МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ НЕПЕРЕРВНОЇ ОСВІТИ
КАФЕДРА ТЕХНОЛОГІЙ УПРАВЛІННЯ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри

_____к.е.н., доц. О.В. Поліщук
«_____» _____20____р.

ДИПЛОМНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

ЗА СПЕЦІАЛІЗАЦІЄЮ «УПРАВЛІННЯ ПРОЕКТАМИ»

Тема: «УПРАВЛІННЯ ПРОЄКТОМ РОЗРОБКИ ПОВНОФУНКЦІОНАЛЬНОГО АВІАСИМУЛЯТОРА АВАРІЙНИХ СИТУАЦІЙ ДЛЯ ПРОФЕСІЙНОГО ВІДБОРУ МАЙБУТНІХ ПІЛОТІВ»

Виконавець:
студент _____ ___Є.О. Воловін___
групи УП-201Мз (підпис) (прізвище, ім'я, по-батькові)

Керівник: _____ ____Ю.М. Тесля_____
д.т.н., проф. (підпис) (прізвище, ім'я, по-батькові)

Нормоконтролер: _____ ____В.В. Дубініна___
(підпис) (П.І.Б.)

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут неперервної освіти
Кафедра технологій управління
Спеціальність: 073 «Менеджмент»
Спеціалізація: «Управління проектами»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____к.е.н., доц. О.В. Поліщук
«21» серпня 2020 р.

ЗАВДАННЯ
на виконання дипломної роботи (проекту)
_____Воловіна Євгенія Олексійовича_____
(П.І.Б. випускника)

1. Тема роботи «Управління проєктом розробки повнофункціонального авіасимулятора аварійних ситуацій для професійного відбору майбутніх пілотів» затверджена наказом ректора №/ст від

2. Термін виконання роботи: з 12 жовтня 2020 р. по 24 грудня 2020 р.

3. Вихідні дані роботи: забезпечення льотних навчальних закладів програмним інструментом оцінювання психофізичних якостей абітурієнтів для відбору тих людей, хто найбільше підходить на професію пілота.

4. Зміст пояснювальної записки:

1. Аналіз і оцінка предмету дослідження.

2. Особливості управління проєктом розробки програмної системи.

3. Реалізація управління проєктом розробки програмної системи.

5. Перелік обов'язкового ілюстративного матеріалу: 24 рисунків, 37 таблиць.

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Вибір та затвердження теми дипломної роботи, підбір літератури	**до 21 серпня 2020 р.**	
2	Розробка змісту дипломної роботи. Узгодження плану випускової роботи з керівником. Затвердження плану випускної роботи завідувачем кафедри технологій управління	**до 28 серпня 2020 р.**	
3	Підбір інформаційного матеріалу для написання Розділу 1 кваліфікаційної магістерської роботи	**до 02 вересня 2020 р.**	
4	Виконання та оформлення Розділу 1 кваліфікаційної магістерської роботи	**до 30 вересня 2020 р.**	
5	Підбір інформаційно-статистичного матеріалу для написання Розділу 2 кваліфікаційної магістерської роботи	**до 03 жовтня 2020 р.**	
6	Виконання та оформлення Розділу 2 кваліфікаційної магістерської роботи	**до 31 жовтня 2020 р.**	
7	Підбір інформаційно-аналітичного матеріалу для написання Розділу 3 кваліфікаційної магістерської роботи	**до 04 листопада 2020 р.**	
8	Виконання та оформлення Розділу 3 кваліфікаційної магістерської роботи. Передзахист	**до 26 листопада 2020 р.**	
9	Дооформлення дипломної роботи, завершення виконання висновків, літератури, презентації	**01 грудня 2020 р.**	
10	Попередній захист кваліфікаційних магістерських робіт	**02 грудня 2020 р.**	
11	Подача кваліфікаційних магістерських робіт для проходження системи антиплагіат	**9-10 грудня 2020 р.**	
12	Оформлення супровідної документації на захист та підпис у завідувача кафедри технологій управління	**до 18 грудня 2020 р.**	
13	Оформлення супровідної документації на захист та підпис у завідувача кафедри технологій управління	**до 20 грудня 2020 р.**	
14	Захист кваліфікаційних магістерських робіт	**24 грудня 2020 р.**	

7. Консультація з окремого(мих) розділу(ів):

Назва розділу	Консультант (П.І.Б.)	Дата, підпис	
		Завдання видав	Завдання прийняв

8. Дата видачі завдання: **21 серпня 2020 р.**

Керівник дипломної роботи (проєкту) _____ _____
(підпис) (науковий ступінь, посада, вчене звання, прізвище та ініціали)

Завдання прийняв до виконання _____ _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Управління проєктом розробки повнофункціонального авіасимулятора аварійних ситуацій для професійного відбору майбутніх пілотів»: 110 с., 24 рис., 37 табл., 41 інформаційне джерело.

Об’єкт дослідження: професійний відбір абітурієнтів льотних навчальних закладів за психофізичними якостями.

Мета роботи: забезпечення льотних навчальних закладів програмним інструментом оцінювання психофізичних якостей абітурієнтів для відбору тих людей, хто найбільше підходить на професію пілота.

Методи дослідження: оцінювання фізіологічних показників абітурієнтів, порівняльний аналіз, обробка літературних джерел.

Результати магістерської роботи рекомендується використовувати під час проведення вступних конкурсних випробувань абітурієнтів у льотних навчальних закладах.

БЕЗПЕКА ПОЛЬОТІВ, ПІЛОТ, АБІТУРІЄНТ, СТУДЕНТ, ПРОГРАМНА СИСТЕМА, ПРОГРАМНИЙ ЗАСТОСУНОК, УПРАВЛІННЯ ПРОЄКТОМ РОЗРОБКИ, ПСИХОФІЗИЧНИЙ ВІДБІР.

ABSTRACT

Explanatory note to the master’s dissertation "Project management of a full-featured flight simulator development for the professional selection of future pilots": 110 pages, 24 figures, 37 tables, 41 information sources.

The object of research is the professional selection of flight school entrants by their psychophysical qualities.

The objective of the dissertation is providing flight schools with a software tool for assessing the psychophysical traits of the entrants in order to select those people who are most suitable for the profession of pilot.

Research methods: evaluation of physiological indicators of entrants, comparative analysis, processing of literature sources.

It is recommended to use the results of the master's dissertation during the entrance competitive examinations of entrants in flight schools.

FLIGHT SAFETY, PILOT, ENTRANT, STUDENT, SOFTWARE SYSTEM, SOFTWARE APPLICATION, PROJECT MANAGEMENT OF DEVELOPMENT, PSYCHOPHYSICAL SELECTION.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

ПМ – Проєктний менеджмент.

УП – Управління проєктами.

ПС – повітряне судно.

НЗ – навчальний заклад.

ПЗ – програмне забезпечення.

ООП – об’єктно-орієнтоване програмування.

ОС – операційна система.

GDI (Graphics Device Interface) – один з трьох основних компонентів або «підсистем», разом з ядром і Windows API, що складають користувальницький інтерфейс Microsoft Windows.

GUI (Graphic User Interface) – різновид графічного інтерфейсу користувача, в якому елементи інтерфейсу (меню, кнопки, значки, списки і т. п.) представлені користувачеві на дисплеї, виконані у вигляді графічних зображень.

HUD (Head-Up Display) – спосіб відображення інформації, призначений для відображення символьної і графічної інформації (наприклад, навігаційно-пілотажна або спеціальна інформація) на екрані або лобовому склі літального апарату.

UML (Unified Modeling Language) — мова графічного опису для об'єктного моделювання в області розробки програмного забезпечення, що використовується для моделювання бізнес-процесів, системного проєктування та відображення організаційних структур.

ВСТУП

Актуальність теми. У сучасному світі авіація займає одне з передових місць у пасажиро- та вантажоперевезеннях. Вона показала себе як один з найзручніших та найшвидших видів транспорту.

Однією з найголовніших задач авіації є забезпечення безпеки пасажирів та вантажів. На безпеку польотів впливають такі фактори:

- надійність літальних апаратів і наземної техніки обслуговування;
- якість підготовки екіпажу повітряного судна;
- якість системи керування повітряного судна;
- навколишні погодні умови.

Дослідження наукових джерел [1] та аналіз авіакатастроф із «чорних скриньок» свідчить про те, що основним чинником є помилки пілотів або авіадиспетчерів. Дослідження було проведене доцентом кафедри безпеки польотів Кіровоградської льотної академії Національного авіаційного університету Кушнір О.А. у 2014 р. для видання «Молодий вчений».

Для того, щоб пілоти могли прийняти правильне рішення в аварійних ситуаціях під час польоту, вони повинні мати високий рівень професійної підготовки та бути психологічно й психічно стійкими до стресових умов.

Як можна зрозуміти, не кожна людина має такі психофізичні якості, а отже, не кожна людина повною мірою підходить цій професії. Кожний майбутній пілот спочатку проходить своє навчання у льотному навчальному закладі, а при вступі абітурієнтів їхні психофізичні якості майже не піддаються аналізу.

Саме тому актуальною задачею є розробка програмної системи – інструменту об’єктивного оцінювання психофізичних якостей абітурієнтів.

Об'єктом дослідження є професійний відбір абітурієнтів льотних навчальних закладів за психофізичними якостями.

Предметом дослідження є управління проєктом розробки програмної системи для оцінювання психофізичних якостей абітурієнтів льотних навчальних закладів.

У процесі дослідження були застосовані наступні **методи**: оцінювання фізіологічних показників, порівняльний аналіз, обробка літературних джерел.

Метою дипломної роботи є забезпечення льотних навчальних закладів програмним інструментом оцінювання психофізичних якостей абітурієнтів для відбору тих людей, хто найбільше підходить на професію пілота.

Для досягнення мети роботи були поставлені такі завдання:

1. Аналіз предметної області;
2. Вивчення існуючих аналогів програмної системи;
3. Аналіз вимог до програмної системи;
4. Розробка документів стартап-проєкту;
5. Проектування та розробка програмної системи.

Представлено проєкт розробки програмної системи для оцінювання психофізичних якостей абітурієнтів льотних навчальних закладів.

Проаналізовано психофізичні аспекти поведінки людини в екстремальних умовах, а також психологічні та психічні проблеми, що виникають у пілотів. Створено план з управління проєктом розробки запропонованої програмної системи у вигляді документів стартап-проєкту. Відповідно до плану розроблено прототип програмної системи.

Результати магістерської роботи рекомендується використовувати під час проведення вступних конкурсних випробувань абітурієнтів у льотних навчальних закладах.

Практичне використання розробленої пропозиції сприятиме підвищенню безпеки польотів шляхом зниження кількості авіакатастроф, спричинених людським фактором, а саме невідповідністю індивідуальних психофізичних якостей пілотів цій професії та особистих характеристик, що вона вимагає.

Структура роботи: список прийнятих скорочень, вступ, 3 розділи, список використаних джерел.

Апробація результатів дослідження. Результати цього дослідження були представлені у тезах до доповіді до XX Міжнародної науково-практичної конференції здобувачів вищої освіти і молодих учених «Політ. Сучасні проблеми науки».

РОЗДІЛ 1. АНАЛІЗ І ОЦІНКА ПРЕДМЕТУ ДОСЛІДЖЕННЯ

1.1. Аналіз впливу психофізичних якостей пілота на безпеку польотів

Забезпечення безпеки польотів завжди було і залишається головним питанням у цивільній авіації. Важливою складовою безпеки польотів є професійна підготовка льотного складу. Безпека польоту, а отже, і професійна надійність льотного складу, мають велике національне значення. За даними Міжнародної організації цивільної авіації (ICAO), частка аварійних льотних подій з вини екіпажу становить від 60 до 90% [2]. Основні причини трагічних подій в авіації кореняться в людському факторі, головним компонентом якого є відсутність спеціальної теоретичної та психофізіологічної підготовки льотного складу.

Внаслідок бурхливого розвитку світової авіації характер роботи всіх фахівців, що експлуатують авіаційне обладнання, суттєво змінився. Безпека польоту залежить від ефективної взаємодії в системі екіпаж-оточення. Під час польоту всі компоненти цієї системи повинні діяти узгоджено. Нечітка взаємодія, втрата (навіть на короткий час) зв'язку в системі може суттєво знизити рівень безпеки польоту. Тому вимоги до сучасної роботи пілота полягають у тому, щоб він став більш розумним, забезпечуючи сприйняття великого масиву інформації, яку потрібно не тільки зрозуміти, переосмислити, а й зробити єдино правильну оцінку ситуації. Тому велика увага при підготовці пілота повинна приділятися льотним здібностям (працездатності), які є сукупністю фізичних, психологічних, психофізіологічних, психофізичних, психосоціальних факторів і визначають успішне оволодіння професією пілота [2].

Зі збільшенням рівня потреб в ефективному функціонуванні системи "екіпаж-літак-середовище", з розширенням завдань, що стоять перед цивільною авіацією, а саме: суттєво збільшена вантажопідйомність та пасажиромісткість сучасних літальних апаратів (літаків), тривалий час

польоту підвищили вимоги до змісту та якості професійної підготовки льотного складу. Вищевикладене має вплив на:

- підвищення вимог до особистих якостей та психофізіологічної підготовки пілотів;
- посилення відповідальності командира повітряного судна за прийняття рішень під час польоту в умовах високого рівня вимог та змісту технологічної дисципліни, як невід'ємного елементу безпеки польотів.

Важливим у навчанні пілота є його професійна ефективність, яка, в свою чергу, є запорукою безпеки польотів, вона включає: професійну підготовку; психофізіологічний стан; дисципліна.

Психологічний стан залежить від багатьох факторів, а саме природного (генетичного) здоров'я, способу життя, шкідливих звичок, навколишнього середовища, стану здоров'я тощо.

Розглядаючи психофізіологічну підготовку авіаційного персоналу, можна сказати, що це є комплексом заходів, спрямованих на ознайомлення пілота з факторами, які можуть впливати на нього під час польоту, а також система профілактичних заходів щодо зниження цих факторів та підвищення працездатності безпосередньо у польоті [3]. Вивчаючи фактори, які можуть вплинути на авіаційний персонал і вплинути на роботу пілота, ми розділили їх на групи, які:

- пов'язані в основному з особистими характеристиками персоналу;
- впливати на взаємодію між членами екіпажу;
- мають відношення до робочого місця.

Особисті характеристики включають:

Антропометрія (зріст, вага, гострота зору, слух тощо).

Здоров'я та працездатність (фізичний стан людини, патологічний стан тощо).

Залежність (наркоманія - корисний механізм, який забезпечує ефективність дій у повторюваних ситуаціях, але залежність може допомогти ігнорувати потенційно небезпечні показники).

Ідентифікація та сприйняття (здатність наших аналізаторів сприймати подразники, коли наш мозок не в змозі обробляти інформацію). Відволікаючі фактори, шум, втома, перевантаження та інші види стресу можуть зменшити сприйняття стимулу. Ця різниця між ідентифікацією сприйняття стимулу надзвичайно важлива при виконанні завдань, що вимагають високого ступеня пильності.

Пильність (спостереження за допомогою слуху, зору). Нудьга є загальним побічним ефектом настороженості.

Стрес (має як позитивний, так і негативний вплив на роботу пілота). До факторів, що спричиняють стрес, належать втома, брак часу, перевантаження, міжособистісні конфлікти, сімейні проблеми, алкоголь, наркотики тощо.

Порушення ритмів організму (безпека польотів, стан здоров'я екіпажу залежать від зміни біологічних ритмів, це відбувається під час тривалих польотів). Екіпажі на чартерних та вантажних рейсах, які виконують нерегулярно або вночі, можуть страждати від зниження показників через порушення циркадних ритмів.

Сон (порушення нормального сну внаслідок тривалих перельотів, можливе подальше безсоння).

Втома (втома може бути як фізичною, так і психічною і є відображенням неповного відпочинку та може знизити ефективність професійної діяльності та погіршення фізичного самопочуття та як наслідок потенційної небезпеки).

Мотивація (заохочення пілота виконувати свої вміння діяти в екстремальній ситуації).

Особистість та установки (індивідуальні якості та особистісні установки впливають на те, як ми поводимося вдома та на роботі).

До *міжособистісних факторів* належать [4]:

Обробка інформації (можливі помилки в отриманні, обробці та інтерпретації інформації).

Лідерство (лідер може впливати на поведінку та результати діяльності членів екіпажу).

Координація роботи екіпажу (процес набуття навичок командної роботи між висококваліфікованими спеціалістами).

До *факторів робочого місця* належать:

Навантаження (для пілотів робота здебільшого пов’язана з розумовим навантаженням). У стані підвищеного навантаження пілот може допустити помилки при виконанні процедур безпеки.

Навчання та оцінка (навчання розглядається як процес, спрямований на розвиток конкретних навичок, знань або ставлення до виконання певних професійних обов’язків та завдань). Під час та після тренінгу необхідно провести оцінку, щоб забезпечити засвоєння завдань, поставлених тренінгом.

Документація (невідповідність ведення авіаційних документів може негативно вплинути на безпеку польотів).

Конструкція робочого місця (кабіна екіпажу повинна бути влаштована таким чином, щоб забезпечити комфортне розміщення екіпажу в польоті, можливість ефективно виконувати функціональні обов'язки на всіх етапах польоту) [5].

До *психофізичних якостей*, що характеризують професійну надійність пілота, належать [6]:

- вміння виконувати додаткову роботу на тлі основної;
- здатність логічно аналізувати вхідну інформацію;
- здатність швидко змінювати структуру дії в складній ситуації;
- здатність долати перешкоди, що виникають в особливих випадках польоту.

Формування особистісних психофізичних якостей визначає ключову проблему навчання професійної надійності пілота. Його можна оцінити за певними характеристиками, а саме витривалістю до екстремальних стресів, стресостійкістю, здатністю швидко «переключатися». Ми пропонуємо деякі аспекти підвищення професійної надійності навчання пілотів під час навчального процесу, а саме [7]:

- покращити зв'язок між навчальним підрозділом та льотною командою (слабкий взаємозв'язок між теорією та практикою);
- підвищити вимоги до перепідготовки та сертифікації викладацького складу та викладачів (обладнання швидко вдосконалюється);
- запровадити професійну орієнтацію з навчальних предметів;
- включити до навчального процесу аналіз інформації про стан авіаційної безпеки як в Україні, так і у світі;
- збільшити кількість навчальних годин на тренажерах, де пілоти повинні розвивати просторову уяву, вміння аналізувати проблемно-ситуативні проблеми, будувати концептуальні моделі образу польоту, приймати та реалізовувати рішення на основі імідж польоту в екстремальних умовах при моделюванні професійної діяльності.

Характер також є важливою складовою надійності пілота. Більшість компонентів яких формуються в процесі навчання в академії. Необхідно пам'ятати, що характер - це набута якість людини, формування характеру - це тривале формування постійних психологічних утворень під впливом об'єктивних і спеціально створених для особистості умов. На формування характеру найбільш суттєво впливають загальні фактори формування особистості: біологічні основи, соціальне середовище та діяльність людини [8]. Особливо важливим для формування характеру є активність студента чи курсанта і, перш за все, спілкування як необхідна умова самопізнання та самореалізації особистості [9].

1.2. Психологічні та психічні проблеми, що виникають у пілотів

Функції людського мозку, як єдиного каналу інформації, мають природні межі. Інформаційні перевантаження відбуваються в льотній практиці значно частіше, ніж це прийнято вважати. Багато десятиліть частка «людського фактору» оцінюється в межах від 60 до 90 відсотків від загального числа причин авіаційних подій. До того ж, ця цифра стосується лише сертифікованого льотного персоналу [9]. З урахуванням польотів у рамках навчального процесу відомчої та приватної авіації частка людського фактору в середньому збільшується на чотири відсотки. З додаванням «людських причин» від авіаційного диспетчера та персоналу технічного обслуговування загальна частка людського фактору виявляється рівною при традиційних класифікаціях і методиках близько 87% [10].

1.2.1. Класифікації та підходи в дослідженнях факторів помилок пілотів

Імовірнісний підхід. У імовірнісному підході всі людські помилки діляться на два класи: детерміновані і стохастичні. Детерміновані помилки вважаються наслідком неадекватних знань і навичок [11].

Стохастичні помилки можуть зменшуватися за допомогою набору спеціальних правил при проектуванні і експлуатації повітряного судна:

- дублюванням систем;
- зниженням числа функцій і їх спрощенням;

забезпеченням раннього, докритичного виявлення ситуації.

Одночасно при цьому будуються стратегії менеджменту в авіакомпанії:

- кожен фахівець повинен отримати навички з розумінням, що він отримав, коли, де і чому він повинен їх використовувати;
- встановлюється чітке делегування обов'язків в організації;
- відповідальність кожного має, по можливості, найменші розумні обмеження. По суті, весь набір правил ймовірнісного підходу втілюється на практиці в процедурах стандартизації авіакомпаній.

Підхід емпіричних класифікацій. Це найбільш поширений спосіб поділу помилок події на «очевидні з точки зору досвіду, здорового глузду (емпірики) за допомогою словесних формулювань: «невипускання шасі», «бракує палива» і подібних. Незважаючи на зовнішню переконливість, емпіричні формулювання вже багато років не призводять до видимих успіхів, так як вказані «помилки» самі є лише наслідком інших, залишається поза увагою дослідника багатьох чинників. Нижче показано на прикладах, як відбувається змішання системних категорій в емпіричних класифікаціях. Тут ми, крім того, продовжуємо раніше розпочате дослідження моделювання класифікацій безпеки польотів [11].

Біхевіористична (поведінкова) оцінка. Поведінка пілота в кабіні залишається областю пильного вивчення дослідників. Головними труднощами є той факт, що поведінка людини в значній мірі визначається несвідомим, наші знання якого ще дуже малі. По Фрейду, помилки супроводжуються зазвичай несвідомим наміром. Намір може бути наслідком витіснених бажань, імпульсів з подолання ланцюга життєвих стресових ситуацій. Стрес - не в ситуації, а в тому, як ми сприймаємо ситуацію. Причому, ознаки ефективного пілота, як правило, виявляються менш досліджуваними і відомими. Велика частина досліджень спрямована на виявлення ознак неефективного пілота, що іноді називають «синдромом помилки авіатора пілота» [11].

1.2.2. Класифікація помилок пілотів

Слово «помилка» несе відображення реалії. Все питання в тому, з яких причин людина поступає не найкращим чином. Дослідження з визначень переходять до моделювання і класифікацій. Емпіричні моделі і класифікації призводять нас до глибшого розуміння в порівнянні, коли ми маємо тільки визначення. Багато з них наводяться в літературі. Нижченаведену модель названо автором «Класи стану ресурсів «3-Н»: нехтування, некомпетентність, недосвідченість. Частки кожного класу стану ресурсів в поведінці оператора визначені автором експертним способом з аналізу безлічі розрізненої статистики.

1Н. Нехтування. Це клас навмисних помилок, свідомих дій, спрямованих на ненавмисний ризик, порушення правил. Цей клас поведінки пілота вбачається в психічному образі польоту і самої професії. Достовірні і багаторазові дослідження підтверджують зв'язок цього класу з класом низького професіоналізму. Навпаки, обережність, поміркованість і тверді рішення – сенс професії. Частка несприятливих результатів в цьому класу становить близько 25%.

2Н. Некомпетентність. Це – помилки непідготовленості, невміння, недостатню професійну підготовку, недостатні навички. Частка цього класу ресурсного стану становить приблизно третину: 35%.

3Н. Недосвідченість. Забування, неуважність, нерішучість. Найбільший клас стану ресурсів - близько 40% несприятливих наслідків. Підвидів цього класу може бути багато [12].

Одним з підвидів класу недосвідченість є так звані помилки вибору. Наприклад: «шасі прибрані» - «шасі випущені». Вибір і осциляція свідомості є характерними прикладами, які викликали в до посадки з прибраним шасі з трьома видами сигналізації: світловий («зелені горять»), звуковий (сирена), механічний (смугастий покажчик на капоті). Це дуже поширений стан свідомості людини. Згадаймо, як ми запам'ятовуємо номер телефону: 22 18 81.

Варто один раз переставити подумки дві пари останніх цифр (18 81) - (81 18), і ми можемо згодом в сумнівах довго заглядати в записну книжку.

Ще один вид цього ж класу - звичка, або звичний патерн (шаблон) поведінки. Всі види помилок по «недосвідченості» скорочуються десятиліттями шляхом розробки і впровадження стратегій попередження: карти обов'язкових перевірок, опис алгоритмів всіх процесів, керівництва, настанови.

Взаємозв'язок елементів структури особистості пілота очевидна, але очевидно і їх відмінність в способах навчання. Технічно грамотний фахівець і майстерний оператор може виявитися слабким в судженні і техніці прийняття рішень. Людина, що володіє і тим, і іншим може мати слабкі уявлення про навички лідерства і делегування в екіпажі, що критично послаблює використання ресурсів екіпажу. Нарешті, маючи все вище перераховане, пілот зі слабкою адаптацією до стресових ситуацій або послабленою адаптацією до наркотизуючих засобів (алкоголю, куріння, таблеток) може не впоратися з керуванням в складній ситуації.

1.3. Психофізичні якості абітурієнтів льотних навчальних закладів

Сучасні вимоги до професійної діяльності авіаційного фахівця вимагає пошуку нових шляхів, форм, методів і способів в навчальному процесі під час підготовки студентів льотних навчальних закладів [13]. Недостатньо розвинені психофізіологічні якості (емоційна стійкість, розподіл і переключення уваги, уміння працювати у високому вимушеному темпі і виконувати додаткову роботу на тлі основної діяльності, тощо) – головна перешкода на шляху успішного освоєння студентами програми навчання льотним спеціальностям [14].

1.3.1.Особливості розвитку соціально-психологічних характеристик осіб юнацького віку

В емпіричному дослідженні визначення рівня емоційного інтелекту використана методика Н. Холла та авторська анкета, що була спрямована на виявлення гендерно-вікових та соціально-психологічних характеристик респондентів. Вибірку досліджуваних склали 438 юнаків та юнок (студентів вищих навчальних закладів) віком від 17 до 21 року [15].

Під час обробки даних констатовано низький рівень емоційного інтелекту: низький і середній рівні мають більш ніж 90% респондентів. Отже, невелика кількість досліджуваних здатна використовувати емоції для сприяння мисленню, розуміти емоції й регулювати їх з метою особистісного зростання, психічного здоров'я, душевної гармонії та високої якість життя.

Високий рівень емоційної обізнаності мають 10,2% респондентів. Вони є компетентні у питанні усвідомлення власних емоцій та їх наслідків, та розуміють глибину почуттів інших, проти 42,2% і 47,6% досліджуваних, які за цим показником мають низький і середній рівні відповідно. Показник управління своїми емоціями надав високий рівень лише у 2,5%, вони є емоційно гнучкими, відповідно низький і середній рівні мають 85,2% і 12,3% юнаків та юнок. Високий рівень самомотивації продемонстрували лише 6,7% молоді. Лише у 11,5% досліджуваних виявлений високий рівень емпатії, а отже ці люди розуміють емоційний стан інших, уміють співпереживати. Відповідно низький і середній 37,1% і 51,4%. Здатність до розпізнавання емоцій інших людей на високому рівні виявлено у 9,3% досліджуваних. Вони здатні розуміти іншу людину при використанні каналів експресії: міміки, мови, вегетативної та рухової реакцій. Низький і середній рівні – у 50,7% і 4,0% відповідно [16].

Отже, одержані результати демонструють, що більшість осіб юнацького віку не вміють керувати своїми емоціями та спрямовувати їх у конструктивне русло, емоційно не гнучкі, погано уявляють, що відчувають оточуючі їх люди і як з цім усім можна взаємодіяти.

Виявлено, що в цілому емоційний інтелект вище у юнаків ніж у дівчат, особливо за рахунок управління юнаками своїми емоціями, при цьому з віком здатність до управління своїми емоціями зростає. Констатовано (на рівні тенденції) нижчий рівень емоційного інтелекту серед осіб юнацького віку, які навчаються за технічними спеціальностями. Низький рівень управління своїми емоціями властивий студентам-психологам порівняно з тими, які обрали інші спеціальності, що є неприпустимо в контексті специфіки їхньої професійної діяльності.

Вважається доцільним побудова та апробація програми розвитку емоційного інтелекту осіб юнацького віку. Тому що сучасна стратегія виховання та навчання спрямована більш на розвиток інтелектуального потенціалу. Разом з тим, розвитку емоційного інтелекту приділяється недостатньо уваги, хоч підвищення його рівня є дуже важливим.

1.3.2. Обсяг уваги авіаційних фахівців як необхідна якість для майбутньої професійної діяльності

Однією з важливих психофізіологічних якостей для авіаційного фахівця є обсяг уваги. Тому, наше дослідження спрямоване саме на цю якість. Обсяг уваги – здатність людини оперувати більшою або меншою кількістю інформації в найкоротший проміжок часу.

При оцінці обсягу уваги тахіскопічним способом з десяти пред'явлених в розкид цифр випробуваний може при експозиції їх 0,5 сек. «схопити», запам'ятати і відтворити в середньому 6,2 цифр. Недолік цього способу полягає в тому, що увага перевіряється у вузькому полі [17].

На основі тахіскопічного способу, викладачами кафедри фізичної і психофізіологічної підготовки КЛА НАУ розроблено тест для оцінки обсягу уваги. Так, десять студентів шикуються на одній лінії з інтервалом «руки зігнуті перед грудьми», а випробуваний розташовується по центру групи, на відстані шести метрів від неї так, щоб він бачив всіх в секторі під прямим кутом. Пропонується знайти дві однакові позиції серед десяти різних.

Розробляється шість варіантів завдань: три з них з різними положеннями рук, три – з різними положеннями рук і ніг. Наприклад, десять студентів приймають різні позиції і тільки третій і восьмий студент – руки на пояс. У цьому варіанті відповідь «3 і 8» буде вірним.

Складається список з шести завдань для десяти студентів і наноситься на лист А-4. Викладач подає команди, стоячи позаду студента, якого оцінюють, вказуючи пальцями або дощечкою із зазначенням варіанту, який варіант буде запропонований. Десять осіб готуються до прийняття позицій, читаючи на своєму листі про свій варіант, потім кладуть листки перед собою на підлогу номерами до випробуваного.

За командою «Положення прийняти» десять осіб приймають заздалегідь обумовлені позиції і викладач включає секундомір. Випробуваний знаходить схожі позиції та називає їх номери. За кожну невірну відповідь нараховується штраф +5 секунд. У викладача є ключ для всіх шести завдань. По закінченні час підсумовується.

Параметри оцінювання – 5 -ти бальна шкала, де:

- «Відмінно» – 84,1с і менше;
- «Добре» – від 84,2с до 94с;
- «Задовільно» – від 94,1с до 103,0с.

Особливу увагу необхідно приділяти технології проведення тесту, виключати можливі підказки, міняти порядок строю студентів, проводити обстеження випробовуваних із запрошенням в зал по одній людині, тощо. Даний тест дозволяє викладачам виявити недоліки у підготовці студентів льотних навчальних закладів та вибрати напрям, способи та засоби формування необхідної психофізіологічної якості. Рекомендується для оцінки обсягу уваги не тільки студентів пілотів і диспетчерів, а й інших майбутніх авіаційних фахівців, як юнаків, так і дівчат.

1.3.3. Рівень адаптації майбутніх пілотів до процесу навчання у ВНЗ

Сучасні умови розвитку українського суспільства висувають нові вимоги щодо підготовки фахівців для всіх сфер людської активності. Вони мають компетентно вирішувати професійні завдання, досягаючи максимального результату продуктивної діяльності. А це можливо за використання компетентнісного підходу до їхнього навчання й здобуття освіти у ВНЗ. Компетентність – це інтегративна якість особистості майбутнього фахівця, сформована на основі його знань, умінь, навичок, особистісних моральних якостей, цінностей, здібностей та досвіду діяльності. Компетентність складається з окремих компетенцій (інтегрований результат опанування змісту вищої освіти, що виражається в готовності студента використовувати засвоєні знання, уміння, навички, способи діяльності у конкретних ситуаціях – життєвих та професійних – для вирішення теоретичних і практичних завдань) [17].

Практика фізичного виховання, результати спеціальних наукових досліджень, проведених в останні роки, розширили уявлення про значення й зміст професійно-прикладної фізичної підготовки (ППФП) майбутніх фахівців технічного профілю. Було встановлено, що в процесі ППФП успішно формується великий комплекс психофізіологічних, особистісних якостей, необхідних працівникові в його професійній діяльності.

Так, розглядаючи адаптаційні можливості майбутніх пілотів, використовується модернізований варіант діагностичного опитувальника К. Роджерса, призначений для виявлення рівня успішності соціально-психологічної адаптації.

У проведеному дослідженні процесу адаптації першокурсників було визначенню такі головні труднощі: негативні переживання, пов‘язані з виходом учорашніх учнів зі шкільного колективу, заснованого на моральній допомозі й моральній підтримці; невизначеність мотивації вибору професії; недостатня психологічна підготовка до неї; невміння здійснювати психологічну саморегуляцію поведінки й діяльності; звичка до повсякденного

контролю з боку педагогів; пошук оптимального режиму праці й відпочинку у нових умовах; налагодження побуту і самообслуговування (особливо при переході з домашніх умов до умов гуртожитку); відсутність навичок самостійної роботи; невміння конспектувати, працювати з першоджерелами, словниками, каталогами, довідниками, комп‘ютерною технікою та ін.

За результатами дослідження було виявлено, що на першому курсі у 85% студентів переважала дезадаптація. Це підтвердило думку, що процес розвитку здійснюється, якщо при зміні умов життя людини або виду діяльності у людини виникають нові форми та способи життєдіяльності. На першому курсі у студентів відбувався перехід від ролі школяра до ролі студента, тобто змінювалися способи навчальної діяльності. Будь-які зміни супроводжуються дезадаптацію на початку зміни діяльності. Впродовж 6–12 місяців відбувається адаптація до нових умов. Тобто, термін дезадаптації триває 6–12 місяців від початку нової діяльності. Якщо процес дезадаптації не виникає з будь-яких причин, це може свідчити про відсутність психологічних та особистісних змін, тобто блокування процесу розвитку. Але, починаючи з другого курсу по п‘ятий, у 99% студентів майбутніх пілотів відбулася адаптація.

Таким чином, усі ці труднощі відрізняються за своїм походженням. Одні є об‘єктивними, інші мають суб‘єктивний характер і пов‘язані зі слабкою підготовкою, недоліками виховання у сім‘ї та школі. При зарахуванні сучасного абітурієнта до ВНЗ, в основному, враховується лише його успішність – рівень знань. Відсутність іншої інформації у викладача призводить до того, що процес залучення студентів до життя у ВНЗ «затягується». Для вироблення тактики і стратегії, що забезпечує оптимальну адаптацію студента до ВНЗ, важливо знати життєві плани, інтереси першокурсника, систему домінуючих мотивів, рівень претензій, самооцінки, здатність до свідомої регуляції поведінки і т. ін.

Саме тому, важливо звернути увагу професорсько-викладацького складу ВНЗ, які мають допомогти студентові у процесі становлення його не тільки як

майбутнього фахівця, але й як особистості, сприяти створенню всередині мікроструктури групи атмосфери свободи, самоповаги і творчості.

1.4. Системи оцінювання психофізичних якостей абітурієнтів

Льотна робота належить до найбільш складних видів професійної діяльності людини, так як протікає в умовах підвищеного психоемоційного напруження. Автоматизація процесів пілотування літака та підвищення ступеню відмовостійкості сучасних повітряних суден не знизили вимог до емоційної стійкості пілотів, а навпаки - ускладнили механізми протікання стресових реакцій при виникненні особливих випадків польоту.

В таких умовах виникла необхідність більш глибокого дослідження ролі емоційної стійкості в структурі професійної надійності льотного складу та посилення заходів щодо підвищення її рівня в процесі професійної підготовки майбутніх пілотів.

Експерти довели [20], що в основі професійної надійності льотного складу лежать професійно важливі якості, наявність яких забезпечується як професійно-психологічним відбором, так і подальшим цілеспрямованим формуванням засобами професійної підготовки. Високу ефективність підтвердили розроблені підходи формування професійно важливих якостей засобами тренажерної і спеціальної фізичної підготовок. Однак при цьому, на думку багатьох дослідників, до теперішнього часу не всі можливості теоретичної підготовки в області формування навичок льотного складу реалізовані.

Серед основних причин такого стану справ те, що людські ресурси екіпажу мають дуже складну, слабо вивчену динаміку, а програми підготовки, технічні системи поки ще дуже механістичні і мало вкладаються в такий складний феномен, яким є людина. Досягнення в області обчислювальної техніки, істотне поліпшення якостей і надійності датчиків параметрів польоту (включаючи місце розташування повітряного судна), положень важелів управління і керуючих поверхонь літака, вдосконалення конструкції силових

приводів рулів, поява електронних індикаторів для відображення пілотажно-навігаційної інформації – все це зумовило, з одного боку, значне розширення функціональних можливостей систем автоматичного управління польотом, а з іншого – суттєві зміни в процесах сприйняття льотчиком приладової (інструментальної) інформації, стрімке багаторазове ускладнення структури його інтелектуальної діяльності.

Таким чином, нові умови пілотування літака вимагають розробки, наукового обґрунтування і впровадження в систему професійної підготовки майбутніх пілотів нових психолого-педагогічних підходів формування їх професійної надійності.

1.4.1. Існуючі системи оцінювання психофізичних якостей

Дана програмна система відноситься до категорії «Авіасимулятори». Найближчими у цій категорії по фунціоналу програмними засобами є:

- Microsoft Flight Simulator;
- X-Plane 11.

Microsoft Flight Simulator - комп'ютерна гра в жанрі авіасимулятора, що розробляється компанією Asobo Studio. Це одинадцята частина серії Microsoft Flight Simulator, наступник Microsoft Flight Simulator X. В симуляторі представлена вся поверхня планети Земля, включаючи всі країни світу, міста і аеропорти - для цього використовуються текстури і топографічні дані з карт Bing Maps. 3D-об'єкти відображаються з використанням технології Microsoft Azure [18].

Microsoft Flight Simulator оснащений ігровим движком Asobo і використовує інформацію з карт Bing Maps, надаючи доступ до більш ніж 2 петабайт даних із хмари. Для фотореалістичного і достовірного відтворення і розміщення на потрібних місцях тривимірних будівель, дерев, об'єктів рельєфу і тому подібного використовуються технології процедурної генерації, фотограмметрії і машинного навчання.

Рис. 1.1. Логотип Microsoft Flight Simulator

X-Plane 11 - авіасимулятор, розроблений для Mac OS X (також доступний для Windows та Linux) компанією Laminar Research. До складу X-Plane 11 входять кілька комерційних, військових і інших літаків, а також глобальний пейзаж, який охоплює більшу частину Землі. Також в поставку авіасимулятора входить програмне забезпечення для створення та налаштування моделей літаків. X-Plane 11 має систему плагінів, що дозволяє користувачам розширювати функціональність симулятора і створювати свої власні світи або копії реальної місцевості [19].

Рис. 1.2. Логотип X-Plane 11

Варто зазначити, що наведені вище наближені програмні продукти не мають механізмів симуляції аварійних ситуацій в польоті, тому «повними» аналогами запропонованої нижче програмної системи їх називати не зовсім коректно.

1.4.2. Запропонована програмна система оцінювання психофізичних якостей

З перших днів появи льотних спеціальностей питання про структуру професійної надійності її носіїв встав дуже гостро, так як ціна помилок і невідповідностей вимогам діяльності в цій сфері праці незрівнянно велика. Виявилося, що далеко не кожен фізично здорова людина може стати хорошим пілотом і що для успішного оволодіння льотним справою він повинен володіти певними психофізіологічними якостями, тобто виникло питання про необхідність практичної оцінки професійної придатності, проведення психологічного відбору льотного складу. Для претендентів на освоєння льотної спеціальності був розроблений і впроваджений професійно-

психологічний відбір, який в значній мірі вирішував проблеми психофізіологічного забезпечення льотної роботи [20].

Продуктом цього проєкту є програмний продукт, призначений для використання льотними навчальними закладами саме з метою забезпечити професійно-психологічний відбір абітурієнтів, що є суттєвим заходом забезпечення майбутньої безпеки польотів.

Програмний продукт являє собою симулятор польоту літака у реальному часі, під час польоту якого можуть відбутися різні позаштатні ситуації, льотні аномалії та аварії. Завданням абітурієнта, що проходить тестування, є швидке прийняття правильних рішень для забезпечення безпеки подальшого польоту до кінцевого пункту маршруту або негайної вимушеної посадки. Ці дії стосуються як безпосереднього керування польотом літака, так і віддання необхідних команд екіпажу.

Набір можливих аварійних ситуацій складено з урахуванням вимог до пілотів міжнародних стандартів ICAO, IATA та IFALPA, наведених нижче [21].

Критерії оцінки компетенції і поведінкові індикатори:

- Застосування процедур: визначає і застосовує процедури відповідно до опублікованих експлуатаційними інструкціями та чинними правилами, застосовуючи відповідні знання;
- Обмін інформацією: Демонструє навички ефективного усного, немовних і письмового обміну інформацією в нормальних і позаштатних ситуаціях;
- Управління траєкторією польоту повітряного судна, автоматизація: Контролює траєкторію польоту повітряного судна за допомогою засобів автоматизації, включаючи відповідне використання систем управління польотом і наведенням;
- Лідерство і робота в команді: Демонструє навички ефективного керівництва і колективної роботи;

• Вирішення проблем і прийняття рішень: Точно визначає ризики і вирішує проблеми. Використовує відповідні процеси прийняття рішень;

• Поінформованість про ситуацію: Сприймає і розуміє всю наявну відповідну інформацію і передбачає, що може трапитися і вплинути на політ;

• Управління робочим навантаженням: Ефективно управляє наявними ресурсами і своєчасно виконує завдання при будь-яких обставин.

Результуюча числова оцінка абітурієнта формується на основі наступних факторів:

• Дотримання початкового маршруту до критичної події;

• Швидкості прийняття рішення після настання події;

• Правильності прийнятого рішення та покрокового його виконання;

• Чи була диспетчерська служба проінформована про настання події;

• Результат польоту: чи був врятований літак та усі люди, що перебували на борту, а також, якщо можливо, апаратура літака та вантаж.

Ця оцінка матиме суттєвий вплив на процес вступу абітурієнта до льотного навчального закладу.

Таким чином, цей програмний застосунок є достатньо об’єктивним засобом оцінювання психофізичних якостей абітурієнтів.

Висновки до розділу 1

Проаналізувавши викладену вище інформацію, можна зробити висновок про те, наскільки важливо враховувати психофізичні якості абітурієнтів при вступі до льотних навчальних закладів.

Правильно відібрані люди, що найбільше відповідають професії пілота, будуть більш зацікавлені у покращенні, відточуванні своїх професійних навичок до майстерності, будуть більш стійкими до негативних аспектів майбутньої роботи, приймати правильні рішення у стресових та аварійних ситуаціях під час польоту літального апарату.

Особливо враховуючи те, що причиною переважної більшості аварій із серйозними наслідками є саме людських фактор, це однозначно максимально забезпечить безпеку польоту і, як наслідок, вбереже людські життя та техніку.

РОЗДІЛ 2. ОСОБЛИВОСТІ УПРАВЛІННЯ ПРОЄКТОМ РОЗРОБКИ ПРОГРАМНОЇ СИСТЕМИ

У людства за всю історію накопичився значний список успішно реалізованих складних проєктів. Від будівництва Пірамід у Гізі до відправки людини на Місяць, найсміливіші людські починання вимагали злагодженої роботи тисяч людей. А це має на увазі складну систему управління проєктами.

Управління проєктами – це керування і організація всього, що потрібно для досягнення мети – вчасно і в рамках бюджету. Чи то розробка нового програмного забезпечення, чи проведення маркетингової компанії або висадка людини на Марс - проєктне управління дозволяє домогтися успіху.

Всі проєкти різні. Не існує ідеальної системи управління проєктами, що підходить для кожного з видів проєктів. Також не існує системи, яка б підходила кожному керівнику і була зручна для всіх членів команди. Однак за час існування проєктного управління було створено чимало ефективних підходів, методик і стандартів, які можна взяти на озброєння. Про найпопулярніших з них ми сьогодні і поговоримо.

Розроблені підходи сильно відрізняються один від одного. Вони розрізняються по областях застосування, деталізованості, самодостатності і формалізації. У заголовку ми назвали їх «методами» для зручності, але насправді в статті представлені стандарти, концепції, методи і фреймворки, які застосовуються в управлінні проєктами.

Імена Ніла Армстронга і Базза Олдрина назавжди увійдуть в історію як символи одного з найбільших досягнень людства - висадки людини на Місяць. Однак основний внесок в цю подію внесли 400 000 співробітників НАСА і 20 000 компаній і університетів, які працювали разом над місією «Аполлон».

У 1961 році Джон Кеннеді поставив завдання висадити людину на супутнику Землі і повернути його назад - при тому, що на той момент НАСА відправляли людини в космос лише на 15 хвилин. Така амбітна мета

потребувала неймовірної кількості ресурсів, кооперації, інновацій та планування.

Як говориться в книзі НАСА «Managing the Moon Program», основна проблема полягала не в тому, «що робити?», А в тому, «як зробити стільки за такий короткий термін?». За словами доктора Макса Фагета, глави інжинірингу в Космічному центру імені Ліндона Джонсона, тоді в НАСА не представляли, як укласти всі необхідні дії в 10 років. А тому першим кроком стало «розбити проєкт на керовані етапи».

Потім важливо було прискорити виконання кожної окремої фази і упевнитися, що команди і компанії, що працюють на кожній фазі, ефективно взаємодіють один з одним і вчасно постачають результати. Це завдання було покладено на доктора Джорджа Мюллера, який керував кожною частиною проєкту «Аполлон», від Білого Дому до постачальника найдрібнішої деталі. Щоб контролювати проєкт було легше, він вирішив розбити проєкт на 5 областей: «Контроль Програми», «Системна Інженерія», «Тестування», «Надійність і Якість» та «Льотна експлуатація». Схема управління програмою Аполлон представлена на рисунку 2.1.

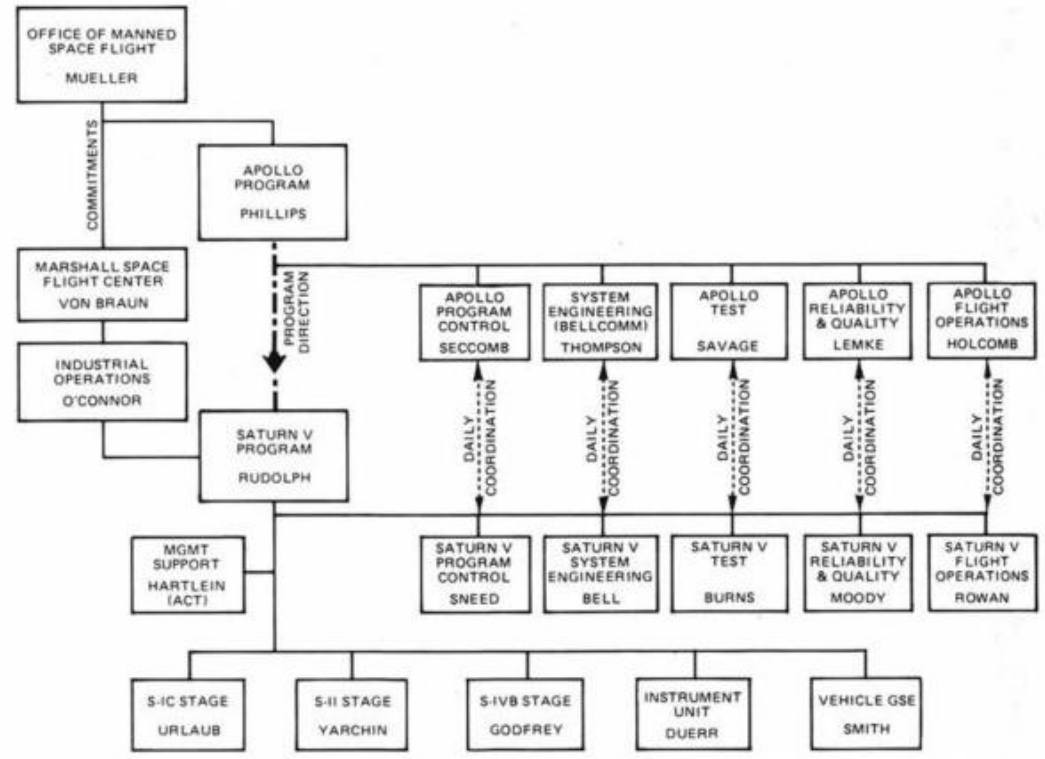

Рис. 2.1. Схема управління космічною програмою «Аполлон»

Ця система з 5 етапів - названих «Етапами GEM» на честь ініціалів доктора Мюллера - була розроблені «заради фокусування на тестуванні продукту, і на його розробці з урахуванням того, що його будуть тестувати», як зазначає сам Мюллер. «Контроль Програми» визначав, що потрібно зробити, керував бюджетом та вимогами, а також керував взаємозв'язками елементів програми. Область «Системна інженерія» відповідала за розробку нових пристроїв і вузлів, «Тестування» за те, що ці нові елементи працюють, «Надійність і Якість» перевіряли розроблені елементи на відповідність вимогам і стандартам, а «Льотна експлуатація» відповідала за те, що ці вузли будуть працювати під час польоту.

Багато спочатку поставилися до методу, запропонованого Мюллером, зі скептицизмом, але врешті-решт йому вдалося переконати членів програми в необхідності проходження даного алгоритму. Дана система показала свою ефективність - проєкт був завершений успішно, і, можна навіть сказати, тріумфально, з випередженням заявлених термінів. Це стало можливо тільки завдяки розбиття масштабного проєкту на керовані, повторювані етапи, що дозволило працювати безлічі окремих компаній і фахівців в єдиному ритмі. Так проєктне управління довело свою ефективність в Космічної гонці.

Однак проєктне управління не було винайдено НАСА і доктором Мюллером. Єгипетські піраміди і Велика Китайська стіна є продуктами проєктного управління з доісторичних епох. На жаль, документальних свідчень того, як проходила реалізація і управління цими проєктами не збереглося, і нинішнє проєктне управління відірвано від знань минулих століть.

2.1. Історія управління проєктами розробки програмного забезпечення

У 1970-х і 1980-х роках індустрія програмного забезпечення дуже швидко зростала, оскільки комп'ютерні компанії швидко визнали порівняно низьку вартість виробництва програмного забезпечення порівняно з

виробництвом апаратних засобів та схем. Для управління новими зусиллями з розробки компанії застосовували встановлені методи управління проєктами, але графіки проєктів провалювались під час тестових запусків, особливо коли в сірій зоні сталася плутанина між специфікаціями користувачів та програмним забезпеченням, що постачається. Щоб уникнути цих проблем, методи управління програмними проєктами зосереджувались на відповідності вимог користувачів до поставленої продукції методом, відомим зараз як модель водоспаду.

По мірі розвитку галузі аналіз помилок управління програмними проєктами показав, що найпоширенішими причинами є наступні:

1. Недостатня участь кінцевого користувача;
2. Погане спілкування між замовниками, розробниками, користувачами та менеджерами проєктів;
3. Нереалістичні або не сформульовані цілі проєкту;
4. Неточні оцінки необхідних ресурсів;
5. Погано визначені або неповні системні вимоги та специфікації;
6. Погана звітність про стан проєкту;
7. Невдало керовані ризики;
8. Використання незрілої технології;
9. Неможливість вирішити складність проєкту;
10. Недбалі практики розвитку;
11. Політика зацікавлених сторін (наприклад, відсутність підтримки виконавчої влади або політика між замовником та кінцевими споживачами);
12. Комерційний тиск.

Перші п’ять пунктів у наведеному вище списку демонструють труднощі у формулюванні потреб клієнта таким чином, що належні ресурси можуть забезпечити належні цілі проєкту. Конкретні інструменти управління програмними проєктами корисні і часто необхідні, але справжнім мистецтвом управління програмними проєктами є застосування правильного методу, а потім використання інструментів для підтримки методу. Без методу

інструменти нічого не варті. З 1960-х років кілька власних методів управління проєктами програмного забезпечення були розроблені виробниками програмного забезпечення для власного використання, тоді як фірми, що займаються комп'ютерними консалтингами, також розробили подібні методи для своїх клієнтів. Сьогодні методи управління програмними проєктами все ще розвиваються, але нинішня тенденція веде від моделі водоспаду до більш циклічної моделі доставки проєктів, яка імітує процес розробки програмного забезпечення.

2.2. Діяльність щодо управління розробкою програмного забезпечення

Управління розробкою програмного забезпечення (англ. Software project management) — це мистецтво планування і керування проєктами з розробки програмного забезпечення; особливий вид управління проєктами, в рамках якого відбувається планування, реалізація, відслідковування і контроль за проєктами з розробки програмного забезпечення [22].

Процес розробки програмного забезпечення стосується насамперед виробничого аспекту розробки програмного забезпечення, на відміну від технічного аспекту, такого як програмні засоби. Ці процеси існують насамперед для підтримки управління розробкою програмного забезпечення та, як правило, перекошені у вирішенні проблем бізнесу. Багато процесів розробки програмного забезпечення можна запускати подібним чином до загальних процесів управління проєктами. Прикладами є:

1. Міжособистісне спілкування та врегулювання та вирішення конфліктів. Активне, часте та чесне спілкування є найважливішим фактором збільшення ймовірності успіху проєкту та пом'якшення проблемних проєктів. Команда розробників повинна шукати залучення кінцевих споживачів та заохочувати участь користувачів у процесі розробки. Відсутність залучених користувачів може призвести до неправильного тлумачення вимог, нечутливості до мінливих потреб клієнтів та нереальних очікувань клієнта.

Розробникам програмного забезпечення, користувачам, менеджерам проєктів, замовникам та спонсорам проєкту потрібно регулярно та часто спілкуватися. Інформація, отримана в результаті цих обговорень, дозволяє проєктній групі аналізувати сильні, слабкі сторони, можливості та загрози та діяти на основі цієї інформації, щоб скористатися можливостями та мінімізувати загрози. Навіть погані новини можуть бути хорошими, якщо їх повідомити відносно рано, оскільки проблеми можна пом'якшити, якщо їх не виявити занадто пізно. Наприклад, випадкові розмови з користувачами, членами команди та іншими зацікавленими сторонами часто можуть виявити потенційні проблеми раніше, ніж офіційні зустрічі. Усі комунікації повинні бути інтелектуально чесними та достовірними, і необхідна регулярна, часта, високоякісна критика роботи з розвитку, якщо вона надається спокійно, шанобливо, конструктивно, не звинувачує, не гнівається. Часті випадкові комунікації між розробниками та кінцевими користувачами, а також між керівниками проєктів та клієнтами необхідні для того, щоб проєкт залишався актуальним, корисним та ефективним для кінцевих користувачів і в межах того, що можна завершити. Ефективна міжособистісна комунікація та врегулювання та вирішення конфліктів є ключем до управління програмними проєктами. Жодна методологія або стратегія вдосконалення процесів не може подолати серйозні проблеми в спілкуванні або безгосподарне управління міжособистісними конфліктами. Більше того, результати, пов'язані з такими методологіями та стратегіями вдосконалення процесів, покращуються завдяки кращому спілкуванню. Спілкування повинно бути зосереджене на тому, чи розуміє команда команду проєкту та чи просувається команда до цієї мети. Кінцеві користувачі, розробники програмного забезпечення та керівники проєктів повинні часто задавати елементарні прості запитання, які допомагають виявити проблеми, перш ніж вони наближаться до стихійних лих. Хоча участі кінцевих користувачів, ефективного спілкування та колективної роботи недостатньо, вони необхідні для забезпечення гарного результату, і їх відсутність майже напевно призведе до поганого результату.

2. Управління ризиками – це процес вимірювання або оцінки ризику, а потім розробка стратегій управління ризиком. Загалом, використовувані стратегії включають передачу ризику іншій стороні, уникнення ризику, зменшення негативного ефекту ризику та прийняття деяких або всіх наслідків певного ризику. Управління ризиками в управлінні програмним проєктом починається з бізнес-обґрунтування запуску проєкту, яке включає аналіз витрат та вигод, а також перелік резервних варіантів відмови від проєкту, який називається планом на випадок непередбачених ситуацій.

3. Підмножиною управління ризиками є управління можливостями, що означає те саме, за винятком того, що потенційний результат ризику матиме позитивний, а не негативний вплив. Незважаючи на те, що теоретично обробляються однаково, використання терміну "можливість", а не дещо негативного терміну "ризик", допомагає тримати команду зосередженою на можливих позитивних результатах будь-якого даного реєстру ризиків у своїх проєктах, таких як спінінгові проєкти, непередбачені та безкоштовні додаткові ресурси.

4. Управління вимогами - це процес ідентифікації, виявлення, документування, аналізу, відстеження, встановлення пріоритетів та узгодження вимог, а потім контролю змін та передачі інформації відповідним зацікавленим сторонам. Управління вимогами, яке включає аналіз вимог, є важливою частиною процесу програмного забезпечення; за допомогою якого бізнес-аналітики або розробники програмного забезпечення визначають потреби або вимоги клієнта; визначивши ці вимоги, вони можуть спроектувати рішення.

5. Управління змінами - це процес ідентифікації, документування, аналізу, встановлення пріоритетів та узгодження змін в обсязі (управління проєктами), а потім контроль змін та передача інформації відповідним зацікавленим сторонам. Аналіз впливу нового або зміненого обсягу змін, що включає аналіз вимог на рівні змін, є важливою частиною процесу програмної інженерії; за допомогою якого бізнес-аналітики або розробники програмного

забезпечення визначають змінені потреби або вимоги клієнта; визначивши ці вимоги, вони можуть переробити або змінити рішення. Теоретично кожна зміна може вплинути на часові рамки та бюджет програмного проєкту, і тому за визначенням перед затвердженням має включати аналіз ризику та вигоди.

6. Управління конфігурацією програмного забезпечення - це процес ідентифікації та документування самої сфери застосування, яка є програмним продуктом, що реалізується, включаючи всі підпродукти та зміни, а також забезпечує їх передачу відповідним зацікавленим сторонам. Загалом, використовувані процеси включають контроль версій, конвенцію про іменування (програмування) та програмні архівні угоди.

7. Управління випусками - це процес ідентифікації, документування, встановлення пріоритетів та узгодження випусків програмного забезпечення, а потім управління графіком випусків та передача інформації відповідним зацікавленим сторонам. Більшість програмних проєктів мають доступ до трьох програмних середовищ, до яких програмне забезпечення може бути випущено; Розробка, випробування та виробництво. У дуже великих проєктах, де розподіленим командам потрібно інтегрувати свою роботу перед випуском для користувачів, часто існує більше середовищ для тестування, які називаються модульним тестуванням, системним тестуванням або інтеграційним тестуванням, перед випуском на тестування прийняття користувачами (UAT).

8. Підмножиною управління випуском, яка привертає увагу, є управління даними, оскільки очевидно, що користувачі можуть тестувати лише на основі відомих їм даних, а "реальні" дані є лише в програмному середовищі, що називається "виробництво". Тому, щоб перевірити свою роботу, програмісти повинні також часто створювати "фіктивні дані" або "заглушки даних". Традиційно колись для цього використовувались старіші версії виробничої системи, але оскільки компанії все більше покладаються на сторонніх співробітників для розробки програмного забезпечення, дані компаній можуть не передаватися командам розробників. У складних середовищах можуть створюватися набори даних, які потім мігрують між

тестовими середовищами згідно з графіком випуску тестів, подібно до загального графіку випуску програмного забезпечення.

9. Технічне обслуговування та оновлення - це процес, де Вимоги та потреби замовника завжди беруть участь. Вони, безсумнівно, знайдуть помилки, можуть запросити нові функції та попросити іншу функціональність та інші оновлення. Отже, всі ці запити повинні перевіряти та виконувати вимоги та задоволення замовника.

Надзвичайно велику роль в управлінні проєктом також відіграє його планування. Мета планування проєкту – визначити його обсяг, оцінити задіяну роботу та створити графік проєкту. Планування проєкту починається з вимог, які визначають програмне забезпечення, яке буде розроблено. Потім розробляється план проєкту для опису завдань, які призведуть до завершення. Виконання проєкту - це процес виконання завдань, визначених у проєктному плані.

Метою моніторингу та контролю проєкту є підтримка команди та керівництва в курсі прогресу проєкту. Якщо проєкт відхиляється від плану, тоді керівник проєкту може вжити заходів для усунення проблеми. Моніторинг та контроль за проєктом передбачає зустрічі щодо статусу з метою отримання статусу від команди. Коли потрібно внести зміни, контроль змін використовується для постійного оновлення продуктів.

2.3. Управління розробкою проєкту програмного забезпечення за стандартами IEEE

Проєктний менеджмент охоплює усі аспекти проєкту розробки програмної системи, які чітко визначаються за міжнародною системою IEEE.

Інститут інженерів електрики та електроніки (англ. The Institute of Electrical and Electronics Engineers, IEEE) - це професійна асоціація електронної техніки та електротехніки (та суміжних дисциплін) з корпоративним офісом у Нью-Йорку та операційним центром у Піскатей, штат

Нью-Джерсі. Він був утворений в 1963 р. Завдяки об’єднанню Американського інституту інженерів-електриків та Інституту радіотехніків [23].

Через розширення сфери застосування на велику кількість суміжних областей, на нього просто посилаються літери I-E-E-E, за винятком юридичних ділових документів. Станом на 2018 рік, це найбільша у світі асоціація технічних професіоналів, яка налічує понад 423 000 членів у понад 160 країнах світу [24]. Його завданнями є освітній та технічний розвиток електротехнічної та електронної техніки, телекомунікацій, обчислювальної техніки та суміжних дисциплін [25].

IEEE видає третину технічної літератури, що стосується застосування комп'ютерів, управління, електроінженерії, та понад 100 журналів, популярних у середовищі професіоналів, проводить на рік понад 300 великих конференцій [26].

Одним з найвідоміших напрямків роботи організації є розроблення стандартів, зокрема, стандартів аспектів розробки програмного забезпечення.

Життєвий цикл розробки програмного забезпечення за IEEE:

- SQA - Забезпечення якості програмного забезпечення (IEEE 730);
- SCM - Управління конфігурацією програмного забезпечення (IEEE 828);
- STD - Документація щодо тестування програмного забезпечення (IEEE 829);
- SRS - Специфікація вимог до програмного забезпечення (IEEE 830);
- V&V - Перевірка та перевірка програмного забезпечення IEEE 1012);
- SDD - Опис дизайну програмного забезпечення (IEEE 1016);
- SUD - Документація користувача програмного забезпечення (IEEE 1063).

Забезпечення якості програмного забезпечення (SQA) - це засіб і практика моніторингу процесів та методів програмної інженерії, що використовуються в проєкті для забезпечення належної якості програмного забезпечення [27]. Він включає стандарти та процедури, які менеджери,

адміністратори або навіть розробники можуть використовувати для перегляду та аудиту програмних продуктів та заходів, щоб переконатися, що програмне забезпечення відповідає критеріям якості, які посилаються на стандарти. Відповідно до Automotive SPICE (який базується на ISO / IEC 15504), забезпечення якості програмного забезпечення є допоміжним процесом, який забезпечує незалежне запевнення у тому, що всі робочі продукти, дії та процеси відповідають наперед визначеним планам та стратегіям якості. SQA охоплює весь процес розробки програмного забезпечення, включаючи розробку вимог, розробку програмного забезпечення, кодування, огляд коду, контроль вихідного коду, управління конфігурацією програмного забезпечення, тестування, управління випусками та інтеграцію програмного забезпечення. Він організований за цілями, зобов'язаннями, здібностями, діями, вимірами, верифікацією та валідацією.

Управління конфігурацією програмного забезпечення (SCM) - це завдання відстеження та контролю змін у програмному забезпеченні, що є частиною великого міждисциплінарного поля управління конфігурацією [28]. Практика СКМ включає контроль за переглядом та встановлення базових рівнів. Якщо щось піде не так, SCM може визначити, що було змінено і хто це змінив. Якщо конфігурація працює добре, SCM може визначити, як її тиражувати на багатьох хостах. Цілі SCM:

- Ідентифікація конфігурації - ідентифікація конфігурацій, елементів конфігурації та базових ліній.
- Конфігураційний контроль - Реалізація контрольованого процесу змін. Зазвичай це досягається шляхом створення ради контролю змін, основною функцією якої є затвердження або відхилення всіх запитів на зміни, що надсилаються проти будь-якого базового рівня.
- Бухгалтерський облік статусу конфігурації - Запис і звітування всієї необхідної інформації про стан процесу розробки.
- Аудит конфігурації - Забезпечення того, щоб конфігурації містили всі призначені для них частини та були надійними щодо їх специфічних

документів, включаючи вимоги, архітектурні специфікації та посібники користувача.

- Управління збірками - управління процесом та інструментами, що використовуються для збірки.
- Управління процесами - Забезпечення дотримання процесу розвитку організації.
- Управління навколишнім середовищем - управління програмним та апаратним забезпеченням, що розміщує систему.
- Командна робота - полегшити взаємодію команд, пов’язану з процесом.
- Відстеження дефектів - переконання, що кожен дефект має можливість простежуватись назад до джерела.

Документація для тестування програмного забезпечення є життєво важливим елементом, який піднімає будь-які експериментальні дії до рівня тестування програмного забезпечення. Міжнародні організації, такі як IEEE та ISO, опублікували стандарти документації до тестування програмного забезпечення. IEEE 829-2008, також відомий як Стандарт 829 для програмної та системної документації для випробувань, - це стандарт IEEE, який визначає форму набору документів для використання на восьми визначених етапах тестування програмного забезпечення та тестування системи, кожен з яких потенційно може створити свій власний окремий тип документа. Стандарт визначав формат цих документів, але не визначав, чи всі вони повинні бути надані, а також не включав жодних критеріїв щодо відповідного змісту цих документів [29].

Специфікація вимог до програмного забезпечення (SRS) - це опис програмної системи, що розробляється. Специфікація вимог до програмного забезпечення викладає функціональні та нефункціональні вимоги, і вона може включати набір випадків використання, що описують взаємодію користувачів, яку програмне забезпечення має надавати користувачеві для ідеальної взаємодії. Специфікація вимог до програмного забезпечення створює основу

для домовленості між замовниками та підрядниками чи постачальниками про те, як повинен функціонувати програмний продукт (у ринковому проєкті ці ролі можуть виконувати відділи маркетингу та розвитку). Специфікація вимог до програмного забезпечення - це ретельна оцінка вимог до більш конкретних етапів проектування системи, і її метою є зменшення подальшого редизайну. Це також повинно забезпечити реальну основу для оцінки витрат на продукцію, ризиків та графіків. За умови належного використання специфікації вимог до програмного забезпечення можуть допомогти запобігти збою програмного проєкту [30].

У документі специфікації вимог до програмного забезпечення перелічено достатні та необхідні вимоги для розробки проєкту. Щоб вивести вимоги, розробник повинен чітко розуміти продукт проєкту, що розробляється. Це досягається завдяки детальному та постійному спілкуванню з командою проєкту та замовником протягом усього процесу розробки програмного забезпечення.

Верифікація та валідація (V&V) в управлінні проєктами програмного забезпечення, тестуванні програмного забезпечення та програмній інженерії – це процес перевірки відповідності програмної системи специфікаціям та відповідності її цільовому призначенню. Це також може називатися контролем якості програмного забезпечення. Зазвичай відповідальність тестувальників програмного забезпечення є частиною життєвого циклу розробки програмного забезпечення. Говорячи простими словами, перевірка програмного забезпечення: "Якщо припустити, що ми повинні створити продукт, чи досягає наше програмне забезпечення своїх цілей без помилок чи прогалин?" З іншого боку, перевірка програмного забезпечення полягає в наступному: "Чи став продукт тим, що ми мали побудувати? Чи продукт проєкту відповідає вимогам високого рівня?" [31].

Опис дизайну програмного забезпечення (він же документ проєктування програмного забезпечення або SDD; також специфікація програмного забезпечення) – це письмовий опис програмного продукту, який розробник

програмного забезпечення пише, щоб надати команді розробників програмного забезпечення загальні вказівки щодо архітектури програмного проєкту [32]. SDD зазвичай супроводжує архітектурну схему з вказівниками до детальних характеристик менших частин конструкції. Опис необхідний для координації роботи великої команди в рамках єдиного бачення, має бути стабільним посиланням та окреслювати всі частини програмного забезпечення та те, як вони працюватимуть.

Документація програмного забезпечення – це письмовий текст чи ілюстрація, що супроводжує програмне забезпечення комп'ютера, або вбудована у вихідний код. Документація пояснює, як працює програмне забезпечення або як ним користуватися, і може означати різні речі для людей, які виконують різні ролі [33].

Документація є важливою частиною програмної інженерії. Типи документації включають:

- Вимоги – Заяви, що визначають атрибути, можливості, характеристики або якості системи. Це фундамент для того, що буде впроваджено.
- Архітектура / Дизайн – Огляд програмного забезпечення. Включає відношення до навколишнього середовища та принципи побудови, які будуть використовуватися при проектуванні програмних компонентів.
- Технічна – Документація коду, алгоритмів, інтерфейсів та API.
- Кінцевий користувач – Посібники для кінцевого користувача, системних адміністраторів та допоміжного персоналу.
- Маркетинг – Як продати товар на ринку та аналіз ринкового попиту.

2.4. Найрозповсюдженіші методології управління проєктами

За всю історію проєктного менеджменту було створено безліч різних методів управління проєктами під практично будь-які потреби. Головне – чітко зрозуміти, що є найважливішим для проєкту – дедлайни, ресурси, дотримання процесу, або відразу кілька факторів – а потім вибрати метод управління проєктом, орієнтований на досягнення цього показника.

2.4.1. Класичне проєктне управління

Найбільш очевидний спосіб зробити свій проєкт більш керованим - це розбити процес його виконання на послідовні етапи. Саме на такій лінійної структурі базується традиційне проєктне управління. У цьому сенсі воно нагадує комп'ютерну гру – можна перейти на наступний рівень не завершивши попередній. Схема робочого процесу приведена на рис 2.2 [34].

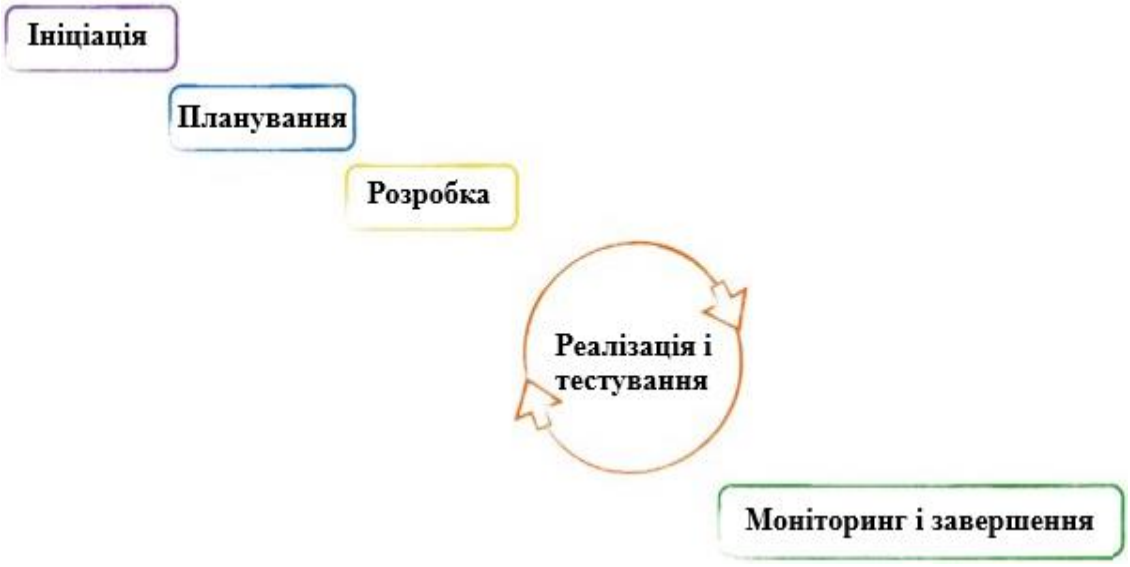

Рис. 2.2. Схема класичного підходу до проєктного менеджменту

Даний підхід орієнтований на проєкти, в яких є строгі обмеження щодо послідовності виконання завдань. Наприклад, будівництво будинку – не можна зводити стіни без фундаменту.

Зазвичай виділяють 5 етапів класичного проєктного менеджменту, але можна додавати і додаткові етапи, якщо того вимагає проєкт.

5 етапів традиційного менеджменту:

Етап 1. Ініціація. Керівник проєкту і команда визначають вимоги до проєкту. На даному етапі часто проводяться наради та «мозкові штурми», на яких визначається що повинен представляти із себе продукт проєкту.

Етап 2. Планування. На даному етапі команда вирішує, як вона буде досягати мети, поставленої на попередньому етапі. На даному етапі команда уточнює і деталізує цілі та результати проєкту, а також склад робіт по ньому. На підставі цієї інформації команда формує календарний план і бюджет, оцінює ризики і виявляє зацікавлені сторони.

Етап 3. Розробка. Дана стадія реалізується не для всіх проєктів – як правило, вона є частиною фази планування. У фазі розробки, характерною для технологічних проєктів, визначається конфігурація майбутнього проєкту та / або продукту і технічні способи його досягнення. Наприклад в IT-проєктах на даному етапі вибирається мова програмування.

Етап 4. Реалізація та тестування. На цій фазі відбувається власне основна робота по проєкту - написання коду, зведення будівлі тощо. Дотримуючись розробленими планами починає створюватися зміст проєкту, певне раніше, проводиться контроль за обраними метриками. У другій частині цієї фази відбувається тестування продукту, він перевіряється на відповідність вимогам Замовника і зацікавлених сторін. У частині тестування виявляються і виправляються недоліки продукту.

Етап 5. Моніторинг і завершення проєкту. Залежно від проєкту дана фаза може складатися з простої передачі Замовнику результатів проєкту або ж із тривалого процесу взаємодії з клієнтами щодо поліпшення проєкту і підвищенню їх задоволеності, і підтримки результатів проєкту. Останнє відноситься до проєктів в області клієнтського сервісу і програмного забезпечення.

Те, що описано вище - база, на якій будуються різні методи управління проєктами. Різних проєктів потрібні різні фази реалізації - декому достатньо і трьох фаз, іншим набагато більше. Іноді використовується так званий

«ітеративний водоспад», в якому кожен етап являє собою підпроєкт, в ході якого завдання реалізуються за фіксованими ітераціями. Але суть залишається одна - проєкт розбитий на етапи, які виконуються в строго певній послідовності.

Завдяки тому, що класичний проєктний менеджмент строго прив'язаний до часу виконання завдань, як правило, заздалегідь визначеному на етапі планування, для реалізації проєктів в рамках даного підходу відмінно підходять інструменти календарно-мережного планування. Найпоширенішим інструментом календарно-мережного планування є вже згадана раніше діаграма Ганта. Існує безліч інструментів для її побудови - від простих таблиць на кшталт Excel і Smartsheet до професійних програмних пакетів на кшталт Microsoft Project і Primavera.

Сильні сторони класичного проєктного менеджменту

Сьогодні досить часто йдеться про те, що класичний водоспадний підхід застарів, але він і не думає здавати позиції. Великим плюсом даного підходу є те, що він вимагає від Замовника і керівництва компанії визначити, що ж вони хочуть отримати, вже на першому етапі проєкту. Раннє включення привносить певну стабільність в роботу проєкту, а планування дозволяє впорядкувати реалізацію проєкту. Крім того, цей підхід має на увазі моніторинг показників і тестування, що абсолютно необхідно для реальних проєктів різного масштабу.

Потенційно, класичний підхід дозволяє уникнути стресів через наявність запасного часу на кожному етапі, закладеного на випадок будь-яких ускладнень і реалізації ризиків. Крім того, з правильно проведеним етапом планування, керівник проєктів завжди знає, якими ресурсами він володіє. Навіть якщо ця оцінка не завжди точна.

Слабкі сторони класичного проєктного менеджменту

Основна слабка сторона класичного проєктного менеджменту – нетолерантність до змін. Керівництво компанії Toyota, знамениту створенням таких систем як Lean і Kanban, часто критикують за те, що вони застосовують

класичний підхід в розробці програмного забезпечення для своєї компанії, причому саме за недолік гнучкості.

Оплот класичного підходу зараз – будівельні та інженерні проєкти, в яких вміст проєкту залишається практично незмінним протягом усього проєкту. Але якщо у проєкті ресурси і час не є ключовими обмеженнями, а зміст проєкту схильний до змін - варто придивитися до інших систем управління проєктами.

2.4.2. Клас методологій управління Agile

Як вже говорилося раніше - не всі проєкти можуть бути структуровані таким чином, щоб бути реалізованими за класичним проєктним підходом. Повертаючись до нашого прикладу з шеф-кухарем: приготування однієї страви ідеально лягає на «водоспадний» підхід, а ось вчасно приготувати і подати вечерю з чотирьох страв буде практично неможливо, якщо доведеться кожного разу чекати закінчення приготування однієї страви, щоб приступити до приготування іншого [34].

Для цієї задачі доцільно використовувати Agile – сімейство гнучких ітеративно-інкрементальних методів до управління проєктами та продуктами. Відповідно до даного підходу, проєкт розбивається нема на послідовні фази, а на маленькі підпроєкти, які потім «збираються» в готовий продукт. Схема роботи приведена на рисунку 5.

Таким чином, ініціація і верхньорівневі планування проводяться для всього проєкту, а наступні етапи: розробка, тестування та інші проводяться для кожного міні-проєкту окремо. Це дозволяє передавати результати цих міні-проєктів, так звані, інкремент, швидше, а приступаючи до нового підпроєкту (ітерації) в нього можна внести зміни без великих витрат і впливу на інші частини проєкту.

Рис. 2.3. Схема роботи за методологіями Agile

Незважаючи на те, що Agile увійшов в моду відносно недавно, ідея ітеративної розробки не нова. Свою нинішню назву сімейство гнучких методологій отримало в 2001 з публікації Маніфесту Agile (Agile Manifesto), закріпив основні цінності і принципи гнучкої розробки програмного забезпечення, в основі яких - командна робота і адаптація, навіть «любов» до змін.

Сам по собі Agile - не метод управління проєктами. Це скоріше набір ідей і принципів того, як потрібно реалізовувати проєкти. Вже на основі цих принципів і кращих практик були розроблені окремі гнучкі методи або, як їх іноді називають, фреймворки: Scrum, Kanban, Crystal, і багато інших. Ці методи можуть досить сильно відрізнятися один від одного, але вони йдуть одним і тим же принципам.

Сильні сторони Agile

Найголовніша перевага Agile – його гнучкість і адаптивність. Він може підлаштуватися під практично будь-які умови і процеси організації. Саме це зумовлює його нинішню популярність і то, скільки систем для різних областей було створено на його основі.

Один із принципів Agile: «Реакція на зміни важливіше проходження плану». Саме швидка і щодо безболісна реакція на зміни є причиною того, що багато великих компаній прагнуть зробити свої процеси більш гнучкими. Крім

того, Agile відмінно підходить для проєктів з «відкритим кінцем» - наприклад, запуску сервісу або блогу.

Вотчина Agile - розробка нових, інноваційних продуктів. У проєктах по розробці таких продуктів висока частка невизначеності, а інформація про продукт розкривається по ходу проєкту. В таких умовах реалізовувати проєкт по «водоспаду» стає неможливо-немає інформації для планування.

Слабкі сторони Agile

На відміну від PRINCE2 і PMBOK Agile - не є ні методологією, ні стандартом. Agile - це набір принципів і цінностей. Слабка сторона полягає в тому, що кожній команді доведеться самостійно складати свою систему управління, керуючись принципами Agile. Це непростий і тривалий процес, який потребує змін всієї організації, починаючи процедурами і закінчуючи базовими цінностями. Це тернистий шлях і не всім організаціям він під силу.

Цей шлях вимагає від лідера змін не тільки знань і наполегливості, а й серйозних адміністративних ресурсів, а також витрат. На щастя, існують готові набори практик, які полегшують Agile-трансформацію організації. До таких наборів відносяться фреймворк Scrum, метод Kanban і багато інших - Crystal, LeSS, SAFe, Nexus.

2.4.3. Методологія управління Scrum

Гнучкий фреймворк, створений в 1986 році, вважається самим структурованим з сімейства Agile. Створений в 1986 році, він поєднує в собі елементи класичного процесу і ідеї гнучкого підходу до управління проєктами. У підсумку вийшло дуже збалансоване поєднання гнучкості і структурованості.

Слідуючи заповітам Agile, Scrum розбиває проєкт на частини, які відразу можуть бути використані Замовником для отримання цінності, так званими зачатками продуктів (product backlog). І незважаючи на те, що «зачаток продукту» – досить вірний переклад і використовується у фаховій літературі, в

українській практиці найчастіше використовується просто «беклог». Потім пріоритет цих частин визначається Власником продукту - представником Замовника в команді. Найважливіші «шматочки» першими відбираються для виконання в спринті - так називаються ітерації в Scrum, що тривають від 2 до 4 тижнів. В кінці Спринту Замовнику видається робочий інкремент продукту – ті найважливіші «шматочки», які вже можна використовувати. Наприклад, сайт з частиною функціоналу або програма, яка вже працює, нехай і частково. Після цього команда проєкту приступає до наступного спринту. Тривалість у Спринту фіксована, але команда вибирає її самостійно на початку проєкту, виходячи з особливостей проєкту і власної продуктивності [34].

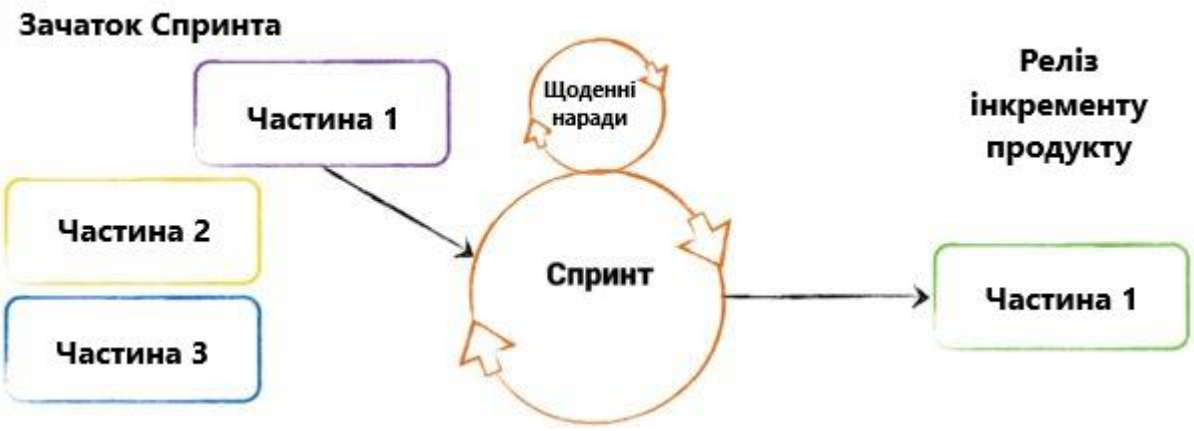

Рис. 2.4. Схема процесу методології Scrum

Щоб упевнитися в тому, що проєкт відповідає вимогам Замовника, які мають властивість змінюватися з часом, перед початком кожного Спринту відбувається переоцінка ще не виконаного змісту проєкту і внесення в нього змін. У цьому процесі беруть участь всі – команда проєкту, Scrum Майстер (Scrum Master, лідер команди проєкту) і Власник продукту. І відповідальність за цей процес лежить на всіх.

Як вже говорилося, Власник продукту є представником Замовника в проєкті, або уособлює всіх клієнтів майбутнього проєкту, в разі якщо Замовника немає. Для цього він повинен досконально знати їх потреби і спосіб мислення, а також розбиратися в продукті і технології його виготовлення.

Scrum Майстер покликаний допомогти учасникам проєкту краще зрозуміти і прийняти цінності, принципи і норми практики Scrum. Він лідер і посередник між зовнішнім світом і командою. Його завдання - стежити, щоб ніхто не заважав команді самостійно і комфортно працювати над поставленими завданнями. Команда ж відповідає за те, щоб в кінці спринту всі необхідні завдання були зроблені, а поставки – виконані.

Основна структура процесів Scrum обертається навколо 5 основних аспектів: упорядкування беклогу, планування Спринту, щоденних нарад, підведення підсумків Спринту і ретроспективи Спринту.

1. Зустріч з упорядкування беклогу (Backlog Refinement Meeting, «Backlog Grooming»): ця зустріч аналогічна фазі планування в класичному проєктному менеджменті, і проводиться в перший день кожного Спринту. На ній розглядається: що вже було зроблено за проєктом в цілому, що ще залишилося зробити і приймається рішення про те, що ж робити далі. Власник продукту визначає, які завдання на даному етапі є найбільш пріоритетними. Даний процес визначає ефективність Спринту, адже саме від нього залежить, яку цінність отримає Замовник за підсумками спринту.

2. Планування Спринту: Після того, як Власник продукту визначив пріоритети, команда спільно вирішує, що ж конкретно вони робитимуть під час прийдешньої ітерації, як досягти поставленої на попередній зустрічі мети. Команди можуть застосовувати різні інструменти планування і оцінки на даному етапі, аби вони не суперечили принципам і логіці Scrum. Планування Спринту проводиться на самому початку ітерації, після Зустрічі з упорядкування продукту.

3. Щоденні наради: Кожен день спринту, в ідеалі, в один і той же час, члени команди витрачають 15 хвилин на те, щоб поділитися інформацією про статус завдань і стан проєкту. На ній не відбувається обговорень проблем або прийняття рішень - якщо після зустрічі виникають питання і конфлікти, Scrum Майстер і залучені учасники обговорюють їх окремо. Летучка ж потрібна для обміну інформацією і підтримки всіх членів команди в курсі стану проєкту.

4. Підведення підсумків Спринту: Мета етапу - обстеження та адаптація створюваного продукту. Команда представляє результати діяльності всім зацікавленим особам. Основне завдання - переконатися, що продукт етапу відповідає очікуванням учасників і узгоджується з цілями проєкту.

5. Ретроспектива Спринту: Проводиться відразу після Підведення підсумків спринту і до планування наступного спринту. На ньому команда з'ясовує, наскільки чітко і злагоджено проходив процес реалізації етапу. Обстеженню піддаються виникли проблеми в роботі, методології та взаємодії. Саме цей етап дозволяє команді провести рефлексію і наступний Спринт провести ефективніше.

Багатьом Scrum може здатися складним для впровадження - новий процес, нові ролі, багато делегування і абсолютно нова організаційна структура. Але це гнучкий і при цьому структурований підхід до реалізації проєктів, який, на відміну від розмитих і загальних принципів Agile, не дозволить роботі піти не в те русло.

Сильні сторони Scrum

Scrum був розроблений для проєктів, в яких необхідні «швидкі перемоги» в поєднанні з толерантністю до змін. Крім того, цей фреймворк підходить для ситуацій, коли не всі члени команди мають достатній досвід в тій сфері, в якій реалізується проєкт - постійні комунікації між членами командами дозволяють брак досвіду або кваліфікації одних співробітників за рахунок інформації і допомоги від колег.

Онлайн телеканал Netflix є відмінним прикладом швидких поставок результатів. Сайт ресурсу оновлюється кожні два тижні завдяки Scrum, який не просто дозволяє працювати з високою швидкістю, але й акумулює призначений для користувача досвід і дає можливість виявити найголовніше для клієнтів.

В ході кожної ітерації, розробники додають і тестують нові функції сайту і прибирають ті, якими не користувалися клієнти. За словами команди Netflix, основна перевага Scrum в тому, що він дозволяє «швидко помилятися». Замість

того, щоб довго і з великими витратами готувати великий реліз, поставки раз в два тижні по Scrum мають невеликий розмір. Їх легко відстежувати і, якщо щось йде не так, швидко виправляти.

Слабкі сторони Scrum

Scrum дуже вимогливий до команди проєкту. Вона повинна бути невеликою (5-9 чоловік) і кросфункціональною - тобто члени команди повинні володіти більш ніж однією компетенцією, необхідної для реалізації проєкту. Наприклад розробник програмного забезпечення повинен володіти знаннями в тестуванні і бізнес-аналітиці. Робиться це для того, щоб частина команди не «простоювала» на різних етапах проєкту, а також для того, щоб співробітники могли допомагати і підміняти один одного.

Крім того, члени команди повинні бути «командними гравцями», активно брати на себе відповідальність і вміти самоорганізовуватися.

Scrum підходить не для всіх команд і організацій ще й тому, що пропонований процес може не підійти для розробки конкретного продукту - наприклад промислового верстата або спорудження будівлі.

2.4.4. Методологія управління Lean

Agile передбачає, що необхідно розбивати на невеликі керовані пакети робіт, але нічого не говорить про те, як управляти розробкою цього пакета. Scrum пропонує нам свої процеси і процедури. Lean ж, в свою чергу, додає до принципів Agile схему потоку операцій (workflow) для того, щоб кожна з ітерацій виконувалася однаково якісно [34].

У Lean, так само, як і в Scrum, робота розбивається на невеликі пакети поставки, які реалізуються окремо і незалежно. Але в Lean для розробки кожного пакета поставки існує потік операцій з етапами, подібними до тих, які були створені для проєкту Аполлон. Як і в класичному проєктному менеджменті, це можуть бути етапи планування, розробки, виробництва, тестування і поставки - або будь-які інші необхідні для якісної реалізації проєктів етапи.

Рис. 2.5. Схема роботи за методологією Lean

Етапи Lean і їх гнучкість дозволяють бути впевненими в тому, що кожна частина проєкту реалізується так, як потрібно. У Lean не прописані чіткі межі етапів, як в Scrum прописані обмеження спринті. Крім того, на відміну від класичного проєктного менеджменту, Lean дозволяє паралельно виконувати кілька завдань на різних етапах, що підвищує гнучкість і збільшує швидкість виконання проєктів.

Як і Agile, Lean це скоріше концепція, образ мислення, ніж щось висічене в камені. Використовуючи ідеї Lean, Ви можете самостійно створити систему, що задовольняє вашим вимогам в управлінні проєктами.

Сильні сторони Lean

Якщо Вам подобаються ідеї Agile, але проєкт вимагає дуже рівного якості і чіткого виконання, Lean надає набір інструментів для того, щоб задовольнити ці вимоги. Lean поєднує гнучкість і структурованість, як Scrum, але в трохи іншому ключі.

Слабкі сторони Lean

Не кожна частина проєкту вимагає однаково детальної і допитливою опрацювання та уваги. Але Lean передбачає саме такий підхід до кожного

завдання і етапу. Це основний мінус застосування Lean для великих і неоднорідних проєктів.

А ще, на відміну від Scrum, Lean не пропонує чіткого робочого процесу для реалізації «шматочків» проєкту, що сприяє розтягуванню термінів проєкту. Ця проблема може бути вирішена за допомогою ефективного керівництва і чітких комунікацій – головне пам'ятати про це.

2.4.5. Методологія управління Kanban

Lean виглядає абстрактним сам по собі, але в комбінації з Kanban його стає набагато простіше використовувати для побудови власної системи управління проєктами. Створений інженером компанії Toyota Тайічі Воно (Taiichi Ono) в 1953 році, Kanban дуже схожий на схему промислового виробництва. На вході в цей процес потрапляє шматочок металу, а на виході виходить готова деталь. Також і в Kanban, інкремент продукту передається вперед з етапу на етап, а в кінці виходить готовий до постачання елемент [34].

Крім того, творець Kanban надихався супермаркетами, а саме їх принципом – «тримай на полицях тільки те, що потрібно клієнту». А тому в Kanban дозволяється залишити незакінчену завдання на одному з етапів, якщо її пріоритет змінився і є інші термінові завдання. Невідредагована стаття для блогу, підвішена без дати публікації або частина коду функції, яку можливо не включатимуть в продукт – все це нормально для роботи за Kanban.

Kanban набагато менш суворий, ніж Scrum – він не обмежує час спринтів, немає ролей, за винятком власника продукту. Kanban навіть дозволяє члену команди вести кілька завдань одночасно, чого не дозволяє Scrum. Також ніяк не регламентовані зустрічі за статусом проєкту - можна робити це як Вам зручно, а можна не робити взагалі.

Для роботи з Kanban необхідно визначити етапи потоку операцій (workflow). У Kanban вони зображуються як стовпці, а завдання позначають спеціальні картки. Картка переміщається по етапах, подібно деталі на заводі, що переходить від верстата до верстата, і на кожному етапі відсоток

завершення стає вище. На виході ми отримуємо готовий до постачання замовнику елемент продукту. Дошка зі стовпцями і картками може бути як справжній, так і електронної – навіть тут Kanban не накладаються ніяких обмежень на користувачів.

Система Kanban може бути настільки гнучкою, наскільки проєктний менеджер забажає – адже багато в чому Kanban є візуалізацією ідеї Agile. Але у Kanban є 4 стовпи, на яких тримається вся система:

1. Картки: Для кожного завдання створюється індивідуальна картка, в яку заноситься вся необхідна інформація про завдання. Таким чином, вся потрібна інформація про завдання завжди під рукою.

2. Обмеження на кількість завдань на етапі: Кількість карток на одному етапі строго регламентовано. Завдяки цьому відразу стає видно, коли в потоці операцій виникає «затор», який оперативно усувається.

3. Безперервний потік: Завдання з беклогу потрапляють в потік в порядку пріоритету. Таким чином, робота ніколи не припиняється.

4. Постійне поліпшення: Концепція постійного поліпшення з'явилася в Японії в кінці XX століття. Її суть в постійному аналізі виробничого процесу та пошуку шляхів підвищення продуктивності.

Сильні сторони Kanban

Як і Scrum, Kanban добре підходить для досить згуртованих команди з хорошою комунікацією. Але на відміну від Scrum, в Kanban немає встановлених чітких дедлайнів, що добре підходить для мотивувати і досвідчених команд.

При правильному налаштуванні і управлінні, Kanban може принести велику користь команді проєкту. Точний розрахунок навантаження на команду, правильна розстановка обмежень і концентрація на постійне поліпшення - все це дозволяє Kanban серйозно економити ресурси і укладатися в дедлайни і бюджет.

Слабкі сторони Kanban

Часто можна чути, що по Kanban, на відміну від Scrum, можна працювати з практично будь-якою командою. Але це не зовсім так. Kanban найкраще підходить для команд, навички членів яких перетинаються один з одним. Таким чином вони можуть допомагати один одному долати труднощі при вирішенні завдань. Без цього Kanban буде не такий ефективний, як міг би бути. Також, як уже було сказано, Kanban краще підходить в тих випадках, коли немає жорстких дедлайнів. Для жорстких дедлайнів краще підходить класичний підхід або Scrum.

2.4.6. Методологія управління 6 Сігм (Six Sigma)

Компанія Motorola, поряд з Toyota, також внесла вклад в розвиток світового проєктного управління. Інженер цієї компанії Білл Сміт створив концепцію 6 сігм в 1986 році. Це більш структурована версія Lean ніж Kanban, в яку додано більше планування для економії ресурсів, підвищення якості, також зниження кількості браку і проблем [34].

Рис. 2.6. Схема роботи за методологією 6 Сігм

Кінцева мета проєкту - задоволення замовника якістю продукту, якого можна досягти за допомогою безперервного процесу поліпшення всіх аспектів проєкту, заснованому на ретельному аналізі показників. У концепції 6 сігм приділяється окрема увага усуненню виникає проблем.

Для цього було запропоновано процес з 5 кроків, відомих як DMEDI:

Визначення (Define): Перший етап дуже схожий на ранні етапи інших систем проєктного управління. На ньому визначається зміст проєкту, збирається інформація про передумови проєкту, ставляться цілі.

Вимірювання (Measure): 6 сігм орієнтована на збір і аналіз кількісних даних про проєкт. На даному етапі відбуваються визначається, які показники будуть визначати успіх проєкту і які дані потрібно збирати і аналізувати.

Дослідження (Explore): На стадії дослідження менеджер проєкту вирішує, яким же чином команда може досягти поставлених цілей і виконати всі вимоги в строк і в рамках бюджету. На даному етапі дуже важливо нестандартне мислення керівника проєктів при вирішенні проблем, що виникли.

Розробка (Develop): На даному етапі реалізуються плани і рішення, прийняті на попередніх етапах. Важливо розуміти, що на даному етапі необхідний детальний план, в якому описані всі дії, необхідні для досягнення поставлених цілей. Також на даному етапі вимірюється прогрес проєкту.

Контроль (Control): Ключовий етап в методології 6 сігм. Його основне завдання - довгострокове поліпшення процесів реалізації проєктів. Даний етап вимагає ретельного документування витягнутих уроків, аналізу зібраних даних і застосування отриманих знань як в проєктах, так у всій компанії в цілому.

6 сігм дуже схожа на Kanban, тільки з встановленими етапами реалізації завдань - плануванням, визначенням цілей і тестуванням якості. Найімовірніше, зустрічей команди при застосуванні 6 сігм буде значно більше, ніж при Kanban, але зате процес реалізації проєктів більш структурований і команді складніше збитися зі шляху. Як і Kanban, 6 сігм можна відносно легко адаптувати до потреб конкретної компанії або команди. Жорсткою вимогою є лише ретельне вимір і контроль показників проєкту на етапах реалізації - без цього неможливо постійне довгострокове поліпшення процесів реалізації проєкту.

Сильні сторони 6 сігм

Концепція 6 сігм надає чітку схему для реалізації проєктів і постійного поліпшення процесів. Визначаючи цілі, потім ретельно аналізуючи їх і переглядаючи ви отримуєте кількісні дані для більш глибокого розуміння проєкту та прийняття більш якісних рішень. І хоча збір, аналіз даних і витяг уроків можуть зайняти певний час, це дозволить поліпшити і оптимізувати процеси реалізації проєкту і заощадити таким чином ресурси в майбутньому.

6 сігм підходить для важких проєктів, в яких багато нових і складних операцій. Даний підхід дозволяє реалізовувати елементи проєкту, вчитися на помилках і підвищувати якість в майбутньому.

Слабкі сторони 6 сігм

Проблема 6 сігм в тому, нехай основною декларованою метою є зниження витрат і підвищення ефективності, але задоволення Замовника часто вивається на перший план. З огляду на деякі відмінності в цілях на різних етапах проєкту, часто у команд виникає плутанина в пріоритетах, і уникнути цього не просто. Це може демотивувати співробітників, що не відчувають задоволення від виконаної роботи. Крім того, якщо проєкт одиничний і компанія не планує в майбутньому реалізовувати подібні проєкти, всі витрати на аналіз і витяг уроків можуть виявитися марними.

Висновки до розділу 2

У цьому розділі дипломної роботи було проаналізовано історію управління проєктами розробки програмного забезпечення та процес управління розробкою програмного забезпечення, виділено та описано основні його аспекти. Виходячи із досвіду світової практики, було описано шість найрозповсюдженіших методологій управління проєктами, які забезпечують ефективний проєктний менеджмент, зокрема управління часом та ресурсами.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ УПРАВЛІННЯ ПРОЄКТОМ РОЗРОБКИ ПРОГРАМНОЇ СИСТЕМИ

3.1. Вимоги до програмної системи

Аналіз вимог – це процес вивчення потреб і цілей користувачів, класифікація і перетворення їх на вимоги до системи, апаратного і програмного забезпечення, встановлення і вирішення конфліктів між вимогами, визначення пріоритетів, меж системи і принципів взаємодії із середовищем функціонування [35].

Цілями процесу аналізу вимог є:

1) досягнення однакового розуміння замовниками, користувачами та розробниками, що повинна робити система;
2) надання розробникам якнайкращого розуміння вимог до системи;
3) визначення границь та обмежень системи;
4) визначення інтерфейсу користувача та системи.

3.1.1. Функціональні вимоги

Функціональні вимоги — вимоги, що описують поведінку системи й функції, які вона виконує, та залежать від типу системи, що розробляється, та потреб користувачів. Функціональні вимоги визначають фундаментальні операції, які повинні виконуватися програмним забезпеченням при прийнятті і обробці вхідних даних, обробці й генерації вихідних даних [36].

Система повинна виконувати наступні функції:

- Керування літальним апаратом:
  - Керування креном, тангажем, рисканням літака;
  - Керування зльотом та посадкою літака;
  - Керування прискоренням та сповільненням літака;
  - Керування внутрішніми системами літака.
- Обчислення та відображення льотної інформації;

- Створення тестування:
  - Додавання та видалення етапів тестування (льотних подій);
  - Зміна порядку етапів тестування;
  - Задавання тривалості тестування.
- Редагування параметрів етапів тестування:
  - Тривалість льотної події;
  - Ступінь впливу льотної події;
  - Критичність льотної події.
- Зміна критеріїв оцінювання:
  - Змінювати кількість балів, що нараховуються або вираховуються на кожному етапі тестування;
  - Задавання умов правильного та неправильного виконання кожного етапу тестування та діапазон відхилення (похибку).
- Оцінювання:
  - Обчислення оцінки за кожний етап тестування;
  - Обчислення результуючої оцінки з урахуванням оцінок за усі етапи тестування.
- Керування обліковими записами користувачів:
  - Реєстрація нового користувача;
  - Авторизація користувача в системі;
  - Перегляд облікових даних користувачів.
- Збереження та резервне копіювання даних користувачів.

В системі передбачено два типи користувачів:

- Звичайний користувач (абітурієнт, що проходить тестування);
- Адміністратор (член комісії оцінювання).

Для аналізу вимог широко застосовуються засоби уніфікованої мови моделювання UML: діаграми варіантів використання, компонентів та розгортання.

Діаграми варіантів використання, або прецедентів (Use Case Diagram) застосовуються для відображення функціональності програмного забезпечення, границь та контексту предметної області, для якої призначено програмне забезпечення, та взаємодії програмного забезпечення із зовнішнім середовищем.

Графічними компонентами діаграми варіантів використання є дійові особи (актори), варіанти використання та зв'язки між ними.

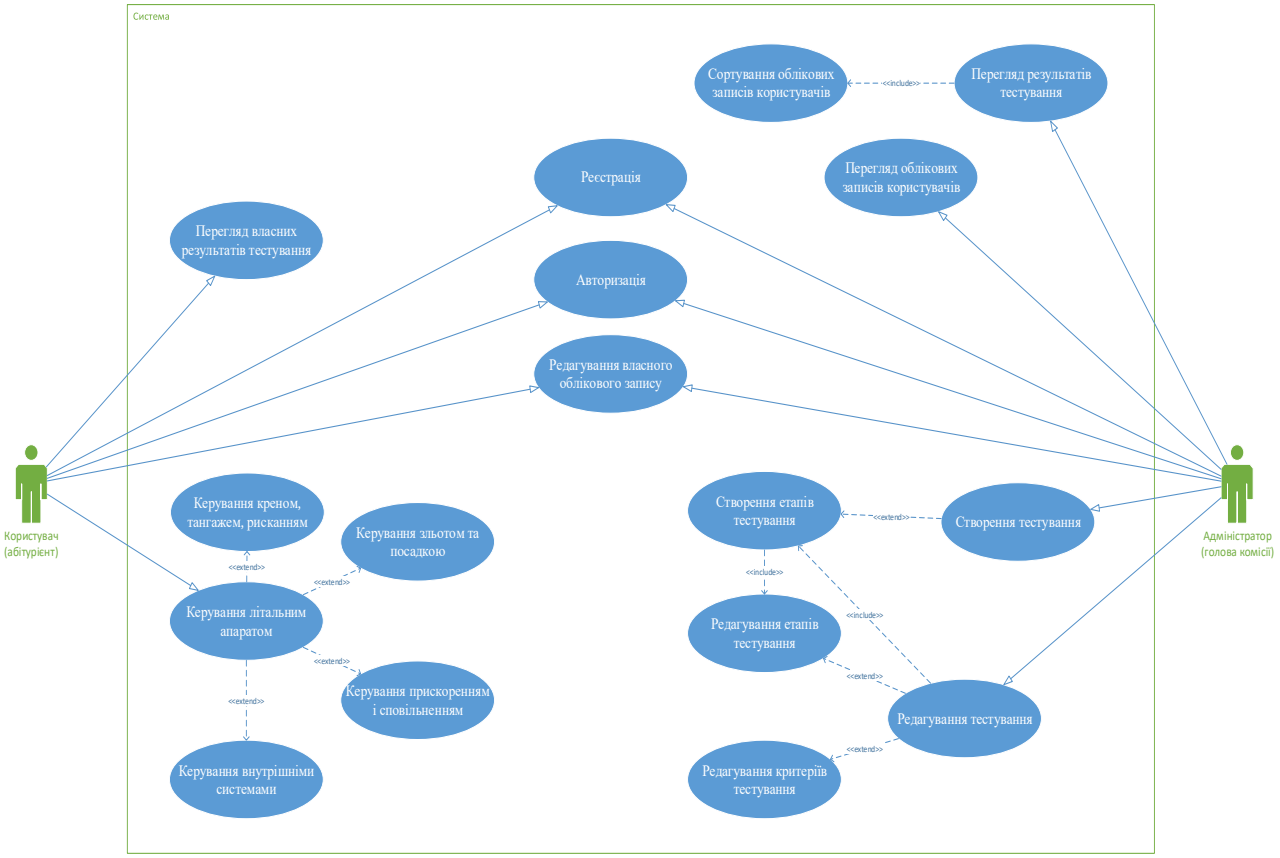

Рис. 3.1. Діаграма варіантів використання

3.1.2. Нефункціональні вимоги

Нефункціональні вимоги — вимоги, що відображають обмеження, пов'язані з функціонуванням системи. Серед нефункціональних вимог виділяють вимоги до зовнішніх інтерфейсів та продуктивності, проєктні обмеження та атрибути.

Вимоги до системи:

- Система має бути написана на C# з використанням Unity API;
- Система має адаптувати зображення під будь-яку роздільну здатність екрану;
- Простий та інтуїтивно зрозумілий інтерфейс;
- Зручність експлуатації системи користувачем;
- Швидка реакція на дії користувача;
- Висока швидкість обробки інформації;
- Швидкість оновлення зображення (фреймрейт) має складати не менше 30 кадрів/сек;
- Максимально можлива надійність та безвідмовність системи;
- Вихідними даними системи повинні бути чітко форматовані документи, що відповідають стандартам звітної документації;
- Вихідні дані системи повинні бути захищені від будь-якої модифікації користувачами.

3.1.3. Системні вимоги

Системні вимоги включають у себе вимоги до програмного й апаратного забезпечення комп'ютера, що виконуватиме цей програмний застосунок.

Мінімальні системні вимоги:

- Операційна система: Windows 7 SP1 або новіша;
- Процесор: Intel Core i3 3-го покоління або AMD Athlon II X3;
- Графічний прискорювач: Nvidia GeForce GTX 650 або AMD Radeon HD 6850;
- Об'єм оперативної пам'яті: 2 GB DDR3.
- 500 Mb вільного простору на жорсткому диску.

Рекомендовані системні вимоги:

- Операційна система: Windows 7 SP1 або новіша;
- Процесор: Intel Core i5 4-го покоління або AMD FX-6300;

- Графічний прискорювач: Nvidia GeForce GTX 960 або AMD Radeon R9 380;
- Об’єм оперативної пам’яті: 4 GB DDR4.
- 500 Mb вільного простору на жорсткому диску.

3.2. Документи стартап-проєкту

3.2.1. Інформаційна карта проєкту

1. Назва проєкту	Авіасимулятор аварійних ситуацій для професійного відбору майбутніх пілотів

2. Автор проєкту	Воловін Євгеній Олексійович

3. Коротка анотація	Проєкт призначений для забезпечення льотних навчальних закладів програмним інструментом оцінювання психофізичних якостей абітурієнтів для відбору тих людей, хто найбільше підходить на професію пілота.

4. Термін реалізації проєкту	8 місяців

5. Необхідні ресурси	Матеріально-технічні: 1. Робочі станції (персональні комп’ютери, смартфони) – 120,000 грн. 2. Багатофункціональний мережевий принтер – 15,000 грн. 3. Меблі – 25,000 грн. 4. Посуд – 1,000 грн. 5. Кавомашина – 16,000 грн. Трудові: 1. Співробітники проєкту – 75,000 грн/міс. Інтелектуальні: 1. Ліцензійне програмне забезпечення – 1,550 грн/міс. Фінансові: 1. Обчислювальне обладнання: 120,000 грн на обчислювальну техніку 2. Додаткове обладнання:

	42,000 грн на меблі та предмети побуту 3. Ліцензія на програмні продукти – 1,550 грн/міс. 4. Зарплата персоналу – 75,000 грн/міс. 5. Комунальні витрати – 4,000 грн/міс.
6. Опис проблеми, яку вирішує проєкт	Аналіз авіакатастроф із «чорних скриньок» свідчить про те, що основним чинником є помилки пілотів або авіадиспетчерів. Для того, щоб пілоти могли прийняти правильне рішення в аварійних ситуаціях під час польоту, вони повинні мати високий рівень професійної підготовки та бути психологічно й психічно стійкими до стресових умов. Як можна зрозуміти, не кожна людина має такі психофізичні якості, а отже, не кожна людина повною мірою підходить цій професії. Кожний майбутній пілот спочатку проходить своє навчання у льотному навчальному закладі, а при вступі абітурієнтів їхні психофізичні якості майже не піддаються аналізу. Саме тому актуальною задачею є розробка програмної системи – інструменту об'єктивного оцінювання психофізичних якостей абітурієнтів.

7. Головні цілі та завдання проєкту	Ціль: Розробити програмний продукт – додаток для персонального комп'ютера, використовуючи програмну платформу та ігровий движок Unity. Завдання: Забезпечити льотні навчальні заклади спеціальним програмним засобом для оцінювання психофізичних якостей абітурієнтів.

8. Очікувані результати
Реалізація проєкту забезпечить підвищення безпеки польотів шляхом зниження кількості авіакатастроф, спричинених людським фактором, а саме невідповідністю індивідуальних психофізичних якостей пілотів цій професії та особистих характеристик, що вона вимагає.

3.2.2. Організаційна структура компанії

Для підприємства обрано організаційно-правову форму товариство з обмеженою відповідальністю (ТОВ).

По-перше, ТОВ може не вести ретельний бухгалтерський облік, якщо вибере спрощену систему оподаткування.

По-друге, створити ТОВ на даний час простіше, ніж будь-яку іншу організацію.

По-третє, за боргами ТОВ бізнесмен буде відповідати в обмеженому обсязі – в межах статутного капіталу організації, який рідко виявляється більше за мінімально необхідної державної суми.

Тобто власник такої фірми завжди знає усі ризики та збитки, які може понести у разі банкротства.

Рис. 3.2. Організаційна структура компанії

3.2.3. Структура команди та завдання проєкту

1. Склад команди

1) Розробник – Виконавець (В)

2) Проектувальник – Генератор ідей (Г)

3) Team Leader – Дипломат (Д)

2. Набір завдань

№ п/п	Завдання	Об’єм (міс)
1	Розробка ТЗ	1.5
2	Пошук інвесторів	1
3	Розподіл обов’язків	0.5
4	Забезпечення ресурсами	8
5	Розробка ПЗ	6
6	Тестування ПЗ	2
7	Аналітика	1
8	Юридичні дії	4
9	Рекламна кампанія	2

3. Взаємодія

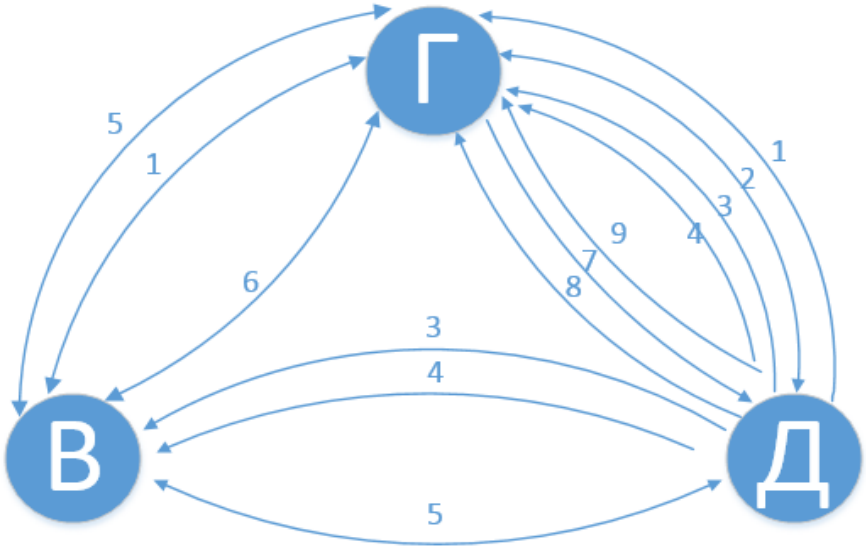

Рис. 3.3. Схема взаємодії між членами команди

В = Вхід/Вихід = 5/4 = 1.25

Г = Вхід/Вихід = 9/5 = 1.8

Д = Вхід/Вихід = 4/8 = 0.5

4. Завантаженість

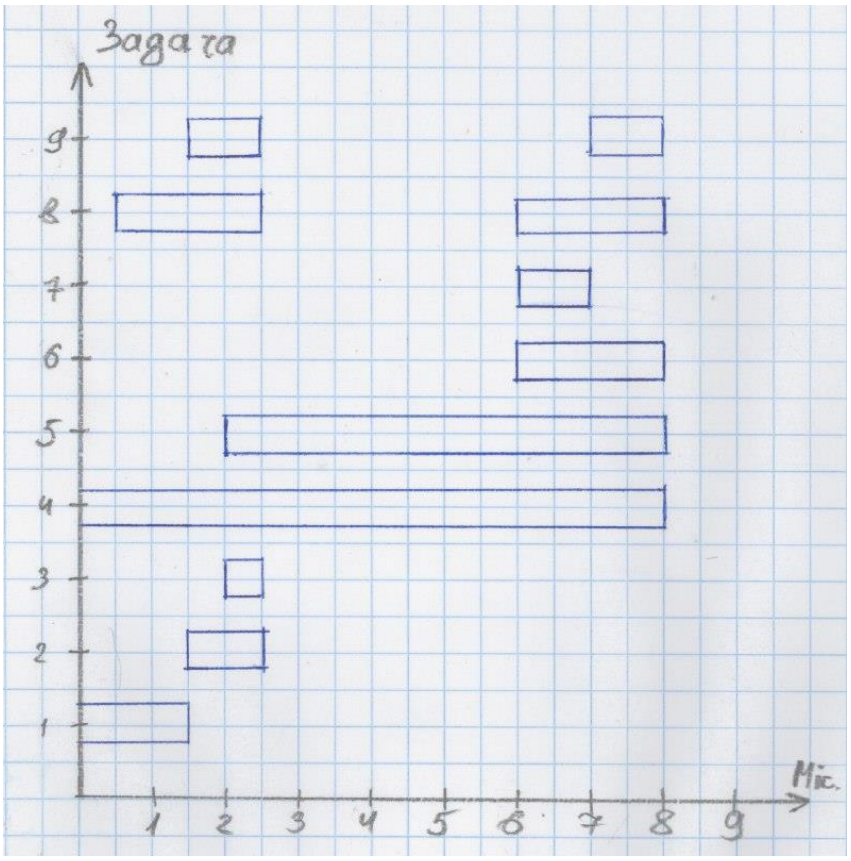

Рис. 3.4. Завантаженість завданнями

Період активності:

В = 0.5 + 3 + 1 = 4.5

Г = 0.5 + 0.5 + 4 + 3 + 1 = 9

Д = 0.5 + 0.5 + 0.5 + 4 + 1 + 4 + 2 = 13.5

Доля періоду активності:

В = 4.5 / 8 = 0,5625

Г = 9 / 8 = 1,125

Д = 13.5 / 8 = 1,6875

Завантаженість:

В = 1.25 / 0,5625 = 2.22

Г = 1.8 / 1,125 = 1.6

Д = 0.5 / 1,6875 = 0.29

5. Особистий внесок

Фактор	Вага	В	Г	Д
Ідея	6	3	7	6
Підготовка бізнес плану	5	3	4	7
Компетентність	7	5	7	6
Залученість і ризики	5	3	4	8
Обов'язки	7	6	7	7

6. Доля особистого внеску

Фактор	В	Г	Д
Ідея	18	42	36
Підготовка бізнес плану	15	20	35
Компетентність	35	49	42
Залученість і ризики	15	20	40
Обов'язки	42	49	49
Разом	110	180	202
Відсоток	22.36	36.59	41.05

3.2.4. Опис продукту проєкту

1. Морфологічна карта

- Персональні комп’ютери – основна платформа розробки.
- Android- та iOS-смартфони – потенційні платформи для розробки.
- Windows 7, Windows 10 – основні операційні системи.
- Android, iOS – потенційні операційні системи.
- Unity Engine API – основне API для розробки.
- Unity Editor, MS Visual Studio – основні IDE для розробки.
- Jetbrains IntelliJ Idea, Android Studio – потенційні IDE для розробки.
- USB-накопичувачі – основний спосіб розповсюдження.
- App Store, Google Play – потенційні способи розповсюдження.

Товар за задумом:

Додаток для персонального комп’ютера, що використовує ігровий движок Unity для симуляції простору та польоту різних 3D-моделей літальних апаратів та симуляції аварійних ситуацій для тестування психофізичних реакцій абітурієнтів льотних навчальних закладів.

Товар у реальному виконанні:

Додаток для персонального комп’ютера, що використовує ігровий движок Unity для симуляції простору та польоту різних 3D-моделей літальних апаратів та симуляції аварійних ситуацій для тестування психофізичних реакцій абітурієнтів льотних навчальних закладів для визначення професійної придатності абітурієнтів до професії пілота.

Товар з підкріпленням:

Крос-платформний додаток, що використовує ігровий движок Unity для симуляції простору та польоту різних 3D-моделей літальних апаратів та симуляції аварійних ситуацій у різних просторових та погодних умовах для тестування психофізичних реакцій абітурієнтів льотних навчальних закладів для визначення професійної придатності абітурієнтів до професії пілота.

2. Еволюція MVP

Головна проблема: льотні навчальні заклади потребують програмну систему (додаток для персонального комп’ютера), завдяки якому матимуть змогу проводити об’єктивне тестування психофізичних якостей абітурієнтів для визначення їхньої професійної придатності.

MVP1. Можливість проведення тестування лише шляхом опитування абітурієнтів та спостереження за їхньою реакцією.

MVP2. Проведення тестування на спеціалізованих літальних тренажерах у спеціально відведених великогабаритних приміщеннях.

MVP3. Проведення тестування за допомогою розроблюваної програмної системи у майже будь-якому приміщенні льотного навчального закладу з персональними комп’ютерами.

MVP4. Відображення 3D-сцени польоту літальних апаратів за допомогою технології віртуальної реальності. Використовуючи спеціальний VR-шолом, абітурієнти зможуть відчути більш глибоко «ефект присутності» для максимальної стимуляції нервової системи.

MVP5. Відображення 3D-сцени польоту літальних апаратів прямо у мозку абітурієнта. У майбутньому, із розвитком біоінженерії стане можливо імплементувати механізм передачі цифрового зображення та інформації у свідомість людини, із цим відпаде потреба використовувати персональні комп’ютери та мобільні пристрої.

3. План оновлення

Опрацювання питань для удосконалення продукту «Авіасимулятор аварійних ситуацій для професійного відбору майбутніх пілотів».

№ п/п	Запитання	Відповідь
1	**2**	**3**
1	Частиною яких систем є продукт?	Частиною програмних систем та комплексу програмних додатків.
2	Які функції може виконувати продукт? Як їх з ним пов’язати?	Продукт виконує тестувальну функцію. Він симулює політ літальних апаратів у просторі та аварійні ситуації, що виникають під час польоту, які призводять до несправності літальних апаратів.
3	Чи можна розділити продукт на частини?	Підсистема графічного інтерфейсу, бізнес-логіки та обробки даних.
4	Чи можна об’єднати (агрегувати) кілька елементів продукту в один?	Систему симуляції 3D-простору та літальних апаратів можна поєднати воєдино із системою оцінки коректності дій абітурієнта.
5	Яким має бути ідеальний продукт?	Система має максимально повну базу даних про літальні апарати, місцевості погодні умови та механічний вплив несправностей на системи літального апарату, безвідмовно та максимально точно переносить 3D-моделі у простір, оброблює дані максимально швидко.
6	Чим можна замінити цей продукт?	Тестуванням на спеціалізованих льотних тренажерах.
7	Яким цей продукт був у минулому?	Опитування, письмове тестування абітурієнтів.
8	На розвиток яких функцій було спрямоване удосконалення продукту?	На підвищення точності розміщення 3D-моделей у просторі, швидкості симуляції та обробки даних.
9	Які функції залишилися «недорозвиненими»	Функція реалістичного знищення літального апарату відповідно до результату впливу усіх фізичних сил.
10	Як можна натепер розвинути ці функції?	Реалізувати підсистему модульно-компонентного руйнування частин літального апарату.

3.2.5. Визначення специфіки продукту проєкту

Ідея

Ідея 1. Використання персональних комп’ютерів

Ідея 2. Розробка програмної системи

Ідея 3. Використання смартфонів

Ідея 4. Розробка мобільного додатку

Ідея 5. Персональний комп’ютер

Ідея 6. Крос-платформа

Ідея 7. Мобільні платформи

Ідея 8. Використання вбудованих датчиків

Ідея 9. Підключення спеціальних датчиків

Агрегування

Агрегування 1. Програмна система для персонального ком’ютера.

Агрегування 2. Мобільний додаток для смартфонів з вбудованими датчиками.

Агрегування 3. Крос-платформна програмна система з підключенням спеціальних датчиків.

Синхронізація завдань

Етапи	Продукти (послідовність заміщення)	
Минуле століття	Тестування шляхом опитування абітурієнтів	Тестування на спеціалізованих льотних тренажерах
Сьогодні	Ідея 1: Використання персональних комп’ютерів	Агрегування 1: Програмна система для персонального ком’ютера
Завтра	Ідея 6: Крос-платформа	Агрегування 3: Крос-платформна програмна система з підключенням спеціальних датчиків

3.2.6. Розроблення ринкової стратегії проєкту

Таблиця 3.1. Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1	Льотні навчальні заклади	Висока – 90%	Високий – 85% Велика к-сть користувачів забезпечить високий попит	Низька – 20%	Легко Продукт є безкошто-вним для абітурієн-тів
Які цільові групи обрано: Льотні навчальні заклади					

Таблиця 3.2. Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проєкту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку**
1	Розвиток технології віртуальної реальності	Власна або залучена система збуту	Перехід у режим відображення у віртуальній реальності	Стратегія зростання
2	Розвиток мобільних платформ	Власна або залучена система збуту	Перехід на крос-платформний режим, швидкодія	Стратегія зростання

Таблиця 3.3. Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проєкт «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні х-ки товару конкурента, і які?*	*Стратегія конкурентної поведінки*
	Ні, подібні рішення є у деяких конкурентів	Продуктом користуватиметься вузька група користувачів – абітурієнти льотних навчальних закладів	Лише концепція та фізичні моделі польоту літальних апаратів	Стратегія заняття вільного сегменту

Таблиця 3.4. Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проєкту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проєкту*
1	Точність тестування абітурієнтів	Стратегія зростання	Адаптивність, врахування перешкод	Точність Адаптивність Інтерактивність
2	Швидкодія	Стратегія зростання	Розкривати в повній мірі можливості апаратної платформи	Швидкість роботи Плавність анімацій Функціональність

3.2.7. Розроблення маркетингової програми стартап-проєкту

Таблиця 3.5. Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами*
1	Безпека	Підвищення безпеки польотів	Програмна система для об’єктивного оцінювання професійної придатності абітурієнтів
2	Тестування	Професійний відбір найбільш відповідних людей	Застосування програмної системи льотними навчальними закладами

Таблиця 3.6. Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Додаток для персонального комп’ютера</td></tr>
<tr><td rowspan="5">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Програмна система для ПК
2. Інтерактивність</td><td>Нм
Нм</td><td>Тл
Тл
Тл</td></tr>
<tr><td colspan="3">Якість: відповідність вимогам льотних НЗ</td></tr>
<tr><td colspan="3">Пакування – USB накопичувач в упаковці</td></tr>
<tr><td colspan="3">Марка: Volovin EA | Авіасимулятор аварійних ситуацій</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу — розробка та здача продукту замовнику</td></tr>
<tr><td colspan="3">Після продажу — отримання орендних виплат</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання:
Закони про захист інтелектуальної власності.</td></tr>
</table>

Таблиця 3.7. Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
	$60 за копію	$60 за копію	Не має значення	2 млн грн

Таблиця 3.8. Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1	Користування льотними навчальними закладами	Тестування абітурієнтів на конкурсній основі	Рівень 1 – через мережу льотного НЗ	Через USB накопичувачі

Таблиця 3.9. Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціонування*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
	Абітурієнти льотних навчальних закладів	Месенджери	Точність; Зручність використання; Якість результату	Залучення клієнтів, демонстрація роботи та можливостей	Реклама у МОН та льотних навчальних закладах

3.2.8. Аналіз ринкових можливостей запуску стартап-проєкту

Таблиця 3.10. Попередня характеристика потенційного ринку стартап-проєкту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	2
2	Загальний обсяг продаж, грн/ум.од	$20000
3	Динаміка ринку (якісна оцінка)	зростає
4	Наявність обмежень для входу (вказати характер обмежень)	К-сть абітурієнтів на час вступної кампанії
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	20%

Таблиця 3.11. Характеристика потенційних клієнтів стартап-проєкту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
	Безпека польотів	Льотні навчальні заклади	Психофізичні якості абітурієнтів	Точність Якість Швидкодія

Таблиця 3.12. Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1	Конкуренти	Вихід на ринок продукту від крупного конкурента	Доповнення додатку новим функціоналом
2	Низька к-сть клієнтів	Обсяг доходів від продажів залежить від к-сті льотних НЗ	Активізація рекламної кампанії – переконання більшої к-сті льотних НЗ
3	Патенти	Виявлення патентів, що заважають реалізації	Відмова від частини функціоналу, реалізація з урахуванням патентів конкурентів

Таблиця 3.13. Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1	Закрите бета-тестування	Бета-тестування з великою к-стю користувачів дозволить виявити некоректну роботу системи	Аналіз відгуків про роботу продукту дозволить покращити її
2	Ціна	Продукт є безкоштовним для абітурієнтів	Безкоштовність забезпечить максимально високе число використань додатку

Таблиця 3.14. Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
Тип конкуренції - монополістична	Галузь є конкурентною. Будь-який виробник може запропонувати свій товар для цього ринку.	В разі необхідності розширювати функціонал продукту унікальними функціями
Рівень конкурентної боротьби - національний	Можливий міжнародний продаж додатку	Входити з продуктом на ринок різних країн
Галузева ознака - міжгалузева	Використовуються технології програмної платформи Unity 3D, апаратного забезпечення	Використання сучасних технологій
Конкуренція за видами товарів: - товарно-родова	Конкуренція з подібними програмними системами	Демонстрація можливостей доповненої реальності як найголовнішу особливість
За характером конкурентних переваг - нецінова	Основна ставка на технологію Unity 3D	Максимально розкривати переваги та можливості технології
За інтенсивністю - не марочна	Абсолютно новий продукт	Рекламувати компанію-розробника та продукт

Таблиця 3.15. Аналіз конкуренції в галузі за М. Портером

Прямі конкуренти в галузі	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
Microsoft, Laminar Research Досить сильна конкурента боротьба	MS Flight Simulator, X-Plane 11 Основним бар'єром є к-сть льотних НЗ	Постачальники ПЗ можуть поставити зависоку ціну на ліцензійне ПЗ	Можлива проблема з льотними НЗ, які необхідно переконати в ефективності системи тестування	Факторів загроз немає, це є програмним продуктом

Таблиця 3.16. Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспромо жності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проєктів значущим)*
1	*Функціональність*	Інтерактивна взаємодія з користувачами
2	*Швидкодійність*	Продукти конкурентів більш вимогливі до апаратної частини ПК, не мають значної частини функціоналу для тестування

Таблиця 3.17. Порівняльний аналіз сильних та слабких сторін «Авіасимулятор аварійних ситуацій»

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з Volovin EA*						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	*Функціональність*	15				*			
2	*Швидкодійність*	18							*

Таблиця 3.18. SWOT- аналіз стартап-проєкту

Сильні сторони: Функціональність Швидкодійність	Слабкі сторони: Орієнтування на відносно невелику к-сть льотних навчальних закладів
Можливості: Закрите бета-тестування	Загрози: Невелика к-сть користувачів

Таблиця 3.19. Альтернативи ринкового впровадження стартап-проєкту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1	Доповнення продукту новим функціоналом	Висока	2-3 міс
2	Активізація рекламної кампанії	Середня	1-2 міс

3.2.9. Виробничий план проєкту

Таблиця 3.20. Календарний план-графік реалізації стартап-проєкту

№ п/п	Етапи реалізації	Період реалізації проєкту						
		0-й рік				1-й рік	2-й рік	3-й рік
		1-й кв.	2-й кв.	3-й кв.	4-й кв.			
1.	Проведення НДДКР	*						
2.	Розробка проєктних матеріалів і ТЕО	*	*					
3.	Робоче проєктування та прив'язка проєкту	*	*					
4.	Створення компанії	*						
5.	Придбання нематеріальних активів, отримання дозвільних документів тощо	*						
6.	придбання й оренда земельних ділянок, будівель, приміщень, споруд	*						
7.	Придбання обладнання, устаткування та пристроїв	*						
8.	Передвиробничі маркетингові дослідження	*		*				
9.	Приймально-здавальні випробування			*				
10.	Пусконалагоджувальні роботи			*				
11.	Освоєння проєктних потужностей				*			
12.	Придбання матеріальних ресурсів	*	*	*				
13.	Запуск виробництва			*				
14.	Продаж продукції				*	*	*	*

Таблиця 3.21. Планова потреба у виробничих площах

№ з/п	*Тип приміщення (будівлі, ділянки, споруди)*	*Кількість одиниць*	*Площа, кв. м*	*Вимоги до приміщення (будівлі, ділянки, споруди)*	*Умови надання*	*Вартість, тис. грн.*
1.	Офіс	1	50	-	1 рік	150
…						
Разом				—	—	150

Таблиця 3.22. Планова потреба у виробничому обладнанні та устаткуванні

№ з/п	*Вид обладнання (устаткування, пристрою)*	*Тип (модель)*	*Виробник обладнання (устаткування, пристрою)*	*Терміни постачання*	*Вартість, тис. грн.*
1.	*ПК*	*Офісний*	*Особиста збірка*	*1 міс*	*10*
2.	*ПК*	*Потужний*	*Особиста збірка*	*1 міс*	*30*
3.	*ПК*	*Потужний*	*Особиста збірка*	*1 міс*	*30*
4.	*Смартфон на iOS*	*iPhone XS*	*Apple*	*1 міс*	*25*
5.	*Смартфон на Android*	*Galaxy S10+*	*Samsung*	*1 міс*	*25*
6.	*Кавомашина*	*EP5310/10 Series 5000*	*Philips*	*1 міс*	*16*
7.	*Меблі*	*Меблева стінка, столи, крісла*	*IKEA*	*2 міс*	*25*
Разом:		—	—	—	*161*

Таблиця 3.23. Плановий обсяг виробництва продукції стартап-проєкту

Вид продукції	*Одиниця виміру*	*Обсяги виробництва за період*		
		1-й рік	*2-й рік*	*3-й рік*
Додаток для ПК	Купівля	10	40	90

Таблиця 3.24. Планова потреба у матеріальних ресурсах та комплектуючих

№ з/п	*Вид ресурсу*	*Одиниця виміру*	*Витрати на одиницю продукції*	*Вартість на одиницю продукції, грн.*	*Вартість за плановим обсягом виробництва за період, грн.*		
					1-й рік	*2-й рік*	*3-й рік*
1.	Ліцензії на користування	—	—	—	—	—	—
1.1.	MS Visual Studio Professional 2019	**шт/рік**	**1**	**13500**	**13500**	**13500**	**13500**
1.2.	Аккаунт розробника App Store	**шт/рік**	**1**	**2500**	**2500**	**2500**	**2500**
1.3.	Аккаунт розробника Google Play	**шт/рік**	**1**	**2500**	**2500**	**2500**	**2500**
Всього ліцензій		—	—		**18500**	**18500**	**18500**

Таблиця 3.25. Планова потреба та витрати на персонал

№ з/п	*Категорія персоналу*	*Чисельність*	*Заробітна плата, грн. на місяць*	*Відрахування на соціальні заходи, грн. на місяць*	*Витрати на оплату праці за період, грн.*		
					1-й рік	*2-й рік*	*3-й рік*
1.	Адміністративний персонал (працівники апарату управління)						
1.1	Team Leader	1	15000	5610	210000	210000	210000

№ з/п	Категорія персоналу	Чисельність	Заробітна плата, грн. на місяць	Відрахування на соціальні заходи, грн. на місяць	Витрати на оплату праці за період, грн.		
					1-й рік	2-й рік	3-й рік
2.	Промислово-виробничий персонал						
2.1.	Розробник	1	10000	3740	150000	150000	150000
2.2.	Конструктор	1	12500	4675	180000	180000	180000
Разом:		3	37500	7500	540000	540000	540000

Таблиця 3.26. Загальні початкові витрати проєкту

№ з/п	Стаття витрат	Обсяги витрат в 0-й рік, тис. грн.
1.	Проведення НДДКР	50
2.	Розробка проєктних матеріалів і ТЕО	40
3.	Робоче проєктування і прив'язка проєкту	30
4.	Витрати на придбання й оренду земельних ділянок, будівель, приміщень, споруд	150
5.	Витрати на придбання обладнання та устаткування та пристроїв	161
6.	Витрати на приймально-здавальні випробування	25
7.	Витрати на пусконалагоджувальні роботи	10
8.	Комплексне освоєння проєктних потужностей	20
9.	Витрати на придбання нематеріальних активів	18,5
10.	Одноразові виплати, зокрема гарантуючим і страховим організаціям	30

№ з/п	*Стаття витрат*	*Обсяги витрат в 0-й рік, тис. грн.*
11.	Витрати на створення оборотного капіталу, необхідного для початку операційної діяльності (створення виробничих запасів, передоплата сировини, матеріалів і комплектуючих виробів, які мають бути поставлені на початку операційної діяльності)	100
12.	Податкові платежі (земельний, комунальний та інші), здійснені до початку операційної діяльності	10
13.	Оплата юридичних послуг	50
14.	Витрати на передвиробничі маркетингові дослідження і створення збутової мережі	70
15.	Витрати, пов'язані з діяльністю персоналу	1080
	Разом:	1844,5

Таблиця 3.27. Планові загальногосподарські витрати

№ п/п	*Стаття витрат*	*Витрати за період, тис. грн.*		
		1-й рік	*2-й рік*	*3-й рік*
1.	Витрати на оренду земельних ділянок, будівель, приміщень, споруд	150	150	150
2.	Витрати на обладнання, устаткування та пристрої	30	30	30
3.	Витрати на придбання нематеріальних активів	18,5	18,5	18,5
4.	Витрати на персонал (на відрядження, соціальні заходи тощо)	20	20	20
5.	Витрати на зв'язок	10	10	10
6.	Витрати на паливо та електроенергію	30	30	30
7.	Витрати на водопостачання	4	4	4
8.	Витрати на утримання обладнання та приміщень	10	10	10
9.	Витрати на збут	5	5	5
10.	Витрати на просування та рекламу	50	50	50
11.	Оплата юридичних послуг	50	50	50
12.	Податкові платежі (земельний, комунальний податки, інші)	15	15	15
	Разом:	402,5	402,5	402,5

3.3. Прототип програмної системи

3.3.1. Інструментальні засоби для розробки системи

Для розробки програмної системі в даній дипломній роботі було використано програмну платформу Unity, інтегроване середовище розробки Microsoft Visual Studio 2019 зі спеціальним плагіном для інтеграції з Unity та мову програмування C# як скриптову мову Unity.

Unity - це кросплатформовий ігровий движок, розроблений Unity Technologies, вперше анонсований і випущений в червні 2005 року на Всесвітній конференції розробників Apple Inc. як ексклюзивний ігровий движок для Mac OS X, а згодом – і для Microsoft Windows. З 2018 року движок був розширений для підтримки більше 25 платформ. Движок можна використовувати для створення тривимірних, двовимірних, віртуальних ігор та ігор з доповненою реальністю, а також для моделювання та інших подій. Цей движок був прийнятий в галузях, не пов'язаних з відеоіграми, таких як кіно, автомобілі, архітектура, інженерія та будівництво [37].

Unity дає користувачам можливість створювати ігри і досвід як в 2D, так і в 3D, а движок пропонує основний API сценаріїв в C # як для редактора Unity в формі плагінів, так і для самих ігор, а також для функції перетягування. До того, як C# був основною мовою програмування, використовуваним для движка, він раніше підтримував Boo, який був вилучений з випуском Unity 5, і версію JavaScript під назвою UnityScript, яка була оголошена застарілою в серпні 2017 року, після випуску Unity 2017.1, на користь C#.

У 2D іграх Unity дозволяє імпортувати спрайт (2D зображення) і надає продвинутий 2D рендерер віртуального світу. Для 3D-ігор Unity дозволяє задавати параметри стиснення текстур, mip-карт і налаштувань дозволу для кожної платформи, підтримуваної ігровим движком, а також підтримку карт рельєфу, відображення, параллакса, оклюзії навколишнього простору екрану (SSAO), динамічних тіней з використанням карти тіней, рендерингу в текстуру і повноекранних ефектів постобробки.

Станом на 2018 рік движок Unity використовувався для створення приблизно половини нових мобільних ігор на ринку і 60 відсотків контенту доповненої реальності та віртуальної реальності.

Microsoft Visual Studio – це інтегроване середовище розробки (IDE) від Microsoft. Воно використовується для розробки комп'ютерних програм, а також веб-сайтів, веб-додатків, веб-сервісів і мобільних додатків. Visual Studio використовує платформи розробки програмного забезпечення Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store і Microsoft Silverlight.

Visual Studio включає в себе редактор коду, що підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Інтегрований відладчик працює як відладчик рівня вихідного коду і відладчик рівня машинного коду. Інші вбудовані інструменти включають профілювальник коду, конструктор форм для створення додатків з графічним інтерфейсом, веб-дизайнер, дизайнер класів і конструктор схем бази даних. Він приймає плагіни, які розширюють функціональні можливості практично на кожному рівні, включаючи додавання підтримки систем контролю версій (таких як Subversion і Git) і додавання нових наборів інструментів, таких як редактори та візуальні дизайнери, для мов, специфічних для предметної області, або наборів інструментів для інших аспектів розробки програмного забезпечення. життєвий цикл (наприклад, клієнт Team Foundation Server: Team Explorer).

Visual Studio підтримує 36 різних мов програмування і дозволяє редактору коду і відладчику підтримувати (у різному ступені) практично будь-яку мову програмування за умови, що існує спеціальна мовна служба. Вбудовані мови включають C, C++, C++ / CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML і CSS. Підтримка інших мов, таких як Python, Ruby, Node.js і M серед інших, доступна через плагіни. Java (і J#) підтримувалися в минулому.

Базова версія Visual Studio, версія Community, доступна безкоштовно. Слоган для Visual Studio Community Edition - «Безкоштовна

повнофункціональна IDE для студентів, розробників програмного забезпечення з відкритим вихідним кодом і окремих розробників» [38].

C# – об'єктно-орієнтована мова програмування, створена як мова розробки додатків для платформи Microsoft .NET Framework [39].

C# відноситься до сім'ї мов з C-подібним синтаксисом, з них його синтаксис найбільш близький до C++ і Java. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (в тому числі операторів явного і неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, LINQ, виключення, коментарі у форматі XML.

3.3.2. Загальна структура програмної системи

Загальний вигляд структури розроблюваної програмної системи можна зобразити за допомогою діаграми компонентів.

Діаграма компонентів – елемент мови моделювання UML, статична структурна діаграма, яка показує розбиття програмної системи на структурні компоненти та зв'язки (залежності) між компонентами. В якості фізичних компонентів можуть виступати файли, бібліотеки, модулі, виконувані файли, пакети тощо [40].

Діаграма компонентів описує особливості фізичного представлення системи, дозволяє визначити архітектуру розроблюваної системи, встановивши залежності між програмними компонентами, в ролі яких може виступати вихідний, бінарний і виконуваний код. У багатьох середовищах розробки модуль або компонент відповідає файлу. Пунктирні стрілки, що з'єднують модулі, показують відношення взаємозалежності, аналогічні тим, які мають місце при компіляції початкового програмного коду. Основними графічними елементами діаграми компонентів є компоненти та залежності між ними.

Діаграма компонентів описує особливості фізичного зображення системи та забезпечує перехід від логічного зображення до реалізації проекту

в формі програмного коду. Основними графічними складовими діаграми компонентів є компоненти та з'єднувачі.

Компонент є частиною фізичної реалізації системи (наприклад, програмний модуль або інтерфейс користувача), який інкапсулює певний набір функціональних можливостей. З’єднувачі здійснюють взаємодію між компонентами.

У розробці діаграм компонентів беруть участь як системні аналітики та архітектори, так і програмісти. Діаграма компонентів забезпечує узгоджений перехід від логічного представлення до конкретної реалізації проекту у формі програмного коду. Одні компоненти можуть існувати тільки на етапі компіляції програмного коду, інші – на етапі його виконання. Діаграма компонентів відображає загальні залежності між компонентами, розглядаючи останніх як відносини між ними [40].

Діаграма компонентів розробляється для наступних цілей:

- Візуалізації загальної структури вихідного коду програмної системи;
- Специфікації виконуваного варіанту програмної системи;
- Забезпечення багаторазового використання окремих фрагментів
- програмного коду;
- Уявлення концептуальної і фізичної схем баз даних.

Загальна структура системи, що розроблюється, показана за допомогою діаграми компонентів (рис. 3.5).

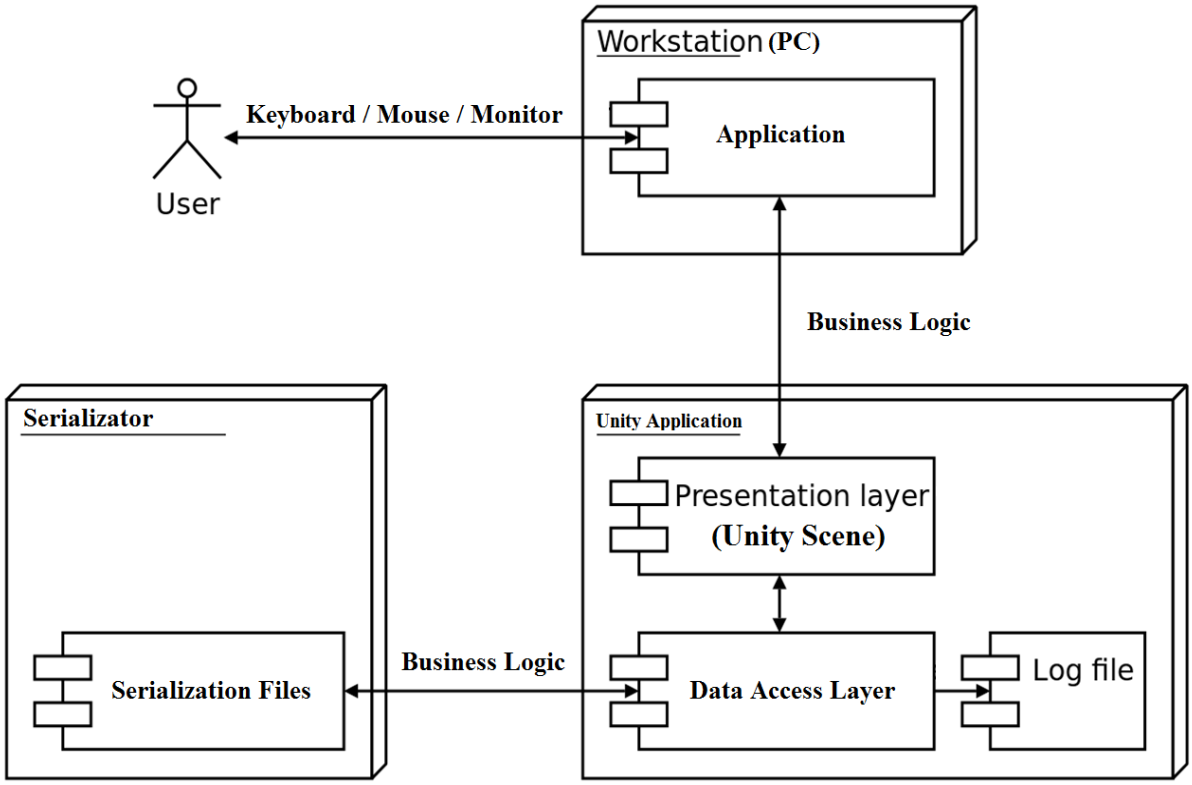

Рис. 3.5. Діаграма компонентів

Зокрема, архітектуру безпосередньо самої програмної системи показано на структурній схемі (рис. 3.6.)

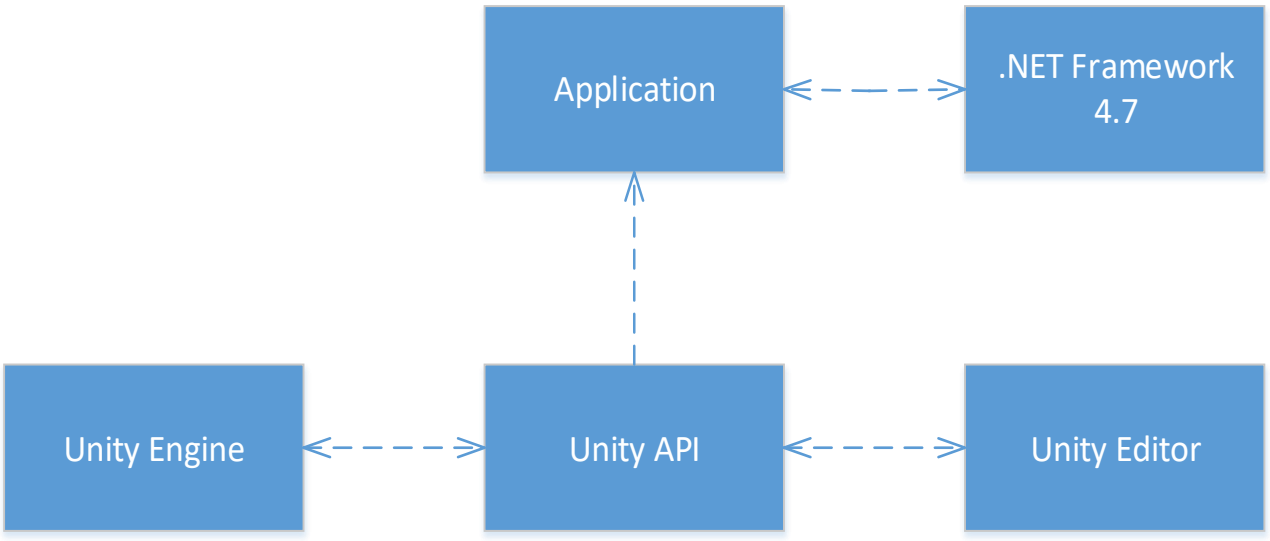

Рис. 3.6. Структурна схема програмної системи

3.3.3. Діаграма класів програмної системи

Діаграма класів (Static structure diagram) - структурна діаграма мови моделювання UML, що демонструє загальну структуру ієрархії класів системи, їх кооперацій, атрибутів (полів), методів, інтерфейсів і взаємозв'язків між ними. Широко застосовується не тільки для документування та візуалізації, але також для конструювання за допомогою прямого або зворотного проектування [41].

Діаграма класів займає центральне місце в проектуванні об'єктно-орієнтованої системи. Нотація класів використовується на різних етапах проектування і будується з різним ступенем деталізації. Мова UML застосовується не тільки для проектування, але і, наприклад, з метою документування.

На діаграмі класів за допомогою спеціальних символів зображуються типи даних програми і відносини між ними, хоча в деяких випадках можуть використовуватися і деякі інші елементи – пакети, екземпляри класів.

Детальний опис видів відносин між класами наведено у таблиці 3.28.

Таблиця 3.28. Опис видів відносин між класами

Вид відносин	**Опис**
Асоціація	Показує, що об'єкти однієї сутності (класу) пов'язані з об'єктами іншої сутності таким чином, що можна переміщатися від об'єктів одного класу до іншого. Є загальним випадком композиції і агрегації.
Агрегація	Різновид асоціації при відношенні між цілим і його частинами. Одне відношення агрегації не може включати більше двох класів (контейнер і вміст). Якщо контейнер буде знищений, то його вміст – ні.

Композиція	Має жорстку залежність часу існування екземплярів класу контейнера й екземплярів класів, що містяться у цьому контейнері. Якщо контейнер буде знищений, то весь його вміст буде також знищено.
Узагальнення (успадкування)	Показує, що один з двох пов'язаних класів (підтип) є частковою формою іншого (надтипу), який називається узагальненням (спадкоємцем) першого.
Реалізація	Відношення між двома елементами моделі, в якому один елемент (клієнт) реалізує поведінку, задану іншим (постачальником).
Залежність	Форма асоціативного відношення, при якому зміна в специфікації одного тягне за собою зміну іншого. Виникає, коли об'єкт виступає, наприклад, у формі параметра або локальної змінної.

Загальний вигляд видів відносин між класами також представлено на рис. 3.7.

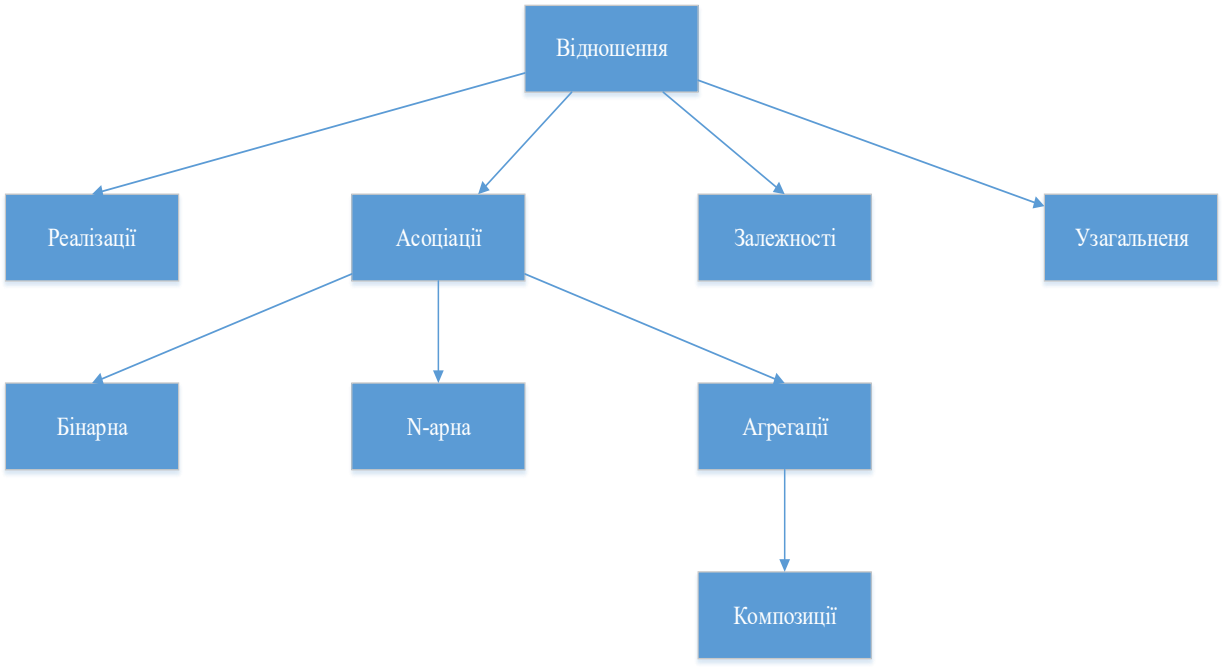

Рис. 3.7. Види відносин між класами

Загальна діаграма класів програмного застосунку представлена на рис. 3.8.

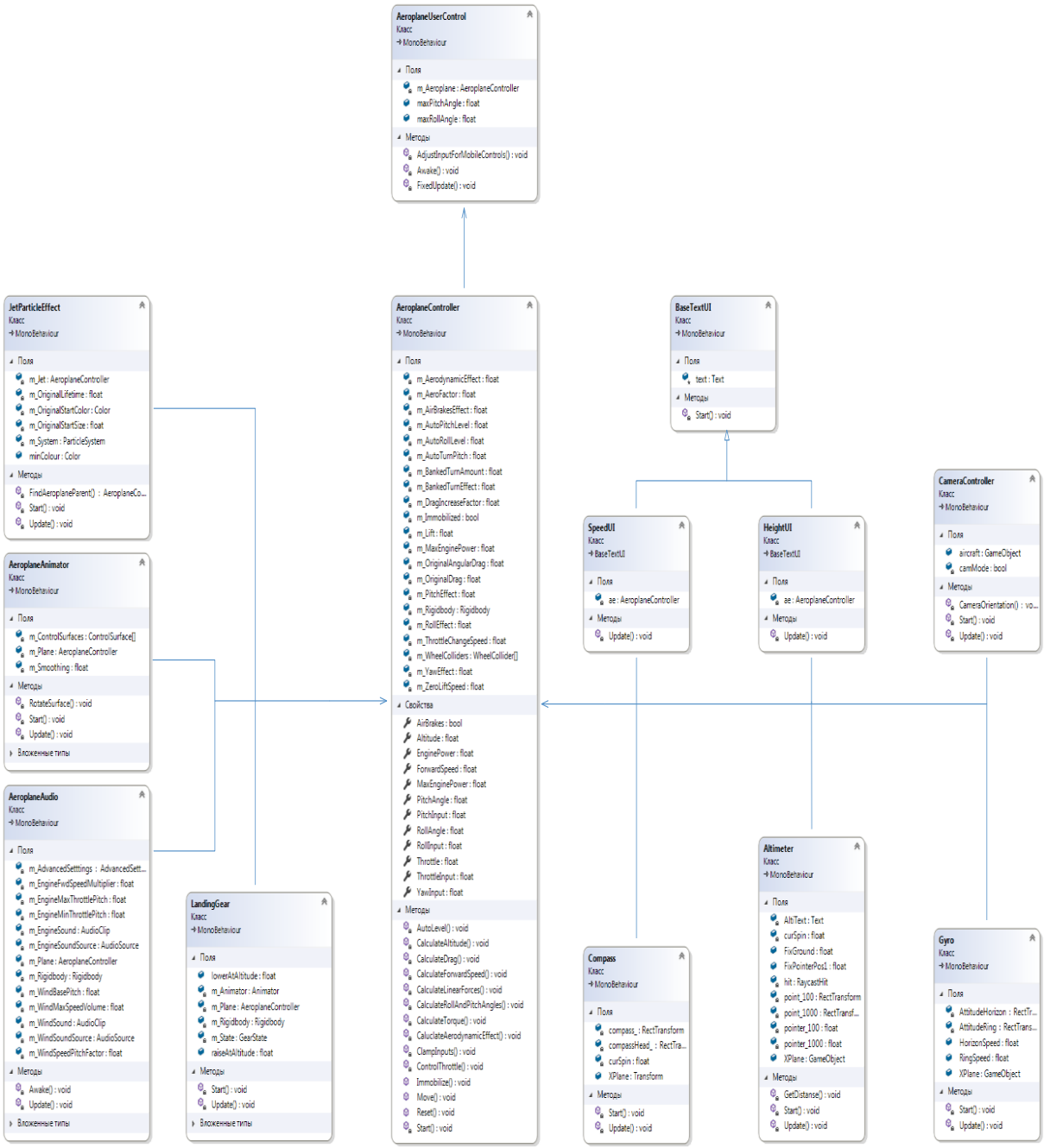

Рис. 3.8. Діаграма класів

Для кращого розуміння функцій кожного класу нижче наведено таблицю 3.29.

Таблиця 3.29. Опис функцій класів

Назва класу	**Функціональна роль**
AeroplaneUserControl	Отримує команди керування літальним апаратом від користувача та передає їх до класу AeroplaneController.
AeroplaneController	Обчислює фізичні сили, що впливають на літальний апарат, керує ним відповідно до команд користувача
AeroplaneAnimator	Керує анімацією (рухом) частин літального апарату відповідно до керуючих команд користувача
AeroplaneAudio	Керує звуком двигунів літального апарату та вітру відповідно до тяги двигунів та швидкості ЛА.
LandingGear	Керує прибиранням та випуском шасі літального апарату в залежності від його висоти над поверхнею.
JetParticleEffect	Керує системою частинок вихлопу струменя, змінюючи його розмір і колір відповідно до поточної величини тяги двигунів.
BaseTextUI	Базовий клас для виведення текстової інформації на графічний інтерфейс користувача.
SpeedUI	Виводить числове значення поточної швидкості ЛА.
HeightUI	Виводить числове значення поточної висоти над поверхнею літального апарату.
Altimeter	Контролює анімацію висотоміру на панелі.
Compass	Контролює анімацію компасу на панелі.
Gyro	Контролює анімацію гіроскопу на панелі.

3.3.4. Прототип інтерфейсу програмної системи

Головне меню зображено на рис. 3.9. Воно є початковим меню при запуску застосунку. При натисненні на кнопку «Реєстрація абітурієнта» відбудеться перехід до меню реєстрації. Натиснувши кнопку «Авторизація», відкриється меню авторизації в системі. Програма буде закрита, якщо натиснути кнопку «Вихід».

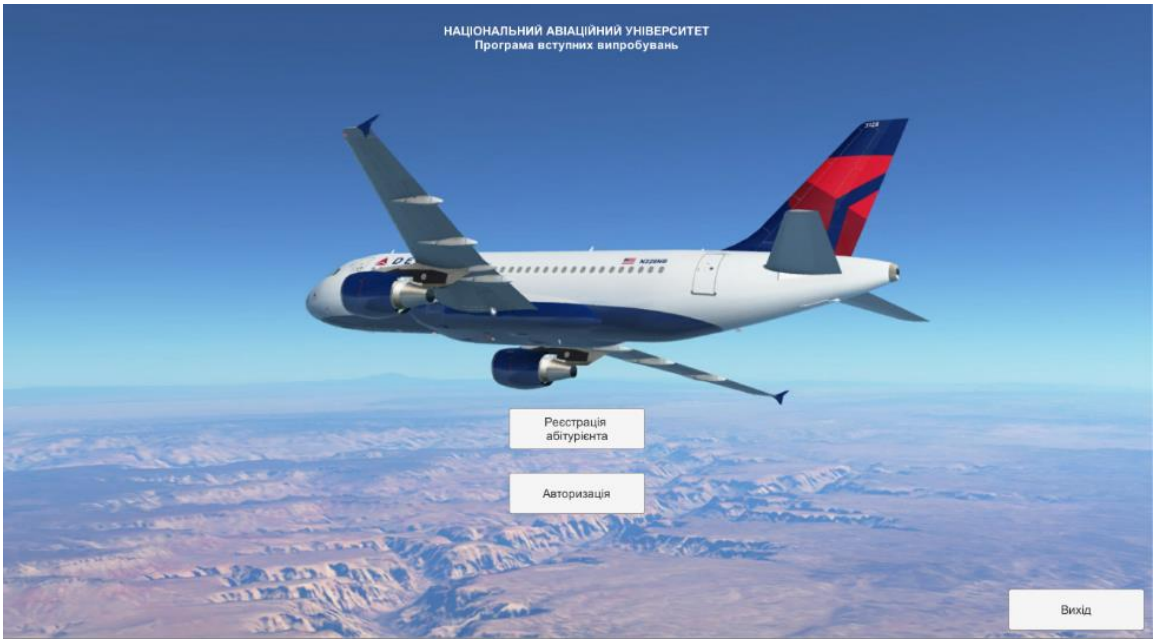

Рис. 3.9. Головне меню програмної системи

Меню реєстрації абітурієнта зображено на рис. 3.10. У ньому відбувається реєстрація нового користувача у системі. Адміністратор не може бути зареєстрований у такий спосіб. Користувач вводить свої персональні дані: прізвище, ім’я, по-батькові, рік народження, серію та номер паспорту, а також особистий пароль. При натисненні кнопки «Зареєструватись» відбудеться перевірка введених даних, після чого система повідомить про успіх або невдачу операції. Після цього (при успіху) відбудеться перехід до передтестувального меню.

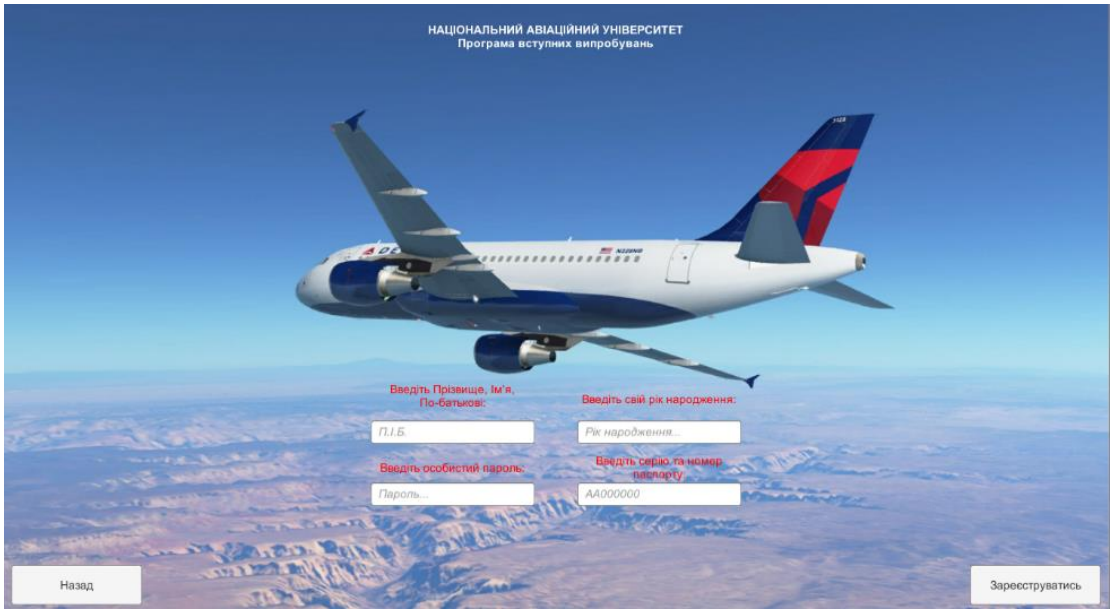

Рис. 3.10. Меню реєстрації абітурієнта

Передтестувальне меню зображено на рис. 3.11. У ньому подається покрокова інформація про послідовність подій впродовж тестування. Натиснувши кнопку «Почати тестування», застосунок завантажить його. При натисканні на кнопку «Назад» відбудеться перехід до головного меню.

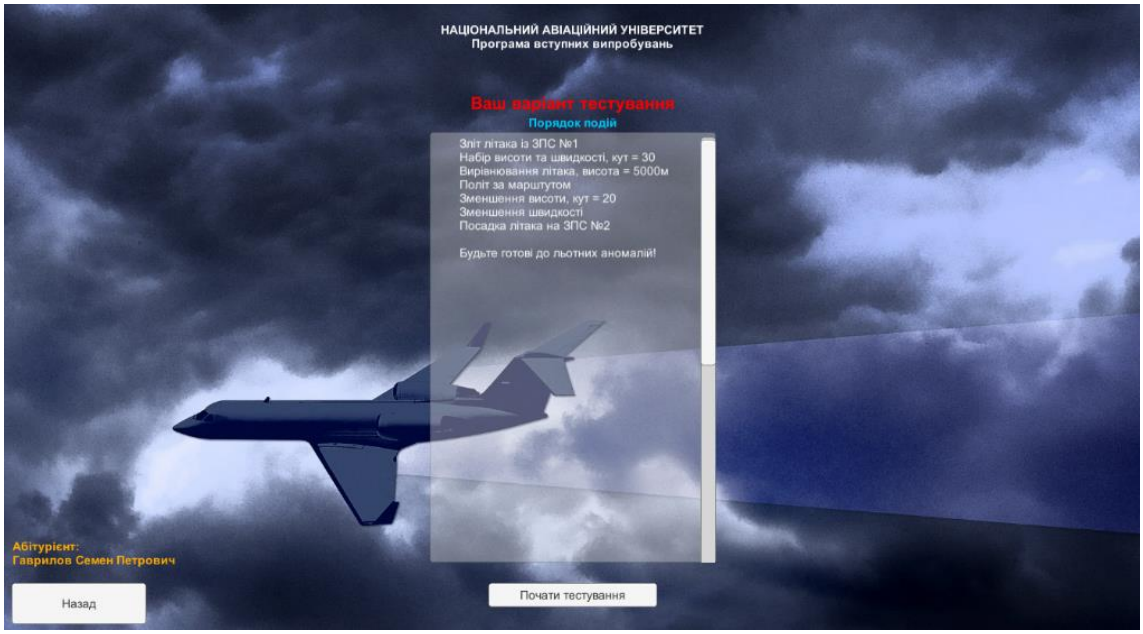

Рис. 3.11. Передтестувальне меню

Меню авторизації зображено на рис. 3.12. У ньому відбувається авторизація існуючого користувача у системі, як абітурієнта, так і адміністратора. Щоб авторизуватися в системі, користувач вводить логін та пароль. При натисненні кнопки «Авторизуватись» відбудеться перевірка введених даних, після чого система повідомить про успіх або невдачу операції. Після цього (у разі успіху) відбудеться перехід до передтестувального меню абітурієнта або меню адміністратора.

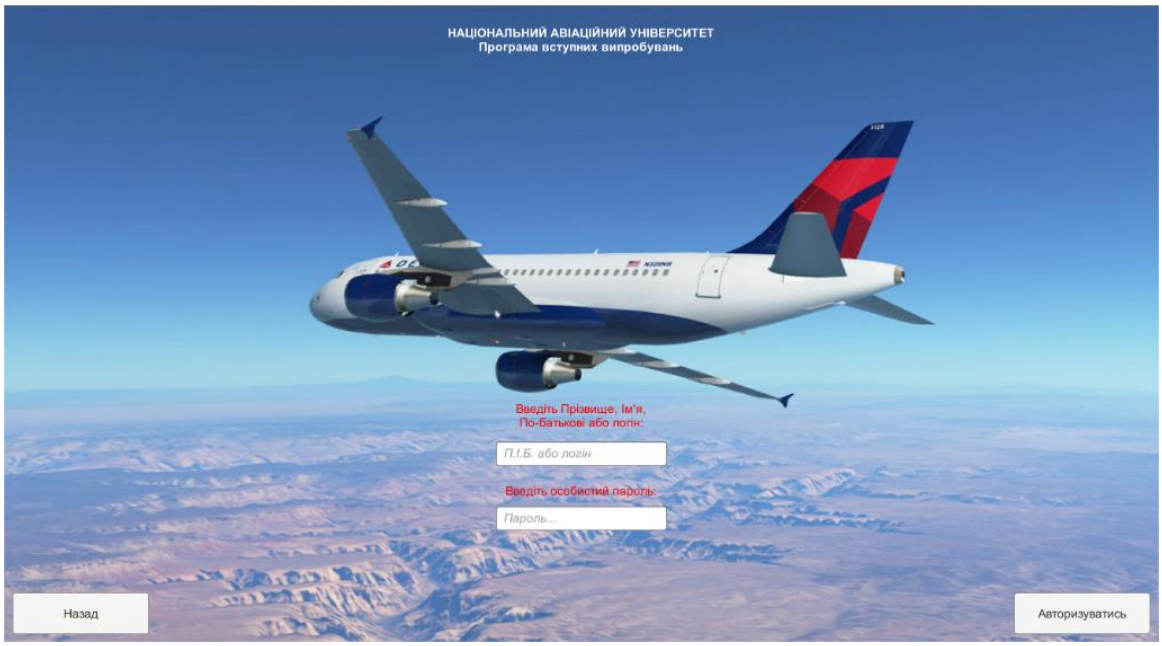

Рис 3.12. Меню авторизації

Меню адміністратора зображено на рис. 3.13. У ньому реалізовано функціонал, яким керує адміністратор.

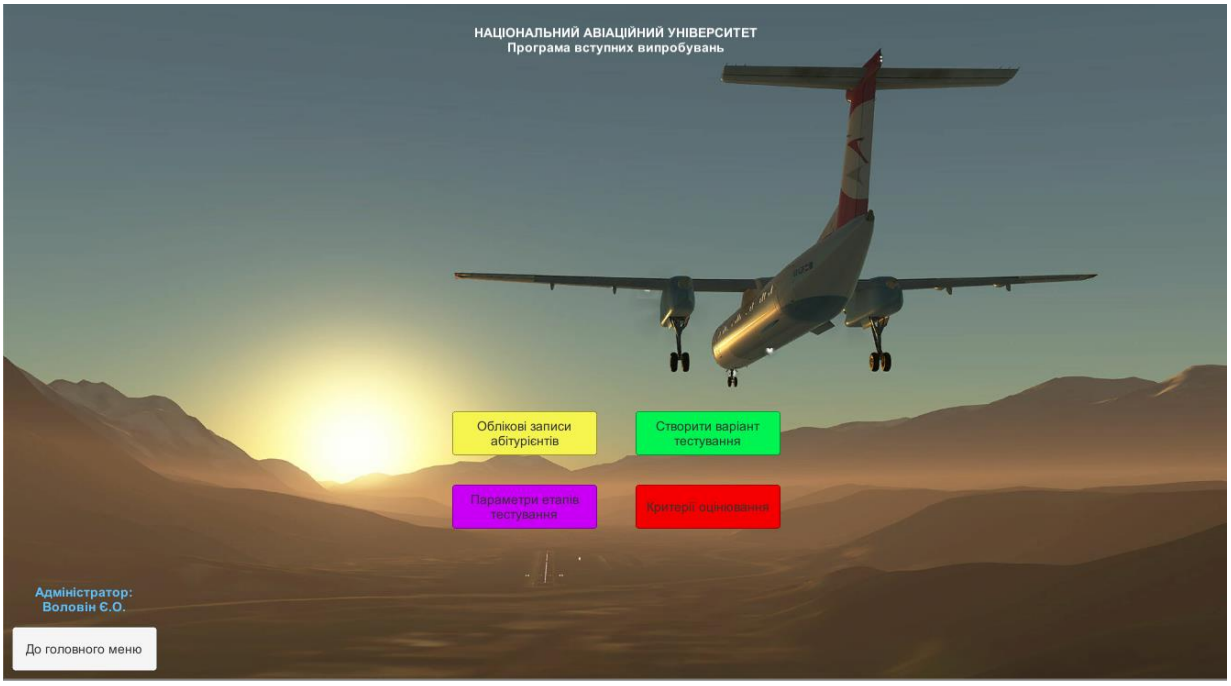

Рис. 3.13. Меню адміністратора

Меню даних користувачів наведено на рис. 3.14. У цьому меню адміністратор може передивлятись персональні дані абітурієнтів.

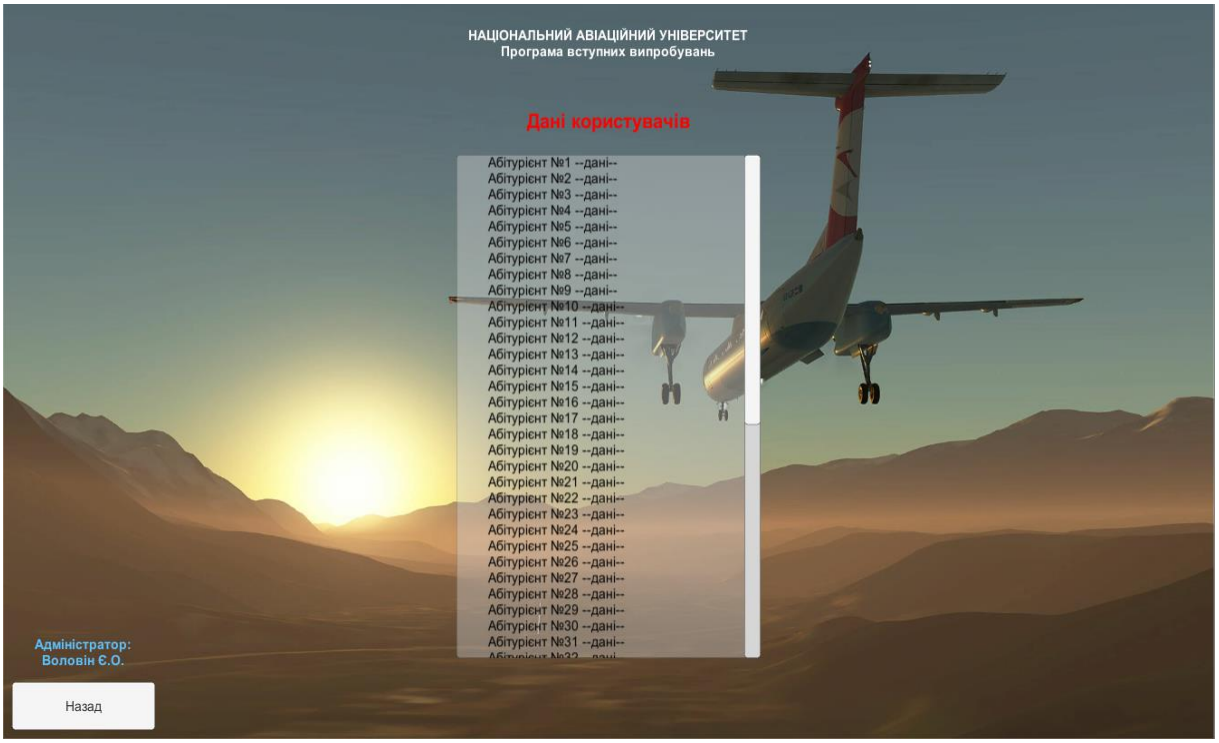

Рис. 3.14. Меню даних користувачів

Меню створення варіанту тестування наведено на рис. 3.15. У цьому меню адміністратор може створювати варіанти тестування з послідовності льотних подій.

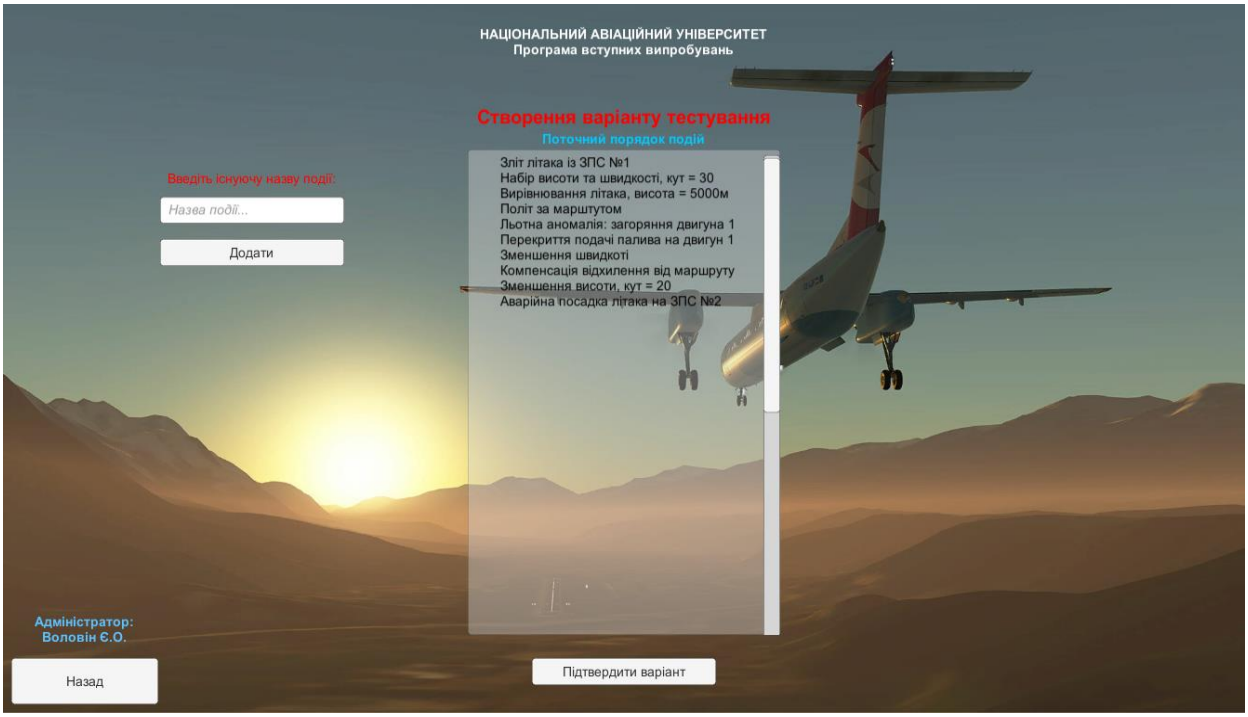

Рис. 3.15. Меню створення варіанту тестування

Інтерфейс під час тестування (польоту) зображено на рис. 3.16.

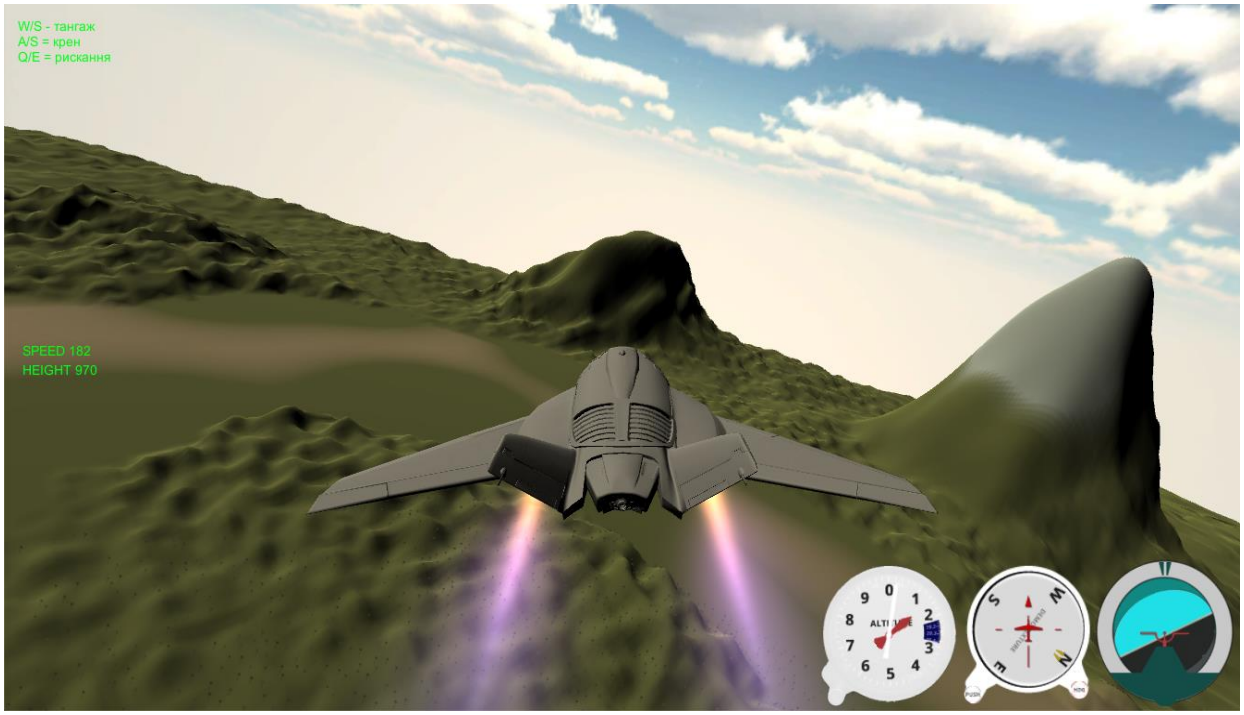

Рис. 3.16. Інтерфейс під час польоту

Висновки до розділу 3

У цьому розділі дипломної роботи були проаналізовані та сформовані функціональні, нефункціональні та системні вимоги до розроблюваного програмного засобу.

Функціональні та нефункціональні вимоги створювалися з урахуванням специфіки програмного засобу та ролі, що він виконуватиме, а також враховуючи особливості майбутніх користувачів.

Саме правильно та в повній мірі сформовані вимоги дозволяють не допускати помилок при розробці програмного забезпечення, насамперед, на етапі проєктування. Тому можна зробити висновок, що розроблюваний програмний засіб працюватиме як очікується та коректно.

Було проведено аналіз ринку та розроблено документи стартап-проєкту, у яких описані усі аспекти управління проєктом розробки програмної системи.

Результатом аналізу стала ринкова (маркетингова) програма, що включає в себе концепції товару, попередній аналіз можливостей ціноутворення, збуту, просування, тощо. Продукт проєкту спирається на цінності та потреби потенційних клієнтів, а саме льотні навчальні заклади, конкурентні переваги ідеї, стан та динаміку ринкового середовища, в межах якого буде впроваджено проєкт, та відповідну обрану альтернативу ринкової поведінки.

Були описані та проаналізовані інструменти розробки програмного застосунку: програмну платформу Unity, інтегроване середовище розробки Microsoft Visual Studio 2019 та мову програмування C#.

Була представлена структура програмної системи за допомогою діаграми класів.

Представлений прототип інтерфейсу користувача не є остаточною версією, оскільки у ході супроводження програмного застосунку будуть виникати нові вимоги до нього, а отже, він буде змінюватися. На даному етапі він демонструє усі функції, реалізовані відповідно до сформованих вимог.

ВИСНОВКИ

Дана дипломна робота висвітлює вкрай актуальну проблему безпеки польотів літальних апаратів, що залежить від пілотів, а саме від їхньої здатності швидко, адекватно та правильно реагувати на несприятливі або небезпечні ситуації під час польоту. Для цього пілоти повинні мати високий рівень професійної підготовки та бути психологічно й психічно стійкими до стресових умов, що, насамперед, визначається їхніми психофізичними якостями.

У ході роботи над даною дипломною роботою були проаналізовані багато психологічних та психофізіологічних аспектів, властивих людині. Були виокремлені ті, що мають найбільший вплив саме на пілотів.

Були проаналізовані та сформовані функціональні, нефункціональні та системні вимоги до розроблюваної системи: її функціонал, продуктивність, зручність використання і т.п.

Після цього було описано структуру програмної системи: проаналізовано архітектуру програмної платформи Unity, що є його основою, та побудовано діаграми варіантів використання, компонентів, класів та ін.

На основі створеної специфікації було розроблено прототип програмної системи для оцінювання психофізичних якостей абітурієнтів льотних навчальних закладів.

Мету дослідження було досягнуто: виконуючи управління проєктом розробки програмної системи було розроблено програмний продукт – інструмент оцінювання психофізичних якостей абітурієнтів для відбору тих людей, хто найбільше підходить на професію пілота, для забезпечення ним льотних навчальних закладів.

Дана програмна система може принести неабияку користь безпеці польотів, оскільки вона теоретично здатна ефективно здійснювати психофізичний відбір абітурієнтів, виокремлюючи тих, хто найбільше підходить на професію пілота.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кушнир О.А. Формирование навыка построения ситуационной осознанности у будущих пилотов путём анализа моделей причинно-следственных связей произошедших авиакатастроф [Електронний ресурс]. Режим доступу: http://molodyvcheny.in.ua/files/journal/2014/6/88.pdf
2. Вплив психофізіологічної підготовки на професійну надійність льотного складу та безпеку польотів / С. В. Бондарчук // Системи управління, навігації та зв'язку. - 2015. - Вип. 3. - С. 11-13. [Електронний ресурс]. Режим доступу: http://nbuv.gov.ua/UJRN/suntz_2015_3_5
3. Державна служба з нагляду за забезпеченням безпеки авіації, Наказ "Про затвердження Правил медичного забезпечення і контролю польотів цивільної авіації України" від 05.12.2005 N 920 [Електронний ресурс]. Режим доступу: http://zakon4.rada.gov.ua
4. Методика лётного обучения / П.В. Картамышев, А.К. Тарасов. – М.: Транспорт, 1974. – 312 с.
5. Doc 9806 AN/763 Основные принципы ИКАО. Основные принципы учета человеческого фактора в руководстве по проведению проверок безопасности полетов. – 2002. – Монреаль; Канада: Межд. организация ГА. [Електронний ресурс] Режим доступу: http://www.aerohelp.ru.
6. Психолого-педагогические основы профессиональной подготовки летного состава / Д.В. Гандер, А.А. Ворона, В.А. Пономаренко. – М.: МАКЧАК, 2010. – 339 с.
7. Развитие системы подготовки персонала авиакомпании в области человеческого фактора / А. В. Захаров // Педагогическое образование. – 2009. – №4. – С. 132–138.
8. Программа по оценке рисков в отношении безопасности полетов / А.В. Линьков, М.В. Кармызов // Научный вестник МГТУ ГА (серия: Эксплуатация воздушного транспорта и ремонт авиационной техники. Безопасность полетов). – 2006. − № 108. – С. 85-90.

9. Авиационная педагогика и психология / Р.Н. Макаров. – М. : МНАПЧАК, 2002. – 490 с.
10. Психологические основы дидактики летного обучения / Р.Н. Макаров Н.А. Нидзий, Ж.К. Шишкин. – М. : МАКЧАК, ГЛАУ, 2000. – 534 с.
11. Н.И. Плотников. Классификации и подходы в исследованиях факторов ошибок пилотов [Електронний ресурс]. Режим доступу: http://shikardos.ru/text/menedjer/
12. Определение и классификация ошибок пилота [Електронний ресурс]. Режим доступу: http://avia.pro/blog/opredelenie-i-klassifikaciya-oshibok-pilota
13. Основи авіації. Вступ до спеціальності / А.О. Комаров. – К.: Вища шк., 1992. – 267 с.
14. Профессиональная подготовка специалистов гражданской авиации : брошюра / Т.С. Плачинда.– Saarbucken, Deutschland : LAPLAMBERT Academic Publishing, 2014. – 104 с.
15. О.О. Бантишева. Особливості та чинники емоційного інтелекту осіб юнацького віку [Електронний ресурс]. Режим доступу:http://es-journal.in.ua/index.php/2227-6246/article/viewFile/158035/157374
16. VII Міжнародна науково-практична конференція «Актуальні проблеми вищої професійної освіти» [Електронний ресурс]. Режим доступу: http://www.kpppo.nau.edu.ua/files/Konfer2019.pdf
17. Л.М. Волкова, А.А. Голубев. Самоконтроль физического состояния будущих специалистов гражданской авиации [Електронний ресурс]. Режим доступу: https://cyberleninka.ru/article/v/samokontrol-fizicheskogo-sostoyaniya-buduschih-spetsialistov-grazhdanskoy-aviatsii
18. Microsoft Flight Simulator. Description and review. [Електронний ресурс]. Режим доступу: https://en.wikipedia.org/wiki/Microsoft_Flight_Simulator
19. X-Plane 11 (simulator). Description and review. [Електронний ресурс]. Режим доступу: https://en.wikipedia.org/wiki/X-Plane_(simulator)
20. Эмоциональная устойчивость в структуре профессиональной надёжности лётного состава и пути её совершенствования средствами теоретической

подготовки / О. А. Кушнир // Вісник Одеського національного університету. Серія : Психологія. - 2015. - Т. 20, Вип. 2(1). - С. 105-114 [Електронний ресурс]. Режим доступу: http://nbuv.gov.ua/UJRN/Vonu_psi_2015_20_2%281%29__14

21. Maintaining Pilot Performance Level and Recovering Confidence - Joint webinar of ICAO, IATA and IFALPA. [Електронний ресурс]. Режим доступу: https://www.icao.int/Meetings/webinar-series/Pages/Maintaining-Pilot-Performance-Level-and-Recovering-Confidence.aspx
22. Stellman, Andrew; Greene, Jennifer (2005). Applied Software Project Management. O'Reilly Media [Електронний ресурс]. Режим доступу: https://web.archive.org/web/20150209011617/http://www.stellman-greene.com/aspm/
23. IEEE. MITcoe.ac.in (MIT College of Engineering SB, Pune) [Електронний ресурс]. Режим доступу: https://web.archive.org/web/20191021081138/http://mitcoe.ac.in/ieee/
24. About IEEE. [Електронний ресурс]. Режим доступу: https://www.ieee.org/about/index.html
25. Institute of Electrical and Electronics Engineers [Електронний ресурс]. Режим доступу:https://en.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers
26. IEEE Standard for Software Quality Assurance Processes, 2014. [Електронний ресурс]. Режим доступу: https://ieeexplore.ieee.org/document/6835311
27. Software Quality Assurance (SQA): Plan, Audit & Review (2018) [Електронний ресурс]. Режим доступу: www.guru99.com/software-quality-assurance-test-audit-review-makes-your-life-easy.html
28. SCM and ISO 9001 by Robert Bamford and William Deibler, SSQC [Електронний ресурс]. Режим доступу: http://www.ssqc.com/do25v6new.pdf
29. Software Test Documentation – How should Test Documentation look like?. THE-SOFTWARE-EXPERTS (2017) [Електронний ресурс]. Режим доступу: http://www.the-software-experts.com/e_dta-sw-test-doku.php

30. Guide to the Software Engineering Body of Knowledge (SWEBOK). IEEE Computer Society. Bourque, P.; Fairley, R.E.. (2014) [Електронний ресурс]. Режим доступу: https://www.computer.org/portal/web/swebok/v3guide
31. Software Reliability. John Wiley & Sons, Inc. c. 567. Pham, H. (1999).
32. IEEE Standard for Information Technology — Systems Design — Software Design Descriptions (2009) [Електронний ресурс]. Режим доступу: https://ieeexplore.ieee.org/document/5167255
33. User preferences of software documentation genres. RH Earle, MA Rosso, KE Alexander (2015) [Електронний ресурс]. Режим доступу: https://dl.acm.org/doi/10.1145/2775441.2775457
34. Топ-7 методів управління проєктами: Agile, Scrum, Kanban, PRINCE2 і інші [Електронний ресурс]. Режим доступу: https://www.pmservices.ru/project-management-news/top-7-metodov-upravleniya-proektami-agile-scrum-kanban-prince2-i-drugie/
35. Функціональні та нефункціональні вимоги [Електронний ресурс]. Режим доступу: http://lvivqaclub.blogspot.com/2008/10/blog-post_17.html
36. Анализ требований разработки ПО. [Електронний ресурс]. Режим доступу: https://unetway.com/tutorial/analiz-trebovanij-razrabotki-po
37. Руководство Unity [Електронний ресурс]. Режим доступу: https://docs.unity3d.com/ru/current/Manual/index.html
38. Visual Studio 2019 [Електронний ресурс]. Режим доступу: https://visualstudio.microsoft.com/ru/vs/
39. Краткий обзор языка C# [Електронний ресурс]. Режим доступу: https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/
40. Буч, Д. Рамбо, А. Джекобсон. Язык UML Руководство пользователя [Електронний ресурс]. Режим доступу: https://habr.com/ru/post/150041/
41. Introduction to UML Static Structure Diagrams. [Електронний ресурс]. Режим доступу: http://santos.cs.ksu.edu/771-Distribution/Reading/uml-section3.21-51.pdf