЕОМ;

t_d – витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у програмному забезпеченні, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C \cdot (1 + p) \quad (2)$$

де q - передбачуване число операторів програми ($q=500$);

C - коефіцієнт складності програми ($C = 1,5$);

p - коефіцієнт корекції програми в ході її розробки ($p = 0,05$).

Звідси умовне число операторів в програмі:

$$Q = 500 \cdot 1,5 \cdot (1 + 0,05) = 787,5 \quad (3)$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \cdot B}{(75 \dots 85) \cdot k}, \quad (4)$$

де B - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі;

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності. При стажі роботи від 3 до 5 років він складає 1,1.

Приймемо збільшення витрат праці в наслідок недостатнього опису завдання не більше 50% ($B = 1,2$). З урахуванням коефіцієнта кваліфікації $k = 1,1$, отримуємо витрати праці на вивчення опису завдання:

$$t_u = \frac{(787,5 \cdot 1,2)}{75 \cdot 1,1} = 11,45 \quad (5)$$

Витрати праці на розробку алгоритму рішення задачі визначаються за формулою:

$$t_a = \frac{Q}{(20 \dots 25) \cdot k} \quad (6)$$

де Q – умовне число операторів програми;

k – коефіцієнт кваліфікації програміста.

Підставивши відповідні значення в формулу (3.4), отримаємо:

$$t_a = \frac{787,5}{20 \cdot 1,1} = 35,79 \quad (7)$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20 \dots 25) \cdot k} \quad (8)$$

$$t_n = 787,5 / (25 \cdot 1,1) = 28,63$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{\text{отп}} = \frac{q}{(4..5) \cdot k} \quad (9)$$

$$t_{\text{отл}} = 787,5 / (5 \cdot 1,1) = 143,18$$

- за умови комплексного налагодження завдання:

$$t_{\text{отл}}^k = 1,5 \cdot t_{\text{отл}} \quad (10)$$

$$t_{\text{отл}}^k = 1,5 \cdot 143,18 = 214,77$$

Витрати праці на підготовку документації визначаються за формулою:

$$t_{\text{д}} = t_{\text{др}} + t_{\text{до}} \quad (11)$$

де $t_{\text{др}}$ - трудомісткість підготовки матеріалів і рукопису:

$$t_{\text{др}} = \frac{q}{(15..20) \cdot k} \quad (12)$$

$t_{\text{до}}$ - трудомісткість редагування, печатки й оформлення документації:

$$t_{\text{до}} = 0,75 \cdot t_{\text{др}} \quad (13)$$

Підставляючи відповідні значення, отримаємо:

$$t_{\text{др}} = 787,5 / (18 \cdot 1,1) = 39,77 \text{ годин.}$$

$$t_{\text{до}} = 0,75 \cdot 39,77 = 29,82 \text{ годин.}$$

$$t_{\text{д}} = 39,77 + 29,82 = 69,59 \text{ годин.}$$

Повертаючись до формули (1), отримаємо повну оцінку трудомісткості розробки програмного забезпечення:

$$t = 50 + 11,45 + 35,79 + 28,63 + 143,18 + 69,59 = 338,64 \text{ годин.}$$

3.2. Рахунок витрат на створення програми

Витрати на створення ПЗ $K_{\text{по}}$ включають витрати на заробітну плату виконавця програми $З_{\text{п}}$ і витрати машинного часу, необхідного на налагодження програми на ЕОМ:

$$K_{\text{по}} = З_{\text{п}} + З_{\text{мв}}, \text{ грн} \quad (14)$$

Заробітна плата виконавців визначається за формулою:

$$З_{\text{п}} = t \cdot C_{\text{пр}}, \quad (15)$$

де: t - загальна трудомісткість, годин;

$C_{\text{пр}}$ - середня годинна заробітна плата програміста, грн/година

З урахуванням того, що середня годинна зарплата становить 60грн / год, отримуємо:

$$З_{\text{зп}} = 338,64 \cdot 60 = 20318,4 \text{ грн} \quad (16)$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ, визначається за формулою:

$$З_{\text{мв}} = t_{\text{отл}} \cdot C_{\text{мч}}, \text{ грн} \quad (17)$$

де $t_{\text{отл}}$ - трудомісткість налагодження програми на ЕОМ, год;

$C_{\text{мч}}$ - вартість машино-години ЕОМ, грн/год (14 грн/год).

Підставивши в формулу (17) відповідні значення, визначимо вартість необхідного для налагодження машинного часу:

$$З_{\text{мв}} = 143,18 \cdot 14 = 2004,52 \text{ грн.}$$

Звідки витрати на створення програмного продукту:

$$K_{\text{по}} = 20318,4 + 2004,52 = 22322,92 \text{ грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p}, \text{ міс} \quad (18)$$

де B_k - число виконавців (дорівнює 1);

F_p - місячний фонд робочого часу (при 40 годинному робочому тижні $F_p = 176$ годин).

Звідки витрати на створення програмного продукту:

$$T = 338,64 / 1 \cdot 176 \sim 2 \text{ міс.}$$

ВИСНОВКИ ДО РОЗДІЛУ 3

Очікуваний час розробки становить 2 місяці. Цей термін пов'язаний з числом операторів, і включає час на дослідження і розробку алгоритму вирішення поставленого завдання, програмування по готовому алгоритму, налагодження програми і підготовку документації.

ВИСНОВКИ

Можливість легкої модифікації параметрів деталі суттєво спрощує роботу конструктора і призводить до суттєвого скорочення часу роботи по підготовці документації для випуску нової продукції. В цих цілях розроблюється модуль API, який має у своєму складі усі зовнішні змінні з їх значеннями і дозволяє користувачу за допомогою присвоєння їм нових значень розмірів автоматично змінювати деталь в цілому, без необхідності звертатися до креслень, розрахунків і перебудовувати модель в SolidWorks.

Розроблений програмний модуль дозволить підвищити продуктивність праці проєктувальників (за рахунок звільнення проєктувальника від проведення багатьох «рутинних» операцій), скоротити терміни проєктування та знизити витрати на розробку технічної документації, збільшити якість проєктування за рахунок підвищення точності виготовлення виробів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Спірінцев В.В. Розробка функціональної схеми процесу автоматизованого проектування / В.В. Спірінцев, І.В. Пихтєєва, Ю.О. Дмитрієв // Регіональний міжвузівський збірник наукових праць. Дніпропетровськ: Системні технології. – 2013. Випуск 1(84). С.129–135.
2. Спірінцев В.В. Розробка спеціалізованого програмного модуля для проектування типових деталей / В.В.Спірінцев // Сучасні проблеми геометричного моделювання // Збірник праць XIV Міжнародної науково-практичної конференції. Мелітополь, ТДАТУ. 2012. С.103–107.
3. Paul J. Schilling, Randy H. Shih. Parametric Modeling with SOLIDWORKS 2022. 2022, 610 p.
4. James D. Bethune. Engineering Design and Graphics with SolidWorks 2016. 2016, 829 p.
5. Zeid I. Mastering SolidWorks: the design approach. 2014, 552 p.
6. Загоруйко М.В. Використання CAD/CAE-систем для підготовки спеціалістів / М.В. Загоруйко, Л.В. Боровик, О.Ю. Рудик // Сучасні та історичні проблеми фундаментальної та прикладної математичної підготовки у закладах вищої освіти: погляд здобувачів вищої освіти і молодих вчених. Харків: ХНАДУ, 2019. С. 97-100.
7. Боровик Л.В. Забезпечення проектування додатками SolidWorks / Л.В. Боровик, О.Ю. Рудик, В.В. Педосюк // Інформаційні технології в металургії та машинобудуванні. ІТММ'2019: тези доповідей міжнародної науково-практичної конференції (Дніпро, 26 – 28 березня 2019 р.). Міністерство освіти і науки України, Національна металургійна академія України, Дніпропетровський національний університет імені О. Гончара, Дніпропетровський національний університет залізничного транспорту імені академіка В. Лазаряна та ін. Дніпро: НМетАУ, 2019. С. 137.

8. Андрощук М.В. Використання SolidWorks для прогнозування міцності матеріалів / М.В. Андрощук, О.В. Диха, О.Ю. Рудик // Перспективні технології на основі новітніх фізико-матеріалознавчих досліджень та комп'ютерного конструювання матеріалів. XII Міжнародна конференція студентів, аспірантів та молодих вчених. Збірка тез доповідей. 18 – 19 квітня 2019 р., Київ, Україна / Укладач Г.Г. Лобачова. К: «КПІ імені Ігоря Сікорського», С. 214-215.

9. Боровик О.В. Впровадження у навчальний процес CAE/CAD-систем / О.В. Боровик, О.Ю. Рудик, О.В. Дідух // Матеріали III Всеукраїнської науковопрактичної Інтернет-конференції «Ресурсно-орієнтоване навчання в «3D»: доступність, діалог, динаміка» / укл. Н.В. Кононець, В.О. Балюк. Полтава: КУЕП ПДАА, 2019. С. 128-132.

10. Калівода Я. Використання автоматизованих систем при проектуванні та випробуванні рухомого складу / Я. Калівода, Л. Недужа, О. Очкасов, Д. Черняєв // Матеріали 78 Міжнародної науково-практичної конференції «Проблеми та перспективи розвитку залізничного транспорту» 17- 18 травня 2018 року, Днепр, 2018. С. 17-19.

11. Darin Grosser. The Future of SolidWorks Has 'Always' Been. 2011. [Електронний ресурс] Ре-жим доступу: <http://blog.dasisolutions.com/2011/09/27/the-future-of-solidworks-has-always-been-in-your-hands/>

12. Bogenschneider B. Die betriebliche Erprobung des Oliflok-verfahrens bei der Ruhrkohle A. G. / B. Bogenschneider, W. Erdman // Aufbereitungs – Technik. 1981. № 4. р. 188-198.

13. Водяницький І.О., Дереза О.О. Моделювання механічних передач. Технічне забезпечення інноваційних технологій в агропромисловому комплексі : Матеріали I Міжнар. наук.-практ. Інтернет-конференції, м. Мелітополь, 01– 24 квітня 2020 р. / ред. кол. В.М. Кюрчев та ін. Мелітополь : ТДАТУ, 2020. С. 83–87.

14. Проців В.В., Зіборов К.А., Твердохліб О.М. Проектування редукторів із використанням САПР КОМПАС. Національний гірничий університет, 2011. С. 178.
15. Кузяєв Д.Р. Вибір та аналіз середовища розробки 3D-моделей. Електронний ресурс:
URL:https://openarchive.nure.ua/bitstream/document/14157/3/Sbornik_%D0%90DED_2019_2_Kuzaev.pdf
16. Дереза О.О., Коломієць С.М., Дереза С.В. Тривимірне моделювання деталей машин в машинобудуванні. Сучасні наукові дослідження на шляху до євроінтеграції : матеріали міжнародного науково-практичного форуму, м. Мелітополь, 21–22 червня 2019 р. Мелітополь : ФОП Однорог Т.В., 2019. Ч. 2. С. 29–31.
17. Gibson I., Rosen D., Stucker B. Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing. 2nd ed. Springer; 2015: 498 p. DOI: 10.1007/978-1-4939-2113-3.
18. Longhitano G.A., Larosa M.A., Munhoza A.L.J., Carvalho Zavagliaa C.A., Ierardia M.C.F. Surface finishes for Ti-6Al-4V alloy produced by direct metal laser sintering. Materials Research. 2015. Vol. 18(4). P. 838–842.
19. Kelly J.F. 3D-Printing: Build Your Own 3D-Printer and Print Your Own 3D-Objects. Que Publishing. 2013. 192 p.
20. Петров О.В. Аналіз гідравлічних втрат тиску в гідророзподільнику за допомогою CAD/CAE систем / О.В. Петров // Вісник Хмельницького національного університету. Серія технічні науки. 2009. №4. С.67-70.
21. Burennikov Yu. A. Metrological characteristic of the test rig with automatic regestering of the proportionallycontrolled hydraulic drive / Yu.A. Burennikov, L.G. Kozlov, D.O. Lozinsky, O.V. Petrov and ect. // Buletinul institutului politehnic din iasi. Tomul LV (LIX), fasc. 1, 2009. P.125-130.
22. Петров О.В. Застосування комп'ютерного моделювання гідродинамічних процесів для визначення втрат тиску у гідросистемах транспортних

засобів / О.В. Петров, П.О. Печенкін, О.О. Павлюк // Сучасні технології та перспективи розвитку автомобільного транспорту: IV міжнародна науково-практична конференція, 25-26 жовтня 2011 року. Вінниця, 2011. С. 63.

23. Brianza, G.: Gerardo: parametric geometric modeling of a spur gear using solidworks. GEARSolutions, 2017, pp. 33-36.

24. Shigley, J.E.: Shigley's Mechanical Engineering Design. McGraw-Hill Primis, New York (2006) <https://doi.org/10.1007/s00707-006-0284-x>

25. Spitas, V., Spitas, C.: Numerical and experimental comparative study of strength-optimised AGMA and FZG spur gears. Acta Mech. 193, 113–126 (2007). <https://doi.org/10.1007/s00707-006-0384-x>

26. Conklin, R.S.: Robust Involute Spur Gear Generator Solidworks. Rochester Automated Systems, Rochester (2018). <https://bit.ly/3cUf5gW>

27. Moro, C., Stromberga, Z., Stirling, A.: Virtualisation devices for student learning: Comparison between desktop-based (Oculus Rift) and mobile-based (Gear VR) virtual reality in medical and health science education. Australas. J. Educ. Technol. 33, 1–10 (2017)

28. Kayur Patel. 2010. Lowering the barrier to applying machine learning. In Adjunct proceedings of the 23rd annual ACM symposium on User interface software and technology - UIST '10. ACM Press, p. 355-358.

29. Mona Rahimi, Jin L.C. Guo, Sahar Kokaly, and Marsha Chechik. 2019. Toward Requirements Specification for Machine-Learned Components. In 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW). IEEE, p. 241-244.

ДОДАТКИ

Лістинг коду

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Diagnostics;
using System.Threading;

namespace SolidTest
{
    public partial class Form1 : Form
    {
        CADet.ProgramObject App = new CADet.ProgramObject();
        Boolean NeedProgramOpen = true;
        Boolean OpenPreview = true;
        String fileFormat = "";
        public Form1()
        {
            InitializeComponent();
            ProgramComboBox.SelectedIndex = 1;
        }

        private void ReBuild_Click(object sender, EventArgs e)
        {
            App.ReBuild();
        }

        private void GetProgram_Click(object sender, EventArgs e)
        {
            if (NeedProgramOpen)
            {
                switch (ProgramComboBox.Text)
                {
                    case "SolidWORKS":
                        App = new CADet.SolidObject();
                        fileFormat = "SolidWorks model|*.SLDPRT";

```

```

        break;
    default:
        MessageBox.Show("NotFound");
        return;
    }
    App.RootForm = this;
    if (App.init())
    {
        ProgramComboBox.Enabled = false;
        ConnectDoneIMG.Visible = true;
        MainPanel.Enabled = true;
        GetProgram.Text = "Від'єднатися";
        FilePanel.Enabled = true;
        NeedProgramOpen = false;
    }
    }else
    {
        if (App != null)
        {
            App.CloseAllInstance();
        }
        FileName.Text = "";
        ReBuild.Enabled = false;
        ProgramComboBox.Enabled = true;
        ConnectDoneIMG.Visible = false;
        MainPanel.Enabled = false;
        GetProgram.Text = "Встановити зв'язок";
        FilePanel.Enabled = false;
        NeedProgramOpen = true;
        FilePreview.Visible = false;
        ParamsPanel.Controls.Clear();
    }

}

private void close_Click(object sender, EventArgs e)
{
    this.Close();
}

private void pictureBox1_Click(object sender, EventArgs e)
{
    if (OpenPreview)
    {

```

```

        this.FilePreview.Width = 30;
        this.FilePreview.Height = 30;
        this.Width = 1000;
        OpenPreview = false;
    }
    else
    {
        this.FilePreview.Width = 40;
        this.FilePreview.Height = 40;
        this.Width = 391;
        OpenPreview = true;
    }
}

private void btnFileOpen_Click(object sender, EventArgs e)
{
    using (OpenFileDialog dialog = new OpenFileDialog())
    {
        dialog.Filter = fileFormat;

        if (dialog.ShowDialog() == DialogResult.OK)
        {
            App.OpenFile(dialog.FileName); App.SavePictForPreview();
            try {
                PreviewPicture.ImageLocation = @"F:\temp.jpg";
            }
            catch(Exception){}
            FilePreview.Visible = true;
            FileName.Text = dialog.SafeFileName;

            MainPanel.Enabled = true;
            ReBuild.Enabled = true;
            App.SetSettings();
        }
    }
}

```

Лістинг коду головного об'єкта

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Inventor;
using System.ComponentModel;
using System.Data;
using System.Drawing;

namespace CADet
{
    public class ProgramObject
    {

        public Form RootForm;
        public Control MainPanel;

        protected Dictionary<string, string> ParamsList = new Dictionary<string, string>();
        protected Dictionary<string, System.Windows.Forms.TextBox> ParamsListOnPanel = new
Dictionary<string, System.Windows.Forms.TextBox>();
        protected string ParamsSearchSuffix = "My";

        protected string FilePath;

        //Settings
        public ProgramObject() {

        }

        //Public methods

        public virtual void CloseAllInstance() { }
        public virtual void ReBuild() { }
        public virtual void GetParamsDictionary() { }
        public virtual void SavePicterForPreview() { }

        public Boolean init()
        {
            preInit();

```

```

        return initApp();
    }
    public virtual Boolean initApp()
    {
        return false;
    }
    public virtual void SetSettings()
    {
        ParamsList.Clear();
        ParamsListOnPanel.Clear();
        GetParamsDictionary();
        AppendParamsIntoForm();
    }
    public virtual void OpenFile(string filePath)
    {
        FilePath = filePath;
        OpenFileByPath();
    }

    protected virtual void OpenFileByPath() { }

    //private methods
    private void preInit()
    {
        MainPanel = RootForm.Controls["MainPanel"].Controls["ParamsPanel"];
    }
    private void AppendParamsIntoForm() {
        if (ParamsList.Count == 0) return;

        var v_i = 0;

        foreach (KeyValuePair<string, string> v_param in ParamsList)
        {
            AddParamBlock(v_param, v_i);
            v_i++;
        }
    }
    private void AddParamBlock(KeyValuePair<string, string> v_param, int v_index)
    {
        int v_shift = 10;
        Panel v_panel = new Panel();
        v_panel.Location = new System.Drawing.Point(10, v_index * 25 + v_shift);
    }

```

```

v_panel.Width = 280;
v_panel.Height = 25;
MainPanel.Controls.Add(v_panel);

    AddLabel(v_param.Key, v_panel);
    ParamsListOnPanel.Add(v_param.Key, AddTextBox(v_param.Value, v_panel));
}
private void AddLabel(string v_value, Panel v_panel)
{
    Label v_label = new Label();
    v_label.Text = v_value;

    v_label.Location = new System.Drawing.Point(10, 0);
    v_panel.Controls.Add(v_label);
}
private System.Windows.Forms.TextBox AddTextBox(string v_value, Panel v_panel)
{
    System.Windows.Forms.TextBox v_textBox = new System.Windows.Forms.TextBox();
    v_textBox.Text = v_value;
    v_textBox.Width = 100;
    v_textBox.Location = new System.Drawing.Point(150, 0);
    v_panel.Controls.Add(v_textBox);
    return v_textBox;
}
}
}

```

Лістинг коду об'єкта SolidObject

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SolidWorks.Interop.sldworks;
using SolidWorks.Interop.swcommands;
using SolidWorks.Interop.swconst;
using System.Diagnostics;
using System.Threading;

namespace CADet
{
    class SolidObject : ProgramObject
    {
        private SldWorks v_App; //приложение
        private IModelDoc2 v_Model; //чертеж
        public override Boolean initApp()
        {
            CloseAllInstance();
            try
            {
                //создаем приложение
                Guid v_myGuid = new Guid("0D825E02-9000-4D82-B4AB-D6BDC2872797");

                object v_processSW =
System.Activator.CreateInstance(System.Type.GetTypeFromCLSID(v_myGuid));
                v_App = (SldWorks)v_processSW;
                Thread.Sleep(5000);
            }
            catch (InvalidCastException v_e)
            {
                return false;
            }

            return true;
        }

        public override void CloseAllInstance()
        {
            //убиваем процесс

```

```

Process[] processes = Process.GetProcessesByName("SLDWORKS");
foreach (Process proces in processes)
{
    proces.CloseMainWindow();
    proces.Kill();
}
}

public override void ReBuild() {

    IModelDoc2 Part = v_App.IActiveDoc2;
    EquationMgr swEquationMgr = default(EquationMgr);
    swEquationMgr = (EquationMgr)v_Model.GetEquationMgr();

    int v_count = swEquationMgr.GetCount();
    for (var v_i = 0; v_i < v_count; v_i++)
    {
        string v_param = swEquationMgr.get_Equation(v_i);
        string v_paramName = v_param.Split(new Char[] { '=' })[0];
        if (v_paramName.Contains(ParamsSearchSuffix))
        {
            swEquationMgr.Add2(v_i, v_paramName + " =" +
Convert.ToDouble(ParamsListOnPanel[v_paramName].Text).ToString() + "mm", false);
            swEquationMgr.EvaluateAll();
            swEquationMgr.Delete(v_i + 1);
        };
    }
    Part.EditRebuild3();
    return;
}

public override void GetParamsDictionary()
{
    IModelDoc2 Part = v_App.IActiveDoc2;

    EquationMgr swEquationMgr = default(EquationMgr);
    swEquationMgr = (EquationMgr)v_Model.GetEquationMgr();

    for (var v_i = 0; v_i < swEquationMgr.GetCount(); v_i++)
    {
        string v_param = swEquationMgr.get_Equation(v_i);
        string v_paramName = v_param.Split(new Char[] { '=' })[0];
        if (v_paramName.Contains(ParamsSearchSuffix))
        {
            ParamsList.Add(v_paramName, swEquationMgr.Value[v_i].ToString());
        };
    }
}

```

```
    }  
}  
public override void SavePictForPreview()  
{  
}  
protected override void OpenFileByPath()  
{  
    v_App.Visible = true;  
    v_Model = (IModelDoc2)v_App.OpenDoc(FilePath, 1);  
    v_Model = v_App.IActiveDoc2;  
}  
}  
}
```