Вступ

Швидкі темпи інформатизації суспільства призвели до значних змін у повсякденному житті і професійній діяльності людини. Інформатизація світу є ключовою умовою готовності фахівців до здатності працювати у кардинально нових, дедалі більше автоматизованих, умовах праці в галузях, навіть не пов’язаних напряму з комп’ютерними науками.

В будь-якій ситуації взаємодію між користувачем та комп’ютером забезпечують прикладні програми. Процес створення програми, розробка програмного забезпечення на певній мові програмування називається програмуванням. Мова програмування – формальна знакова система, призначена для запису програми, яка зазвичай представляє собою деякий алгоритм в формі, зрозумілій для виконавця-комп’ютера [4].

В програмуванні дуже часто зустрічається термін «структури даних», яким позначається спосіб організації даних в комп’ютерах. Правильний підбір структур даних є надзвичайно важливим для ефективного функціонування алгоритмів їх обробки.

Основною структурою даних в мовах програмування є рядок, що представляє собою такий тип даних, який дозволяє відображати інформацію у текстовому вигляді. Для візуального відображення даних використовуються графічні зображення та відеопотоки. Відеоформати також містять окрім візуальної компоненти акустичну складову – так званий аудіопотік. Окремо звукові дані можуть зберігатися в аудіофайлах.

Метою даного дипломного проекту є розробка програмного мультимедійного комплексу для проведення інтерактивних заходів з підтримкою аудіо- та відеоконтенту, яка може виконувати такі функції:

* інтерактивне відображення інформації різних форматів;
* відображення текстової інформації;
* відображення графічної інформації;
* відображення відеоінформації;
* відтворення аудіоданих;
* налагодження параметрів відображення;
* відлік часу;
* можливість зміни елементів відображення в режимі реального часу;
* редагування текстової інформації;
* керування програмою за допомогою елементів навігації;
* підтримка віддаленого керування програмою;
* розподіл прав доступу до елементів програми;
* підтримка різних режимів відображення інформації: демонстраційного (фронт-енд) та прихованого (бек-енд).

Програма має бути написана на мові С++ і працювати під керуванням операційної системи Windows з використанням віконного інтерфейсу.

Область застосування програми – проведення виховних та розважальних заходів змагального типу в закладах освіти, медіапідтримка телевізійних та інтерактивних ігрових проектів в режимі реального часу.

1 Загальна частина

1.1 Інтерактивні режими взаємодії

Інтерактивність – це принцип організації системи, при якому мета досягається інформаційним обміном елементів цієї системи. Елементами інтерактивності є всі елементи взаємодіючої системи, за допомогою яких відбувається взаємодія з іншою системою або людиною (користувачем). Інтерактивність даного проекту полягає у можливості організації процесу демонстрації, інформування та обліку певних даних в режимі реального часу.

Система реального часу – це система, яка повинна реагувати на події у зовнішньому по відношенню до системи середовищі або впливати на середовище в межах необхідних тимчасових обмежень [9].

Користувачами такої інтерактивної системи можуть бути як оператор програми так і глядачі, які безпосередньо не беруть участь у керуванні інтрефейсною частиною, а лише переглядають демонстраційний варіант, наприклад, за допомогою проектора.

Такий варіант найкраще підходить для проведення ігрових заходів, в яких беруть участь кілька користувачів, розділених в залежності від функцій на кілька ролей: ведучі, оператори програми, учасники гри, журі, глядачі тощо.

В даному проекті до інтерактивних елементів, які мають забезпечувати взаємодію програми і користувачів різних категорій, відносяться наступні елементи:

- інформація про учасників заходу (наприклад, назва команди, прізвище, ім’я, по батькові, гасло, примітка тощо);

- графічне зображення (логотип, емблема, банер тощо);

- кількісна характеристика (наприклад, поточний рахунок);

- часова характеристика (відлік часу);

- демонстраційний матеріал: текст, зображення, аудіо, відео.

1.2 Представлення мультимедійної інформації

Інформація, представлена користувачу за допомогою комп’ютерної техніки, може бути поділена на кілька базових категорій: текстова, графічна, звукова, відео.

Кожна з категорій інформаційних даних може зберігатися у блоках даних, які утворюють файли певних типів, або форматів.

Текстові дані – це представлення інформації в обчислювальній системі у вигляді послідовності друкованих символів. Текстовими даними називаються послідовності з підмножини знаків, що включають лише друковані знаки (літери, цифри, знаки пунктуації) та деякі керуючі знаки (пробіли, табуляції, переведення рядка) [12].

Текстова інформація зазвичай представлена у графічному вигляді за допомогою відповідних елементів вікна.

Текстовий файл – це послідовність символів, зазвичай згрупованих в рядки. Такий файл може містити як форматований, так і неформатований текст. Неформатований текст зазвичай зберігається у файлах з розширенням .txt [12].

Графічне зображення може бути представлене у вигляді дрібних точок, які утворюють характерний візерунок – растр. Таким чином, будь-яке зображення можна закодувати за допомогою координат точок, що мають індивідуальну яскравість. Для кодування кольорових графічних зображень застосовується принцип декомпозиції (розкладання) кольору на основні складові: наприклад, червоний (Red), зелений (Green) і синій (Blue). Цей принцип базується на тому, що будь-який колір можна отримати шляхом змішування трьох зазначених кольорів.

У сучасній машинній графіці використовуються десятки спеціалізованих форматів даних. Деякі з них розроблені окремими фірмами під конкретні програмні засоби, інші створені науково-дослідними установами, у більшій чи меншій мірі пов'язаними співпрацею з Міжнародною організацією стандартів (iso.org). Проте у повсякденній практиці зустрічається всього лише декілька [12].

Найпростіший формат – BMP (BitMaP, тобто бітова карта), який з’явився з першими версіями операційної системи Microsoft Windows. Він зберігає дані про точки зображення в повному форматі, без стиснення.

Для скорочення витрат на графіку було розроблено спеціальні форми стиснення файлів. Найвідоміший з них JPG – базується на першому міжнародному стандарті для збереження зображень із деякою втратою якості JPEG (Joint Photographic Expert Group). Головним чином він призначений для фото, характерною рисою яких є плавні переходи напівтонів і розмиття чітких ліній.

Аудіо – загальний термін, що стосується звукових технологій. Найчастіше під терміном аудіо розуміють звук, записаний на звуковому носії. Розрізняють аналогове та цифрове аудіо (або, аналоговий звук та цифровий звук) [15].

Цифрові звукові формати – формати файлів для збереження звукових даних у комп’ютерних системах. Файли таких форматів називають також аудіофайлами, або звуковими файлами.

Загальний принцип збереження аудіо на цифрових носіях полягає у послідовній фіксації значень амплітуди звукових коливань, які при відтворенні звуку відповідатимуть положенню мембран у динаміках. Ці значення записуються з певною частотою дискретизації та певним амплітудним розділенням. Для зменшення обсягів, ці дані можуть бути стиснені з втратами або без втрат.

Існує три основні групи аудіофайлів:

- нестиснені формати, до яких відноситься стандартний WAV;

- формати із стисненням без втрат;

- формати із стисненням з втратами, до яких належить і популярний формат MP3 [15].

Під терміном «відео» розуміють широкий спектр технологій запису, обробки, передачі, зберігання й відтворення візуального і аудіовізуального матеріалу на моніторах. У побутовому значенні відео означає відеоматеріал, телесигнал або кінофільм, записаний на фізичному носії (відеокасеті, відеодиску тощо).

Відеодані характеризуються такими параметрами як кількість кадрів на секунду, розгортка, роздільна здатність, співвідношення сторін екрану, кількість кольорів і кольорова розрядність, бітова швидкість або ширина відеопотоку.

Відеодані зазвичай зберігаються у файлах-контейнерах. Мультимедійний контейнер – формат файлів, що може містити дані різних типів, стиснених різними кодеками і дозволяє зберігати аудіо, відео i текстову інформацію в єдиному файлі. Найпоширеніші відеоформати-контейнери: MP4, ASF, AVI, Matroska, MOV [21].

Формат AVI (від англійського Audio Video Interleave, що буквально можна перекласти як «чергування аудіо та відео») – мультимедійний контейнер для аудіо-відео даних, що був впроваджений 1992 року компанією Microsoft для пристосування системи Windows для обслуговування мультимедіа [22].

Запис даних уможливлюється процесом кодування, а зчитування – декодування. Технологія, використана у форматі AVI, дає можливість кодування як нестиснених, так і стиснутих даних. Найчастіше використовуються формати стиснення XviD, DivX, Intel Real Time Video, Indeo, Cinepak, MPEG-4 та інші.

Відповідно до універсальних форматів. Які доречно використати в проекті, можна віднести наступні:

- текстовий TXT;

- графічні BMP, JPG;

- аудіо WAV, MP3;

- відео AVI.

2 Конструкторська частина

2.1 Розширений аналіз завдання

Правильне вирішення поставленої задачі потребує проведення аналізу завдання, з’ясування всіх вимог до проекту і на основі цих даних побудову попереднього опису та плану дій, який буде враховуватися на етапах проектування та розробки програми.

Завдання до даного дипломного проекту потребує розробити програмний мультимедійний комплекс для проведення інтерактивних заходів з підтримкою аудіо- та відеоконтенту, відповідно для взаємодії програми с користувачем необхідно розробити зручний віконний інтерфейс з елементами навігації та відображення. Для забезпечення інтерактивності програмного комплексу до завдань проекту належить відображення інформації про учасників заходу, ведення рахунку гри, відлік часу, демонстрація інформації у різних форматах, керування відображенням за допомогою мережевих комунікацій.

Зважаючи на поставлені завдання, можна визначити такі базові задачі:

* реалізація віконного інтерфейсу користувача;
* реалізація функції відображення текстової інформації;
* реалізація режиму завантаження текстової інформації;
* реалізація режиму редагування текстової інформації;
* реалізація режиму збереження текстової інформації;
* реалізація режиму налагодження параметрів відображення;
* реалізація режиму відтворення графічної інформації;
* реалізація режиму відтворення відеоінформації;
* реалізація режиму відтворення аудіоінформації;
* реалізація режиму керування числовими показниками;
* реалізація режиму налагодження числових показників;
* реалізація режиму відліку часу;
* реалізація режиму налагодження часового відліку;
* реалізація режиму віддаленого керування додатком;
* реалізація режиму віддаленого доступу з обмеженими правами;
* реалізація режиму виходу з програми.

Код програми розробляється таким чином, щоб подальша модернізація надавала змогу удосконалити інтерфейс додатку за допомогою нових елементів навігації та відображення, реалізувати підтримку додаткових форматів графічних, аудіо- та відеофайлів, керування процесами відтворення потокових даних, збереження та завантаження конфігураційних параметрів програми.

2.2 Визначення структур даних для зберігання та обробки даних

Згідно завдання до проекту основними елементами опрацювання даних в програмі є файли та рядки. Серед файлових визначені такі типи даних: текстові, графічні, відео та аудіоформати, обробка яких виконується функціями опрацювання та відтворення, розрахованими на роботу з повноцінними блоками даних, що утворюють файли. Відповідно такі структури даних можна розглядати як стандартні складні типи, обробка яких виконується стандартними засобами за певними алгоритмами, розробка яких передбачена задачами проекту.

Додатково програма має оперувати наборами даних, що пересилаються за допомогою комунікаційних засобів, для обміну інформації між різними модулями програми. Класичним варіантом обміну в такій ситуації є використання мережевих пакетів даних, що представляють собою структурований набір різнотипних даних, для компановки і читання якого використовуються відповідні алгоритми обробки.

Враховуючи, що інформація в пакеті може бути різного типу, очевидним є використання для його опису комбінованого типу даних – структури (запису) з полями різного типу і розміру. Приклад структури пакета наведено на рисунку 2.1.

|  |  |  |
| --- | --- | --- |
| Адреса отримувача | | |
| Адреса відправника | | |
| Порядковий номер | | |
| Заголовок | | |
| Параметри | | |
| Прапорці | | |
|  | Дані |  |
| Контрольна сума | | |
| Кінцевий біт | | |

Рисунок 2.1 – Приклад структури пакету

Також важливо враховувати, що для різних задач і напрямків пересилки можуть бути різні набори даних, які в разі їх окремої реалізації потребують наявності відповідних правил опрацювання, які в мережевій термінології називаються «протоколами». Це ускладнює алгоритми обробки, тому може бути застосовано другий варіант – один алгоритм обробки для пакетів узагальненої структури, які є складнішими за одиничні екземпляри, але універсальними для розробленого алгоритму.

Отже, доцільно використати в програмі саме структуру даного типу.

2.3 Розробка структури програмного пакету

Процес інтерактивного керування програмою передбачає наявність трьох модулів:

- серверного (демонстраційного);

- адміністративного (керуючого);

- клієнтського.

Серверний модуль програми призначений для збору та збереження інформації, поєднання інтерактивних елементів програми, візуальної демонстрації даних. Серверний модуль відповідно є зовнішнім представленням програмного пакету, утворюючи з елементами внутрішньої структури, так званий «фронт-енд» - загальнодоступну частину візуального інтерфейсу програми. До складу серверного модулю за вимогами до проекту входять три блоки, які відповідають за різні режими роботи:

- рахунковий блок, що відображає результати в режимі реального часу;

- блок таймеру, що відповідає за відлік часу;

- блок відображення, що відповідає за відтворення мультимедійного контенту.

Кожен з цих блоків може мати доступ до файлів, необхідних для функціонування програми у відповідному режимі.

Можливість віддаленого керування серверним модулем забезпечується окремим мережевим модулем, який виконує обмін даними з адміністративним і клієнтським модулями програми за допомогою відповідних комунікаційних засобів.

Адміністративний модуль програми призначений для віддаленого керування серверним модулем.

Клієнтський модуль програми призначений для передачі даних до серверного модулю від користувачів з обмеженими правами доступу.

Адміністративний та клієнтський модулі приховані від перегляду сторонніми користувачами, відповідно утворюючи «бек-енд», тобто ту частини програмного інтерфейсу, яка доступна лише окремим користувачам з відповідними спеціалізованими правами доступу.

Основною частиною цих двох блоків є мережевий блок для забезпечення комунікацій з серверним модулем, а також інтерфейсна частина для роботи відповідного користувача.

Схема зв’язків серверного, адміністративного та клієнтського модулів в загальній структурі програми представлена на рисунку 2.2.

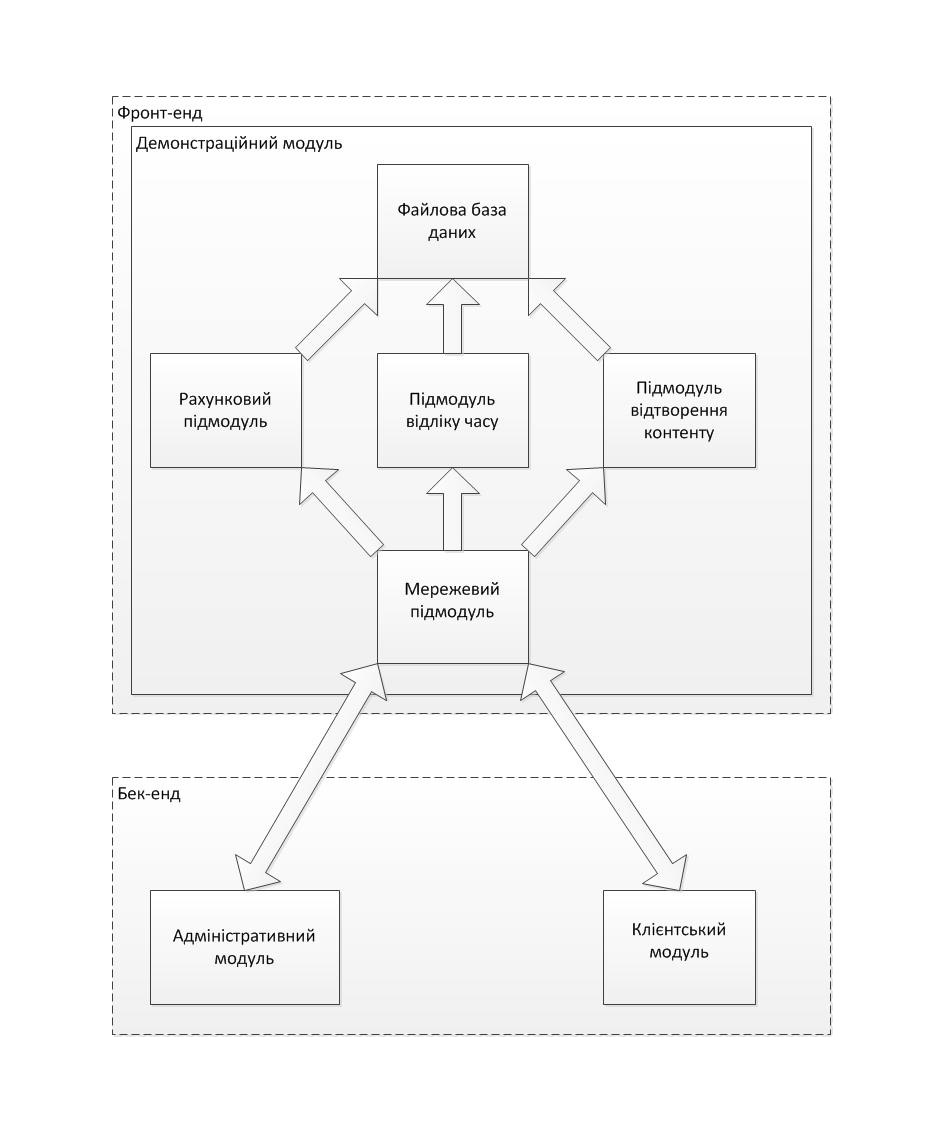


Рисунок 2.2 – Схема зв’язків між модулями програми

2.4 Апаратна підтримка програмного забезпечення

Для забезпечення роботи програми необхідне дотримання певних вимог щодо апаратного та системного програмного забезпечення комп’ютера, на якому воно використовується.

Системні вимоги – це стале поняття, яке використовується для опису характеристик, яким повинен відповідати комп’ютер для коректної роботи певного програмного забезпечення. Ці вимоги можуть описувати, як апаратне забезпечення, так і програмне забезпечення (необхідні драйвери, операційна система тощо). Розрізняють мінімальні та рекомендовані системні вимоги. Якщо мінімальні системні вимоги показують, яка конфігурація системи цілковито необхідна для запуску додатку, то рекомендовані системні вимоги показують, яка конфігурація здатна забезпечити максимально комфортні умови роботи програми [6].

Програма «Медіапомічник» не використовує складних графічних обчислень, тож вимоги до медіа-системи не є високими, а головним фактором працездатності програми є наявність операційної системи сімейства Windows та комп’ютерної мережі. Виходячи з цього мінімальні вимоги для роботи програми наступні:

- процесор частотою 600 МГц;

- оперативна пам’ять 256 Мбайт;

- відеокарта 8 Мбайт;

- жорсткий диск 100 Мбайт вільного простору;

- операційна система Windows XP;

- мережевий адаптер з підтримкою швидкості передачі даних 10Мбіт в секунду.

Рекомендовані параметри для досягнення оптимального режиму роботи з підтримкою бази мультимедійного контенту такі:

- процесор частотою 2,4 ГГц;

- оперативна пам’ять 2048 Мбайт;

- відеокарта 256 Мбайт;

- жорсткий диск 4 Гбайт вільного простору;

- операційна система Windows 7.

- мережевий адаптер з підтримкою швидкості передачі даних 100Мбіт в секунду.

3 Програмна частина

3.1 Вибір моделі розробки, мови та середовища програмування

Розробка програмного забезпечення – це рід діяльності та процес, спрямований на створення та підтримку працездатності, якості та надійності програмного забезпечення, використовуючи технології, методологію та практики з інформатики, керування проектами, математики, інженерії та інших областей знання [7].

Важливим елементом розробки є вибір методології програмування – сукупності методів, застосовуваних на різних стадіях життєвого циклу програмного забезпечення, що мають спільний та, відповідно до цього підходу, дозволяють забезпечити найкращу ефективність.

Виділяють такі поширені методології програмування:

- водоспадна модель;

- макетування (прототипування);

- ітеративна та інкрементна розробка;

- спіральна модель;

- швидка розробка програмного забезпечення (RAD, rapid application development);

- екстремальне програмування (XP, extreme programming);

- різні види методологій гнучкої розробки (FDD, Scrum, OpenUp та інші).

Зручною методологією для невеликих проектів є швидка розробка застосунків, RAD – концепція створення засобів розробки застосунків, програмних продуктів, що приділяє особливу увагу швидкості й зручності програмування, створенню технологічного процесу, що дозволяє програмістові максимально швидко створювати комп’ютерні програми. Концепцію RAD також часто зв’язують із концепцією візуального програмування [7].

Основні принципи RAD наступні:

- інструментарій має бути націлений на мінімізацію часу розробки;

- створення прототипу для уточнення вимог замовника;

- циклічність розробки, тобто кожна нова версія продукту ґрунтується на оцінці результату роботи попередньої версії замовником;

- мінімізація часу розробки версії за рахунок перенесення вже готових модулів і додавання функціональності в нову версію;

- команда розробників повинна тісно співпрацювати, кожен учасник повинен бути готовий виконувати декілька обов’язків;

- управління проектом повинне мінімізувати тривалість циклу розробки.

Порівняння схема роботи даної моделі з традиційною лінійною (каскадною) представлена на рисунку 3.1.



Рисунок 3.1 – Порівняльна схема традиційної моделі розробки і моделі RAD

Технологія RAD передбачає, що розробка програмного забезпечення здійснюється невеликою командою розробників в установлений невеликий термін шляхом використання інкрементного прототипування із застосуванням інструментальних засобів візуального моделювання та розробки. Технологія RAD передбачає активне залучення замовника (куратора) вже на ранніх стадіях – обстеження процесу, вироблення вимог до системи. Останнє із зазначених властивостей передбачає повне виконання вимог замовника як функціональних, так і не функціональних, з урахуванням їх можливих змін в період розробки системи, а також отримання якісної документації, що забезпечує зручність експлуатації і супроводу системи. Це означає, що додаткові витрати на супровід відразу після поставки будуть значно менше. Таким чином, повний час від початку розробки до отримання прийнятного продукту при використанні цього методу значно скорочується [7].

Технологію RAD доцільно застосовувати, коли чітко визначені деякі пріоритетні напрямки розробки проекту:

- необхідно виконання проекту в стислі терміни;

- проект виконується в умовах обмеженості бюджету;

- інтерфейс користувача (GUI) є головним фактором;

- можливе розбиття проекту на функціональні компоненти;

- низька обчислювальна складність програмного забезпечення.

Всі ці напрямки відносяться й до розробки даного проекту, тому швидка розробка застосунків актуальна для нього. Також важливим елементом цієї концепції є застосування візуального програмування, в якому основним інструментом є візуальні середовища розробки.

Візуальне середовище програмування – це інтегроване середовище розробки програмних засобів (IDE, integrated development environment), яке включає редактор вихідного коду, компілятор або інтерпретатор, засоби автоматизації збірки та засоби для спрощення розробки графічного інтерфейсу користувача. Середовища для візуального програмування також надають змогу конструювати програми шляхом оперування графічними об’єктами. Багато сучасних візуальних середовищ програмування використовуються для реалізації принципів об'єктно-орієнтованого підходу у розробці програмного забезпечення [3].

Технологія візуального програмування дозволяє користувачу візуально спостерігати в процесі розробки основні компоненти програми. За допомогою миші можна створювати та переміщувати окремі компоненти, змінювати їх розмір, а також змінювати властивості об`єкту. При цьому програмування не потрібне, система сама формує початковий текст програми та автоматично вносить до нього зміни. Тим самим значно скорочується час на розробку програми, тому що ручний набір коду потрібен лише для реалізації обчислювальних алгоритмів [7].

Узагальнена структура візуального середовища програмування представлена на рисунку 3.2.

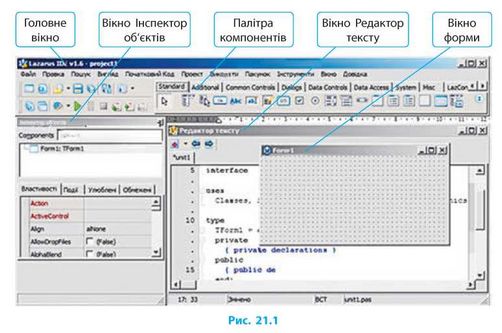


Рисунок 3.2 – Структура візуального середовища програмування

Отже, одним із завдань інтегрованих середовищ розробки є зменшення часу, необхідного на конфігурацію різноманітних інструментів розробки, натомість пропонуючи весь інструментарій в одному пакеті. Це може збільшити продуктивність розробника та оптимізувати процес розробки. Наприклад, синтаксичний аналіз коду може відбуватися безпосередньо під час його редагування, тим самим виявляючи помилки ще до трансляції коду.

Деякі інтегровані середовища розробки призначені для використання певної мови програмування або декількох споріднених мов, надаючи набір можливостей, які більш підходять до парадигми програмування відповідної мови. Такими середовищами є, наприклад, Xcode, С++ Builder та Delphi.

Також існує велика кількість більш універсальних середовищ розробки, які є багатомовними, наприклад, Eclipse, Oracle JDeveloper, NetBeans, Visual Studio, що мають у складі модулі для програмування (генерації коду) різними мовами.

Переважна більшість сучасних середовищ розробки мають графічний інтерфейс користувача.

Одним з популярних середовищ розробки є C++ Builder, що надає програмісту зручні інструменти розробки Windows-додатків з використанням великого об’єму засобів об’єктно-орієнтованого програмування, таких як класи, структури даних, динамічні об’єкти, візуальні компоненти, інструменти роботи з бібліотеками WinAPI.

Всі ці елементи використовуються при розробці даного програмного проекту, тому середовище C++ Builder є необхідним компонентом процесу проектування та реалізації прикладного додатку

Відповідно мовою розробки обирається мова С++, мова програмування високого рівня з підтримкою декількох парадигм програмування: об’єктно-орієнтованої, узагальненої та процедурної. Мова С++ є сумісною з мовою С з деякими важливими нововведеннями:

* підтримка об'єктно-орієнтованого програмування через класи;
* підтримка узагальненого програмування через шаблони;
* доповнення до стандартної бібліотеки;
* додаткові типи даних;
* обробка винятків;
* простори імен;
* вбудовані функції;
* перевантаження операторів;
* перевантаження імен функцій;
* посилання і оператори управління вільно розподіленою пам'яттю.

Універсальність і функціональність мови С++ дозволяє використовувати її для створення програм різної направленості, складності та об’єму.

Отже, в процесі розробки буде використовуватись мова програмування С++ та середовище програмування С++Builder.

3.2 Розробка інтерфейсу програми

3.2.1 Інтерфейс демонстраційної частини

Інтерфейс користувача – це сукупність засобів, за допомогою яких користувач спілкується з різними модулями та функціями програми [5].

Інтерфейс користувача об’єднує усі елементи і компоненти програмного забезпечення, які здатні впливати на взаємодію користувача з програмним забезпеченням. До таких елементів належать: набір задач, які користувач розв’язує за допомогою програмного забезпечення; використовуване програмним забезпеченням середовище; елементи управління програмного забезпечення; навігація між блоками програмного забезпечення; візуальний дизайн вікон та екранних форм програми та інші складові.

Задоволеність користувача програмним продуктом або зручністю його використання в значній мірі визначається інтерфейсом користувача. Взагалі, задоволеність користувача – це функція невеликої кількості факторів:

- можливостей інтерфейсу користувача;

- часу відгуку;

- надійності;

- пристосованості до інсталяції;

- інформаційної підтримки;

- пристосованості до супроводження тощо [5].

# Концептуальна схема взаємодії елементів інтерфейсу представлена на рисунку 3.3.

# E:\share\workCB\2017-18\21mediaHelperDip\+pic\vis001.jpg

Рисунок 3.3 – Схема взаємодії елементів інтерфейсу

# Згідно рисунку 3.3 інтерфейс програми можна розбити на три частини, кожна з яких виконує певну функцію:

- блок представлення є візуальним модулем, за допомогою якого користувач має можливість безпосередньо взаємодіяти з програмою, певними діями задаючи алгоритм операцій і переглядаючи результат їх виконання;

- блок контролера обробляє повідомлення, що ініціюються діями користувача, і керує даними за певним внутрішнім алгоритмолм програми;

- блок моделі зберігає дані, структуровані певним чином і керовані контролером, передаючи результат виконання операцій чи запитів в модуль представлення.

# З урахуванням всіх названих факторів та концепції інтерфейсної оболонки необхідно розробити віконний інтерфейс з набором візуальних елементів керування. Відповідно головне вікно програми представляє собою форму з робочою областю та навігаційними елементами, які дозволяють керувати режимами роботи програми.

# Схема зміни інтерфейсних режимів роботи програми представлена на рисунку 3.4.

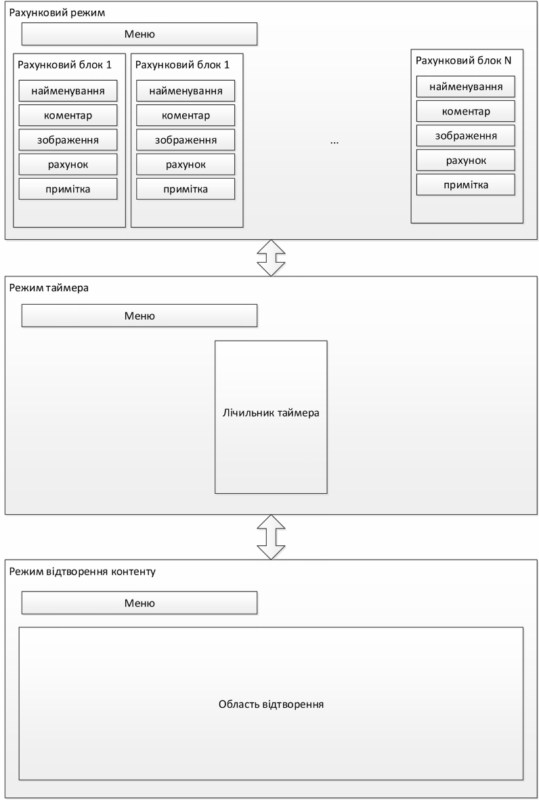


Рисунок 3.4 – Схема зміни інтерфейсних режимів програми

# Головними елементами вікна, яке підтримує всі візуальні режими та мережевий інтерфейс, є наступні:

* набір вкладок (PageControl) – багатосторінковий компонент з функціями групування;
* головне меню (MainMenu) – ієрархічна структура меню з набором команд для головного вікна програми;
* компоненти мережевої комунікації за протоколом UDP (IdUDPServer та IdUDPClient) [16].

# Режим відображення інформації представлено на рисунку 3.5.

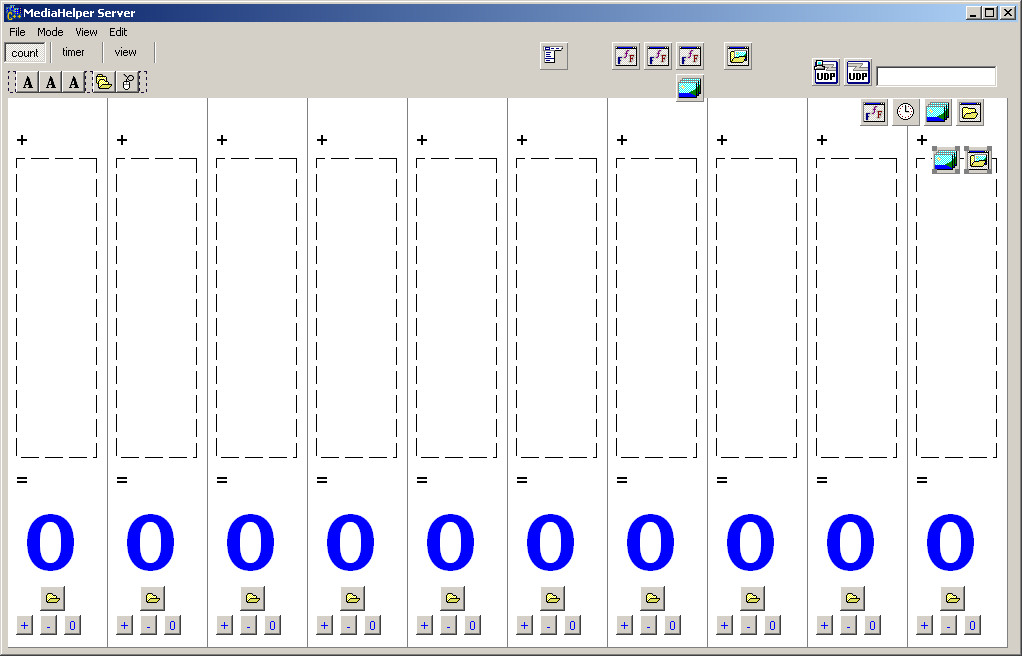


Рисунок 3.5 – Розташування елементів на формі вікна програми в інформаційному режимі

В робочій області розташовуються всі елементи безпосередньої взаємодії користувача з програмою:

* кнопки (Button) – елементи для активації певної дії за допомогою маніпуляцій користувача;
* написи (Label) – елементи для відображення текстової інформації для перегляду;
* зображення (Image) – візуальний компонент для організації роботи з графічними зображеннями.

# Для групування і зручності вибору окремих команд у вікні та системній оболонці доцільно використати спеціальний навігаційний компонент - панель інструментів (ToolBar) – спеціальну панель для розташування елементів для швидкого доступу до основних функцій програми.

Для доступу до об’єктів файлової системи доцільно використати діалогові компоненти, що дозволяють автоматизувати діалоги операцій збереження та завантаження даних:

* вікно завантаження зображення (OpenPictureDialog) – компонент доступу до діалогового вікна для вибору файлу для завантаження даних у графічному форматі;
* вікно завантаження (OpenDialog) – компонент доступу до діалогового вікна для вибору файлу для завантаження даних;
* вікно параметрів шрифту (FontDialog) – компонент доступу до діалогового вікна налагодження параметрів шрифту.

Режим таймеру представлено на рисунку 3.6.

В цьому режимі використовуються аналогічні компоненти Panel, ToolBar, FontDialog, OpenDialog, ImageList для візуалізації інформації, а також наступні:

* таймер (Timer) – компонент для відліку часу;
* програвач медіапотоків (MediaPlayer) – елемент керування відтворенням аудіо- та відеофайлів.

Компонент MediaPlayer – це універсальний програвач аудіо- і відео- інформації. Компонент можна використовувати в двох режимах. По-перше, можна надати користувачеві можливість управляти відтворенням інформації за допомогою кнопкового інтерфейсу, що нагадує панель управління різними програвачами. По-друге, можна зробити сам компонент невидимим і управляти відтворенням інформації за допомогою його методів [13].

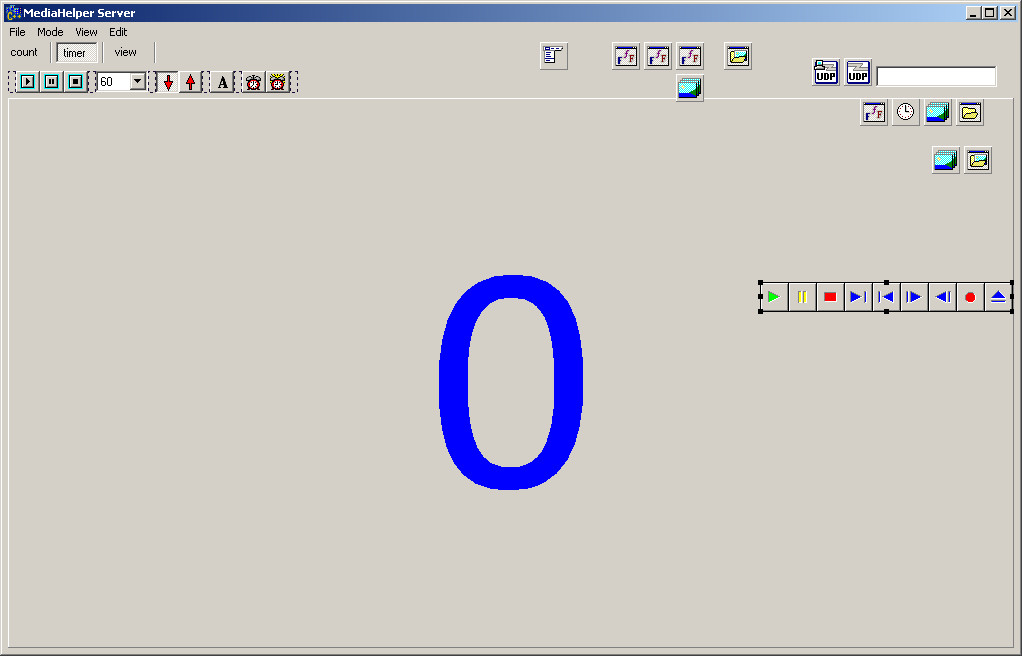


Рисунок 3.6 – Розташування елементів на формі вікна програми в режимі таймеру

В режимі відображення використовуються аналогічні компоненти Panel, Image, ToolBar, OpenDialog, ImageList, MediaPlayer.

Компонент MediaPlayer в демонстраційному режимі вбудований в панель інструментів і працює в візуальному режимі за спрощеною схемою курування.

За допомогою панелі інструментів і набору кнопок на ній відбувається керування вибором відповідного типу контенту:

- відео та аудіо вибирається в діалоговому режимі і підключається до медіапрогравача, що автоматично активується, причому відеопотік проектується в робочій області вікна;

- графічні зображення вибараються за допомогою окремого діалогу і відображаються напряму в робочій області;

- скидання об’єкту відтворення з очищенням робочої області та деактивацією медіапрогравача.

Вікно програми у візуальному режимів відображення контенту представлене на рисунку 3.7.

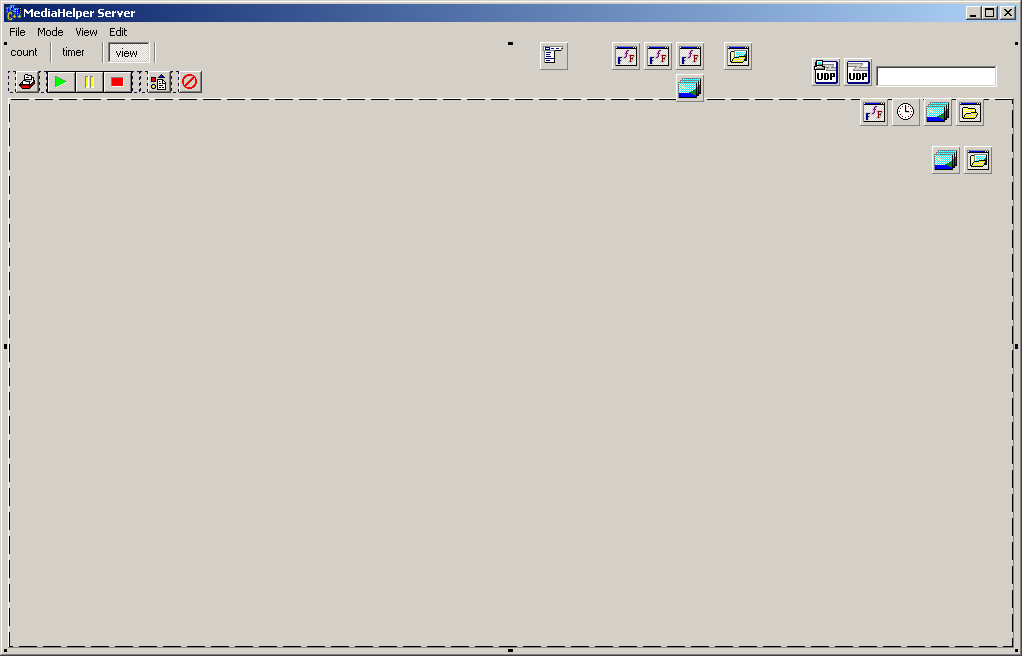


Рисунок 3.7 – Розташування елементів на формі вікна програми в режимі відображення

В додатковому режимі редактора текстових файлів вікно програми представлене у вигляді форми на рисунку 3.8.

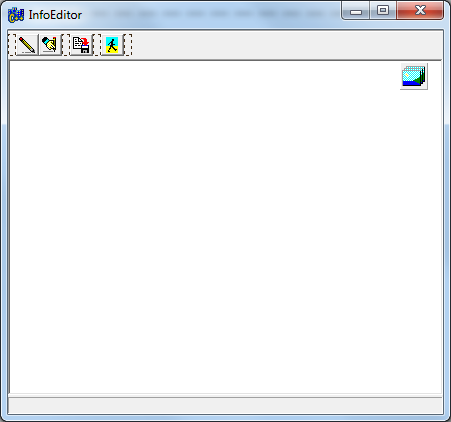
В цьому вікні реалізовані наступні компоненти:

* область введення (Memo) – багаторядкове поле введення текстової інформації для перегляду, введення та редагування;

- панель інструментів (ToolBar) – спеціальна панель для розташування елементів для швидкого доступу до основних функцій програми;

- набір зображень (ImageList) – компонент з набором зображень, що використовуються для прив’язки до ієрархічних компонентів (меню, панелей інструментів, тощо);

- рядок стану (StatusBar) – спеціальна панель для відображення інформації про стан програми.



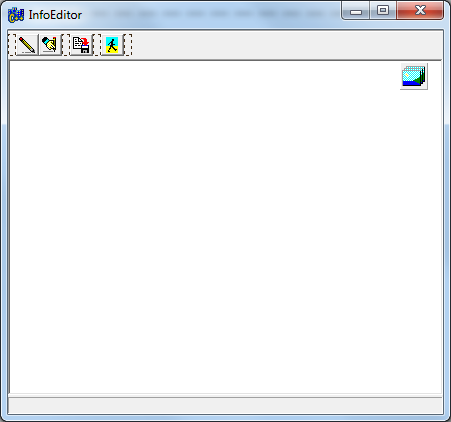


Рисунок 3.8 – Розташування елементів на формі вікна редактора

3.2.2 Інтерфейс модулю адміністратора

Адміністративний модуль виконує сервісну функцію віддаленого керування демонстраційним (серверним) модулем, відповідно його інтерфейсна частина має бути спрощена та уніфікована для швидкого доступу та оперативного реагування користувача-адміністратора.

В даному випадку зручно розташувати панелі керування трьома режимами роботи демонстраційного модулю в одній робочі області за допомогою комбінацій наступних елементів:

* панель (Panel) – контейнер для групування та компоновки візуальних елементів керування;
* головне меню (MainMenu) – ієрархічне меню з набором команд для головного вікна програми;
* компоненти мережевої комунікації за протоколом UDP (IdUDPServer та IdUDPClient).

Для керування та відображення інформації використовуються прості компоненти BitBtn (кнопка з малюнком), Label (текстовий напис).

Зовнішній вигляд вікна адміністративного модулю представлений на рисунку 3.9.

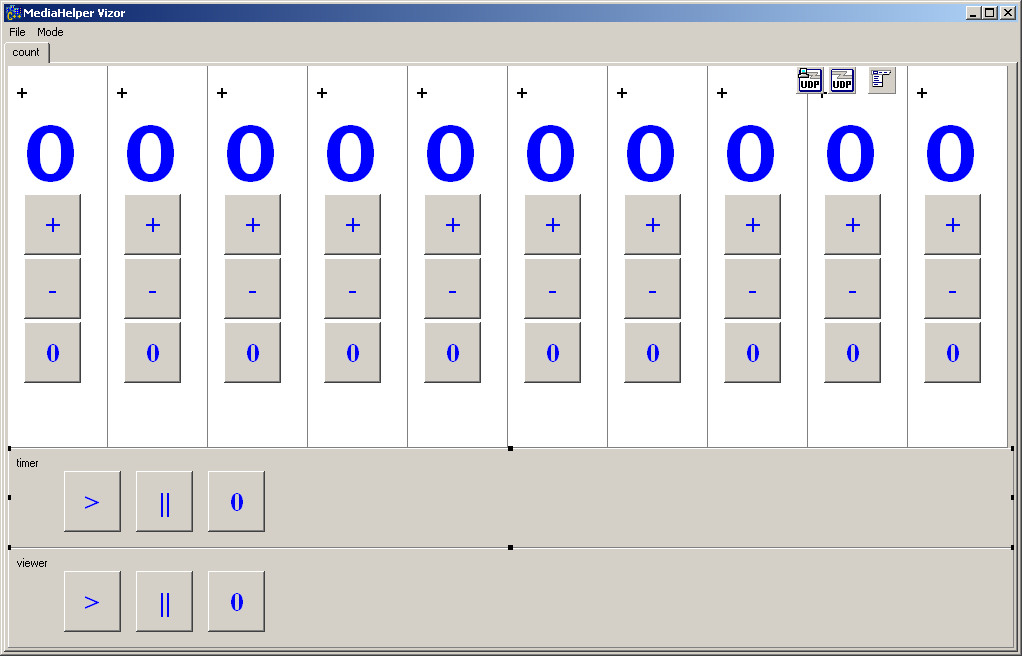


Рисунок 3.9 – Розташування елементів на формі вікна адміністративного модулю

3.2.3 Інтерфейс модулю клієнта

Клієнтський модуль виконує сервісну функцію відправки віддалених інформаційних пакетів демонстраційному (серверному) модулем, відповідно його інтерфейсна частина повинна бути мінімізована до найпростіших елементів керування, що дозволяються виконувати функції введення та відправки. Для здійснення цих операцій використовуються наступні елементи: Panel (панель), MainMenu (головне меню), IdUDPServer (мережевий сервер), IdUDPClient (мережевий клієнт), BitBtn (кнопка з малюнком), Edit (поле введення), Label (текстовий напис).

Зовнішній вигляд вікна клієнтського модулю представлений на рисунку 3.10.

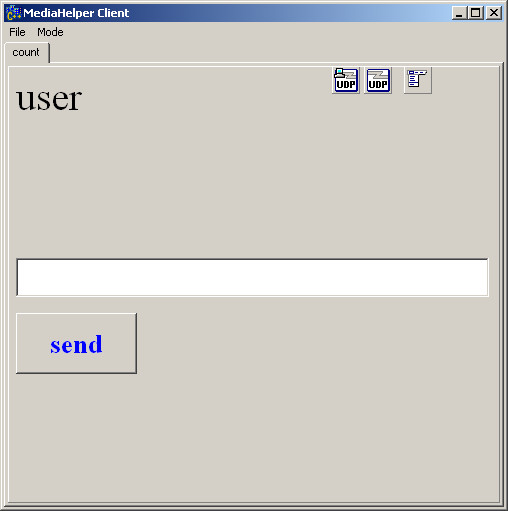


Рисунок 3.10 – Розташування елементів на формі вікна клієнтського модулю

3.3 Розробка алгоритму роботи основних та додаткових функцій

Реалізація програми за визначеними завданням вимогами передбачає наявність основного алгоритму дій, навколо якого будується зовнішній обслуговуючий інтерфейс програмної оболонки. В свою чергу кожна операція у програмі може бути розбита на допоміжні алгоритми, які реалізують ту чи іншу функцію програми. Дискретність алгоритму залежить від типу задачі та алгоритму обробки даних. Відповідно кількість кроків алгоритму залежить від виду операції опрацювання інформації в межах заданого блоку основної програми [1].

Загальний алгоритм програми розбивається на три модулі за кількістю окремих програмних модулів програмного пакету:

- серверний модуль;

- адміністративний модуль;

- клієнтський модуль.

Базовим є серверний модуль, що може працювати як незалежний демонстраційний модуль з підтримкою функцій локального адміністрування. Основний алгоритм цього модулю реалізований за допомогою інтерфейсних функцій стандартного вікна та елементів керування. Робота програми передбачає використання трьох базових режимів з елементами налагодження кожного з них:

- інформаційний режим;

- режим таймера;

- режим відтворення контенту.

Також для інформаційного режиму передбачено використання окремого алгоритму, пов’язаного з редактором текстової інформації, яка використовується в даному режимі.

Віддалений адміністративний модуль реалізує спрощений алгоритм керування режимами роботи серверного модуля за допомогою мережевого зв’язку та механізму відправки спеціалізованих пакетів даних, які формують команди для серверного модуля, що має задіяти відповідний алгоритм зі згаданих вище для відображення інформації в одному з трьох режимів.

Віддалений клієнтський модуль також відправляє комунікаційні пакети даних до серверного модуля, але не керує режимами його роботи, а лише надає певну інформацію для відображення в режимі реального часу.

3.4 Розробка блок-схем алгоритмів програми

Блок-схема алгоритму – це графічне зображення алгоритму у вигляді спеціальних блоків з необхідними зв’язками та поясненнями. Кожний етап алгоритму представляється у вигляді геометричної фігури (блоку), що має певну форму в залежності від характеру операції. Блоки на схемі з’єднуються лініями зв’язку, які визначають послідовність виконання операцій та утворюють логічну структуру алгоритму [8].

Алгоритми поділяють на чотири типи, які зазвичай використовуються в межах однієї програми в різних комбінаціях:

* лінійні;
* розгалуження;
* циклічні;
* допоміжні.

Алгоритмами основних функцій програми передбачено виконання певної послідовності операцій за двома базовими напрямами:

- візуальна складова, що надає інтерфейс користувача та відображає інформацію в режимі реального часу;

- сервісна мережева складова, що надає комунікаційні можливості віддаленого керування основними модулями програми.

Інтерфейсний алгоритм для сервісного модулю є найбільш складним і передбачає вибір режимів роботи з відповідними операціями в межах цих режимів. Мережева складова навпаки є простою і полягає у відправці простих повідомлень і отриманні повідомлень від адміністративного та клієнтського модулів. Блок-схема основного алгоритму візуальної складової серверного модуля програми представлена на рисунку 3.11.

Натомість інтерфейсний алгоритм адміністративного та клієнтського модулів реалізований за спрощеною схемою і представляю собою вибір відповідної опції керування чи повідомлення. Мережева ж складова є більш складною і включає в себе операції ідентифікації (авторизації), формування пакету даних за обраними або введеними даними та відправку сформованого пакету за допомогою мережевого інтерфейсу. Відмінність алгоритмів полягає у внутрішній реалізації операції авторизації та способах формування інформації для компоновки пакету даних.

Блок-схеми алгоритму мережевої складової адміністративного та клієнтського модулів програми представлена відповідно на рисунку 3.12 та 3.13.

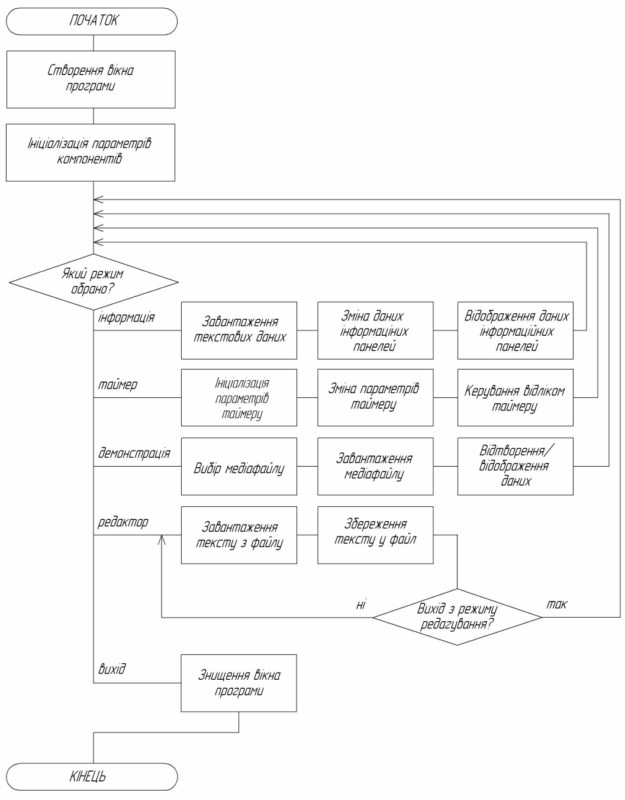


Рисунок 3.11 – Блок-схема основного алгоритму візуальної складової серверного модуля програми

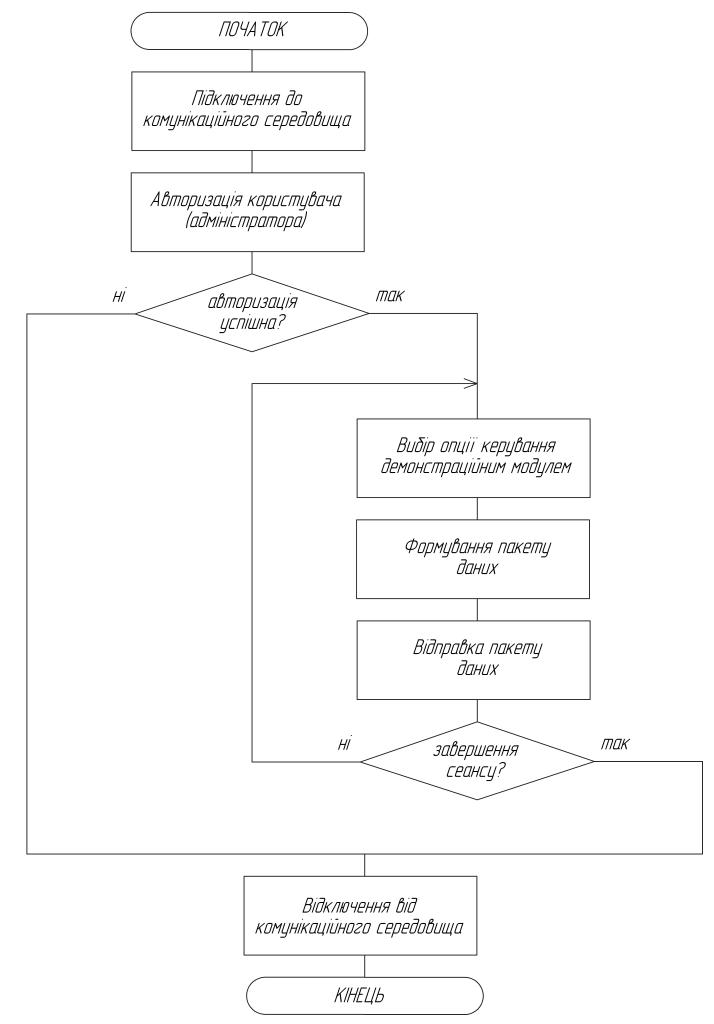


Рисунок 3.12 – Блок-схема алгоритму мережевої складової адміністративного модуля програми

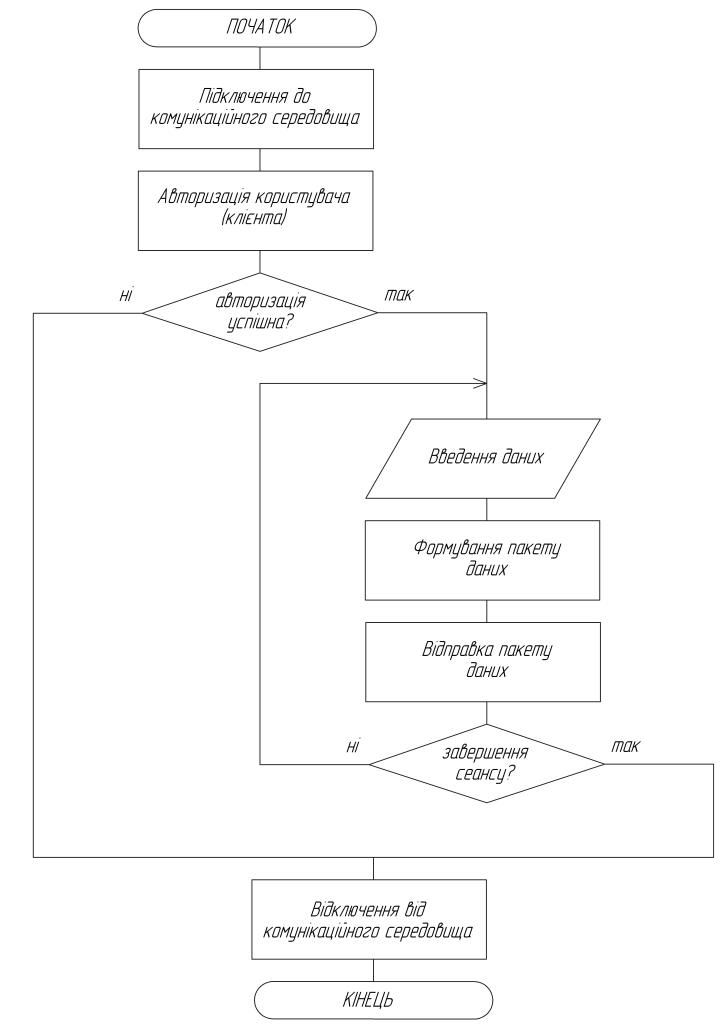


Рисунок 3.13 – Блок-схема алгоритму мережевої складової клієнтського модуля програми

3.5 Розробка програми

3.5.1 Підключення бібліотек, опис типів, змінних та констант

В мові програмування C++ бібліотека – це набір класів і функцій, які написані на базовій мові і є частиною стандарту C++ ISO. Бібліотеки C++ реалізують набір базових контейнерів, функцій для використання і управління цими контейнерами та потоками даних, включаючи взаємодію і файловий ввід/вивід, підтримку деяких особливостей мови програмування і звичайні функції [2].

Для роботи з окремими функціями у процедурах реалізації функціоналу програми потрібно підключити бібліотеки, що надають доступ до вже реалізованих в програмному середовищі інструментів.

Підключення бібліотек у програмах на мові C++ відбувається за допомогою команди #include.

Згідно завдання до проекту в модулях програми підключається модуль візуальних компонентів для роботи з елементами керування у вікнах проекту: #include <vcl.h> - підключення модуля VCL (Visual Components Library).

Для зв’язування виконуючого файлу проекту з зовнішніми бібліотеками використовуються спеціальні директиви:

- #pragma hdrstop вказує компілятору кінець списку загальних заголовних файлів, спільних для всіх модулів;

- #pragma package(smart\_init) визначає послідовність ініціалізації пакетів за посиланнями модулів, що їх використовують;

- #pragma resource "\*.dfm" вказує препроцесору, що для форми треба використовувати файл .dfm з тим же ім’ям, що і ім’я файлу модуля.

Для підключення заголовного файлу з описом об’єктної моделі програми застосовується стандартна команда з вказанням відповідного імені файлу основної форми #include "Unit1.h".

Для підключення заголовного файлу другої форми застосовується аналогічна команда з вказанням відповідного імені файлу #include "Unit2.h".

Змінна – це певний інформаційний блок, який може використовуватись у програмі і характеризується певним типом, що дозволяє конкретизувати поведінку програми з такою змінною [10].

Тип даних визначає множину припустимих значень, формат їхнього збереження, розмір виділеної пам’яті та набір операцій, які можна робити над даними.

Для оголошення змінних у програмі використовуються такі типи даних:

* int – цілі числа;
* AnsiString – динамічний рядок змінного розміру.

В модулі описуються загальні змінні з глобальною областю видимості для використання різними процедурами.

У програмі оголошується такій набір глобальних змінних:

- int num\_players – кількість учасників для відображення в інформаційному режимі;;

- int minT – мінімальне значення таймера для відліку;

- int maxT – максимальне значення таймера для відліку;

- int intT – крок зміни відліку таймера;

- AnsiString soundName1, soundName2 – шлях до файлів звукового супроводу відліку таймера.

Для передачі пакету даних описана відповідна структура з набором полів та змінні даного типу:

typedef struct packMsg {

unsigned short type; // package type

char ip[sSize]; // ip-address

char name[sSize]; // name or login

char word[sSize]; // password or answer

unsigned short mode; // mode type

unsigned short opt; // option type

} packMsg;

packMsg pMsg, pMsgR;

Для передачі пакетів мережевим середовищем виділяється пам’ять для потокових об’єктів, які відповідають за роботу з буфером обміну мережевими повідомленнями:

TMemoryStream \*MS1 = new TMemoryStream;

TMemoryStream \*MS2 = new TMemoryStream;

Окремо також зберігається набір користувачів, підключених до серверної частини в поточний момент, для чого використовується масив рядкових значень – AnsiString ArrAddr[3][aSize]

В окремих процедурах зазвичай використовуються також змінні з локальною областю видимості.

В програмі в процедурах використовуються наступні локальні змінні:

- int i – лічильник циклу з параметром;

- int w – змінна ширина панелі учасника для інформаційного режиму;

- AnsiString vAsCurDir – рядок для формування шляху до поточного каталогу програми.

Також задіяно об’єкт TStringList \*List, який використовується для опрацювання набору рядків, що завантажуються з файлу, опрацьовуються та зберігаються у файл. TStringList – це структура даних, що зберігає в пам’яті набір рядків, має властивості і методи для роботи з рядками типу TString [14].

В додатковому модулі редактора використовуються наступні локальні змінні:

- AnsiString vAsProgDir – рядок для формування шляху до поточного каталогу програми;

- AnsiString vFile - рядок для формування шляху до файлу з текстовою інформацією.

Для збереження статичних даних, необхідних для визначення розмірів певних структур даних, в програмі оголошено також константні величини за допомогою директиви #define:

- #define sSize 256 - розмір текстового поля;

- #define aSize 15 - кількість елементів у масиві користувачів.

3.5.2 Розробка функцій модулів

У мові С++ при програмуванні для Windows початковою точкою входу в програму є функція WinMain(), в яку буде передано управління відразу після запуску програми.

Основна функція проекту WinMain() використовується для ініціалізації вікна та управління формами віконного додатку. Ця функція описана в головному файлі програмного проекту, а для окремих вікон передбачений механізм роботи з процедурами-методами, що визначають реакцію на дії користувача в конкретному вікні.

Функція – це фрагмент програмного коду (підпрограма), до якого можна звернутися з іншого місця програми. В об’єктно-орієнтованому програмуванні функції, оголошення яких є невід’ємною частиною визначення класу, називаються методами [11].

В модулі вікна програми описаний набір наступних функцій-методів:

* функція ініціалізації програми під час створення форми FormCreate();
* функція перемикання між віконним та повноекранним режимами Screen1Click();
* функція підключення до мережевих комунікацій Server1Click();
* функція перерахунку розмірів інформаційних панелей учасників при зміні розміру форми FormResize();
* функція активації режиму завершення програми Exit1Click();
* функція отримання і обробки мережевих повідомлень IdUDPServer1UDPRead();
* функція формування та відправки мережевих повідомлень ActionServerSend();
* набір функцій збільшення рахунку BitBtn1AddPanel1Click(), BitBtn1AddPanel2Click(), BitBtn1AddPanel3Click(), BitBtn1AddPanel4Click(), BitBtn1AddPanel5Click(), BitBtn1AddPanel6Click(), BitBtn1AddPanel7Click(), BitBtn1AddPanel8Click(), BitBtn1AddPanel9Click(), BitBtn1AddPanel10Click();
* набір функцій зменшення рахунку BitBtn1SubPanel1Click(). BitBtn1SubPanel2Click(), BitBtn1SubPanel3Click(), BitBtn1SubPanel4Click(), BitBtn1SubPanel5Click(), BitBtn1SubPanel6Click(), BitBtn1SubPanel7Click(), BitBtn1SubPanel8Click(), BitBtn1SubPanel9Click(), BitBtn1SubPanel10Click();
* набір функцій скидання рахунку BitBtn1ResetPanel1Click(), BitBtn1ResetPanel2Click(), BitBtn1ResetPanel3Click(), BitBtn1ResetPanel4-Click(), BitBtn1ResetPanel5Click(), BitBtn1ResetPanel6Click(), BitBtn1Reset-Panel7Click(), BitBtn1ResetPanel8Click(), BitBtn1ResetPanel9Click(), BitBtn1-ResetPanel10Click();
* функція відстеження натиснення кнопок програвача MediaPlayer1PostClick();
* функція відкриття діалогового вікна завантаження медіафайлу ToolButtonViewMediaOpenClick();
* функція відкриття діалогового вікна завантаження графічного файлу ToolButtonViewImageClick();
* функція очищення робочої області демонстраційного режиму ToolButtonViewClearClick();
* функція перемальовування вікна FormPaint();
* функція виклику діалогового вікна налогодження шрифту основного інформаційного напису ToolButtonNameFontClick();
* функція виклику діалогового вікна налогодження шрифту додаткового інформаційного напису ToolButtonPlayerFontClick();
* функція виклику діалогового вікна налогодження шрифту числового значення рахунку ToolButtonDigitFontClick();
* функція відображення/приховування панелі інструментів інформаційного режиму ToolBarCount1Click();
* функція відображення/приховування панелі інструментів режиму таймера ToolBarTimer1Click();
* функція відображення/приховування панелі інструментів демонстраційного режиму ToolBarView1Click();
* функція перемикання режиму відображення кнопок вибору графічних файлів для інформаційних панелей ToolButtonOpenPicClick();
* набір функцій зміни кількості учасників N12Click(), N22Click(), N31Click(). N41Click(), N51Click(), N61Click(), N71Click(), N81Click(), N91Click(), N101Click();
* набір функцій відкриття діалогового вікна завантаження інформаційного зображення BitBtn1PicPanel1Click(), BitBtn1PicPanel2Click(), BitBtn1PicPanel3Click(), BitBtn1PicPanel4Click(), BitBtn1PicPanel5Click(), BitBtn1PicPanel6Click(), BitBtn1PicPanel7Click(), BitBtn1PicPanel8Click(), BitBtn1PicPanel9Click(), BitBtn1PicPanel10Click();
* функція запуску таймера ToolButtonTimerStartClick();
* функція зупинки таймера ToolButtonTimerStopClick();
* функція скидання таймера ToolButtonTimerResetClick();
* функція зміни значень таймера при виборі максимального значення ComboBox1Change();
* функція роботи таймера TimerTimerTimer();
* функція зміни параметрів шрифту значення таймера ToolButton1Click();
* функція запуску діалогового вікна вибору файлу озвучки щосекундного відліку ToolButtonTimerDingClick();
* функція запуску діалогового вікна вибору файлу озвучки закінчення відліку ToolButtonTimerGongClick();
* функція зміни напрямку відліку часу на зворотній SpeedButtonTimerModeDownClick();
* функція зміни напрямку на прямий SpeedButtonTimerModeUpClick();
* функція запуску вікна редактора InfoEditor1Click();
* функція завантаження інформаційних даних GetInfo1Click();
* функція очистки інформаційних полів ClearInfo1Click().

В модулі вікна редактора описаний набір наступних функцій-методів:

* функція активації режиму завершення роботи редактора ToolButton7Click();
* функція завантаження даних основного інформаційного поля ToolButton2Click;
* функція завантаження даних додаткового інформаційного поля ToolButton3Click();
* функція завантаження даних відповідного інформаційного поля у файл ToolButton5Click().

Також за алгоритмом програми реалізація окремих функцій передбачає застосування стандартних API-функцій та методів об’єктів:

- ExtractFilePath() – API-функція для отримання шляху до каталогу поточного додатку;

- FileExists() – API-функція для перевірки існування файлу;

- DirectoryExists() – API-функція для перевірки існування каталогу;

- LoadFromFile() – метод для завантаження даних з файлу;

- SaveToFile() – метод для збереження даних у файл;

- Execute() – метод для запуску на виконання діалогових вікон відповідних діалогових компонентів;

- Read() – метод потоку повідомлень для зчитування пакету даних з буферу потоку;

- SendBuffer() – метод комунікаційного компонента для відправки пакету даних через потік повідомлень;

- FreeImage() – метод очищення поля графічного зображення;

- Close() – метод закриття форми (завершення програми).

В адміністративному та клієнтському модулі набір функцій значно менший і частково дублює операції серверного модуля як в інтерфейсній частині, так і мережевій.

3.5.3 Розробка схеми взаємодії модулів програми

В структурі програми всі функції-методи та функції користувача прив’язані до певних компонентів, тому керування кожною функцією-методом відбувається з основної віконної функції шляхом обробки відповідних повідомлень.

Серверний модуль основної програми містить набір функцій, які можна розділити на чотири блоки:

- загальний блок інтерфейсу;

- інформаційний блок;

- блок таймеру;

- демонстраційний блок.

Розроблена згідно описаного алгоритму роботи програми схема взаємодії представлена на рисунку 3.14.

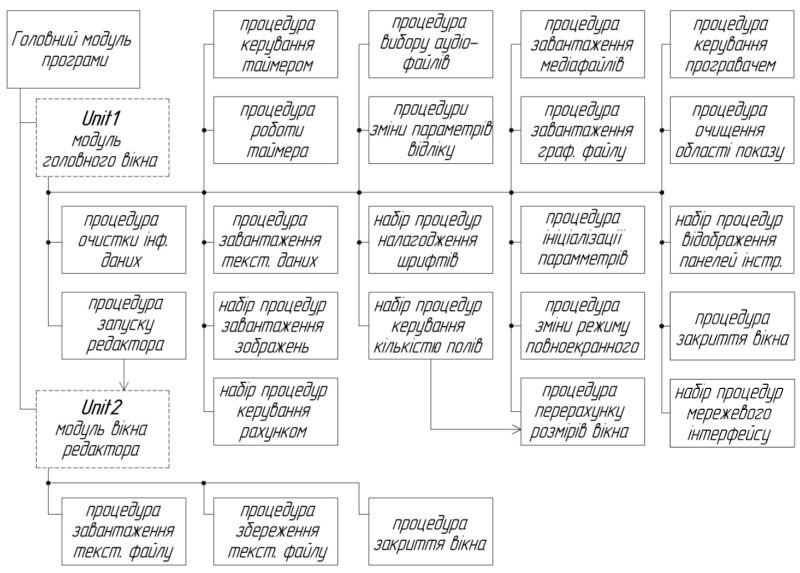


Рисунок 3.14 – Схема взаємодії модулів програми

Адміністративний та клієнтський модуль містять набір простих процедур, які поділяються на дві групи, поєднані між собою:

- процедури авторизації;

- набір процедур вибору опції або формування рядка введення;

- процедура формування та відправки пакету даних через мережеве середовище.

3.6 Створення інсталяційного пакету програмного продукту

Готова програма для спрощення користування нею має включати в себе всі бібліотеки та файли ресурсів. Бібліотеки програмування, які використовуються в процесі розробки, підключаються до виконуваного файлу в процесі компіляції коду та компоновки файлу додатку. Також у виконуваний файл додаються необхідні ресурси: піктограми, курсори, вказівники миші тощо.

Натомість зовнішні бібліотеки та необхідні файли користувача: бази даних, медіадані, рисунки, текстові документи – мають додаватися до програмного проекту на етапі підготовки готового продукту. Зазвичай для цього використовується механізм роботи з інсталяційним пакетом, який містить в згорнутому вигляді всі виконувані, конфігураційні, ресурсні файли, які мають встановлюються чи копіюватися на комп’ютер користувача у вигляді набору папок та файлів.

Єдиний інсталяційний пакет має бути компактним та запускатися на будь-якій робочій станції, не потребуючи встановлення додаткових наборів бібліотек та ресурсних баз [18].

Для створення та налагодження інсталятора можна використати спеціальні програмні утиліти, що дозволяють на базі існуючої виконуваної програми створити інсталяційний пакет, до складу якого можуть входити всі файли з ресурсами, що використовуються у програмному додатку.

Для забезпечення належної функціональності та простоти використання доцільно обрати програмний пакет Inno Setup, основними особливостями якого є:

* сучасний стиль оформлення;
* можливість персонального налагодження інтерфейсу;
* невеликий розмір встановлюваного модуля;
* підтримка різномовних інсталяторів;
* інтуїтивно зрозумілий інтерфейс користувача;
* можливість розбивки інсталятора на пакети [18].

4 Налагодження, експлуатація та обслуговування програми

4.1 Інсталяція та видалення програмного пакету

Для використання програми необхідно її коректно встановити за допомогою спеціального інсталяційного пакету, який включає в себе всі файли програми. Інсталятор підтримує кілька мов інтерфейсу, вибір яких виконується на першому етапі установки (рисунок 4.1).

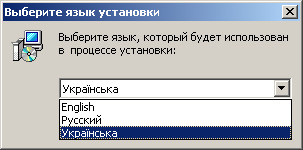


Рисунок 4.1 – Вікно вибору мови інтерфейсу інсталятора

Потім необхідно вказати шлях на диску, куди буде встановлено програмний пакет (рисунок 4.2), ім’я папки для створення ярликів у головному меню робочого столу, у панелі швидкого запуску та на робочому столі, після чого запустити процес установки з заданими параметрами (рисунок 4.3).

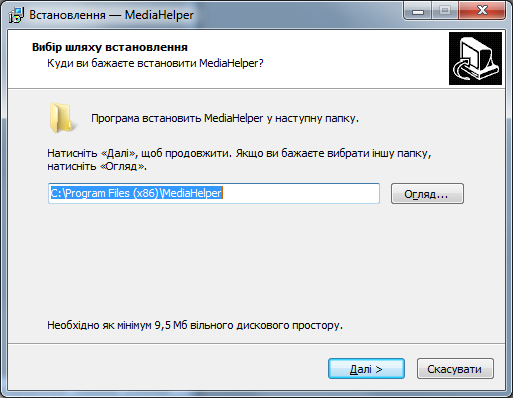


Рисунок 4.2 – Вікно вибору шляху встановлення програми

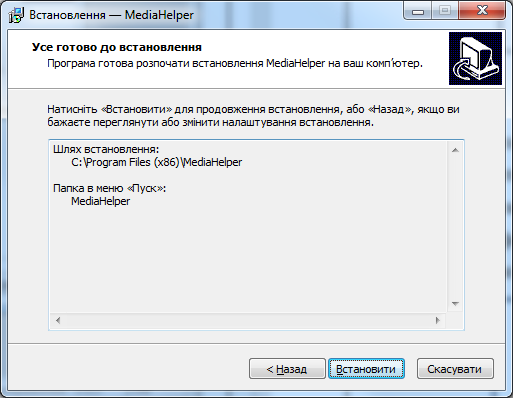


Рисунок 4.3 – Вікно запуску процесу установки

Після установки програми можна відразу запустити її на виконання, якщо встановити відповідний прапорець в діалоговому вікні інсталятора на останньому етапі установки (рисунок 4.4), або вручну за допомогою виконуваного файлу чи створених ярликів у головному меню, на робочому столі чи на панелі швидкого запуску.

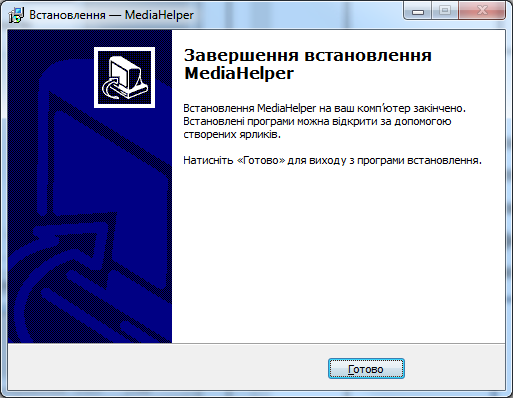


Рисунок 4.4 – Вікно завершення процесу установки

У разі необхідності програму завжди можна видалити за допомогою деінсталятора uninsXXX.exe, який створюється автоматично в процесі установки програми і зберігається у папці з файлами програмного пакету. Деінсталяційний файл можна запустити також за допомогою ярлика у головному меню.

Після видалення всі файли та ярлики, створені під час інсталяції програми будуть видалені з системи.

4.2 Обслуговування та налагодження програми

Для зручної роботи з програмою користувачу надається можливість корегувати зовнішній вигляд вікна та режими роботи програми за допомогою спеціальних інструментів – головного меню, панелі інструментів та іконки в області повідомлень панелі задач. Для цього використовується механізм переведення вікна з активного режиму у прихований з відображенням в області повідомлень.

Будь-яка операція з даними, ініційована користувачем, повертає програму в демонстраційний режим автоматично. Для переведення в активний режим вручну можна скористатися значком у області повідомлень або контекстним меню, яке до нього прив’язане.

Всю довідкову інформацію щодо режимів роботи програми, включаючи варіанти налагодження інтерфейсу, можна переглянути у інформаційному вікні «Допомога», яке викликається за допомогою відповідного пункту меню чи кнопки на панелі інструментів.

4.3 Функціональні обмеження програми і засоби усунення помилок

Програма працює під керуванням операційної системи Windows і створена для 32-розрядної її версії.

Для забезпечення працездатності програми не потрібні встановлені окремо пакети бібліотек, тому що всі необхідні ресурси входять до складу виконуваного файлу програми.

Програма також має вбудовані параметри для візуального оформлення інтерфейсу, прив’язані до україномовного варіанта локалізації.

Програма використовує стандартний механізм роботи операційної системи з розкладками клавіатури.

Для забезпечення повного складу файлів програмного пакету слід користуватися інсталятором, а у разі відмови від використання програми – деінсталятором.

5 Охорона праці та навколишнього середовища

5.1 Аналіз небезпечних і шкідливих факторів виробничих процесів

Виробничою зоною розробника-програміста є його робоче місце в офісі, інструментом праці – комп’ютер та периферійні пристрої, об’єктом розробки – програмне забезпечення, відповідно серед небезпечних та шкідливих факторів виробничого процесу є лише ті, які стосуються його робочого місця та засобів праці.

В загальному значенні небезпечними та шкідливими називають фактори, що призводять до раптового погіршення здоров’я людини, або навіть смерті, під час її трудової діяльності [23].

До числа небезпечних та шкідливих факторів, які впливають на розробника-програміста, слід віднести наступні категорії:

- нервово-психічні перевантаження, викликані розумовим перевантаженням, монотонністю праці, емоційним перевантаженням;

- фізичні перевантаження статичної та динамічної дії;

- невідповідність фізичних параметрів мікроклімату робочої зони санітарним нормам.

Серед фізичних факторів мікроклімату, що можуть заподіяти шкоду або бути небезпечними для людини, можна виділити такі:

- підвищена та знижена температура;

- надмірна запиленість та загазованість повітря;

- підвищена та понижена вологість повітря;

- недостатня або надмірна освітленість робочого місця;

- рівні шуму, що перевищують допустимі норми;

- підвищений рівень іонізуючого повітря;

- підвищений рівень електромагнітних хвиль;

- підвищений рівень статичної електрики;

- небезпека ураження електричним струмом;

- несприятливе забарвлення стін та підлоги, віддзеркалення [20].

5.2 Заходи щодо забезпечення здорових і безпечних умов праці

Зважаючи, що працездатність програміста-розробника залежить в першу чергу від його фізичного стану та параметрів мікроклімату робочої зони, забезпечення здорових і безпечних умов праці полягає переважно в правильній організації робочого місця.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови:

- оптимальне розміщення устаткування, що входить до складу робочого місця;

- достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення;

- необхідно природне і штучне освітлення для виконання поставлених завдань;

- рівень акустичного шуму не повинен перевищувати допустимого;

- достатня вентиляція робочого місця [23].

Ергономічними аспектами проектування відеотермінальних робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різного розташування документів, відстань від очей користувача до екрану, документа, клавіатури тощо), характеристики робочого крісла, вимоги до поверхні робочого столу, можливість регулювання робочого місця і його елементів.

Головними елементами робочого місця програміста є письмовий стіл і крісло. Основним робочим положенням є положення сидячи. Робоча поза сидячи викликає мінімальне стомлення програміста. Раціональне планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Те, що потрібно для виконання робіт частіше, розташоване в зоні легкої досяжності робочого простору.

Важливим є правильна організація загального простору на робочому місці, а також його складових, таких як моторне поле, зона досяжності, оптимальна робоча зона.

Моторне поле – це простір робочого місця, в якому можуть здійснюватися рухові дії людини.

Максимальна зона досяжності рук – це частина моторного поля робочого місця, обмеженого дугами, що описуються максимально витягнутими руками при русі їх в плечовому суглобі.

Оптимальна зона – частина моторного поля робочого місця, обмеженого дугами, описуваними передпліччями при русі в ліктьових суглобах з опорою в точці ліктя і з відносно нерухомим плечем [17].

Оптимальне розміщення предметів праці і документації програміста-розробника в зонах досяжності рук наступне:

- дисплей розміщується в центрі попереду;

- клавіатура – в центрі;

- системний блок – в зоні зліва;

- принтер – в зоні праворуч;

- документація в зоні легкої досяжності долоні – ліворуч – література і документація, необхідна при роботі, а в висувних ящиках столу – література, якою користуються непостійно.

При проектуванні письмового столу варто враховувати наступне:

- висота столу повинна бути вибрана з урахуванням можливості сидіти вільно, в зручній позі;

- нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, ні змушений підбирати ноги;

- поверхня стола повинна мати властивості, що виключають появу відблисків в полі зору програміста;

- конструкція столу повинна передбачати наявність висувних ящиків (не менше трьох для зберігання документації, лістингів, канцелярських приналежностей, особистих речей).

Висота робочої поверхні рекомендується в межах 680-760 мм. Висота робочої поверхні, на яку встановлюється клавіатура, повинна бути 650 мм [17].

Велике значення надається характеристикам робочого крісла. Так, рекомендується висота сидіння над рівнем підлоги повинна бути в межах 420-550 мм. Поверхню сидіння рекомендується робити м’якою, передній край закругленим, а кут нахилу спинки робочого крісла – регульованим [17].

Необхідно передбачати при проектуванні можливість різного розміщення документів: збоку від відеотермінала, між монітором і клавіатурою та інші варіанти.

Положення екрану визначається:

- відстанню зчитування (60 плюс 10 см);

- кутом зчитування, напрямком погляду на 20 градусів нижче горизонталі до центру екрану, причому екран перпендикулярний цьому напряму [19].

Повинна передбачатися можливість регулювання екрану:

- по висоті – 3 см;

- по нахилу – від 10 до 20 градусів щодо вертикалі;

- в лівому і правому напрямках.

Зоровий комфорт підкоряється двом основним вимогам:

- чіткості на екрані, клавіатурі і в документах;

- освітленості і рівномірності яскравості між навколишніми умовами і різними ділянками робочого місця [23].

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з’явитися болі в м’язах, суглобах і сухожиллях. Вимоги до робочої пози наступні: шия не повинна бути нахилена більш ніж на 20 градусів між віссю «голова-шия» і віссю тулуба, плечі повинні бути розслаблені, лікті знаходитися під кутом 80-100 градусів, а передпліччя і кисті рук – в горизонтальному положенні. Причина неправильної пози користувачів обумовлена наступними факторами: немає гарної підставки для документів, клавіатура знаходиться дуже високо, а документи – занадто низько, нікуди покласти руки і кисті, недостатній простір для ніг. З метою подолання зазначених недоліків даються загальні рекомендації: краще пересувна клавіатура, ніж вбудована; повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури, документів і екрана, а також підставка для рук [17].

Узагальнені характеристики робочого місця:

- висота робочої поверхні столу 750 мм;

- висота простору для ніг 650 мм;

- висота сидіння над рівнем підлоги 450 мм;

- поверхня сидіння м'яка з закругленим переднім краєм;

- передбачена можливість розміщення документів праворуч і ліворуч;

- відстань від ока до екрана 700 мм;

- відстань від ока до клавіатури 400 мм;

- відстань від ока до документів 500 мм;

- можливе регулювання екрана по висоті, по нахилу, у лівому і в правому напрямках.

Створення сприятливих умов праці і правильне естетичне оформлення робочих місць на виробництві має велике значення як для полегшення праці, так і для підвищення його привабливості, позитивно впливає на продуктивність праці. Забарвлення приміщень і меблів повинна сприяти створенню сприятливих умов для зорового сприйняття, гарного настрою. У службових приміщеннях, в яких виконується одноманітна розумова робота, що потребує значної нервової напруги і великого зосередження, фарбування повинна бути спокійних тонів.

5.3 Безпека експлуатації устаткування

До роботи з персональним комп’ютером допускаються робітники, які досягли 18-ти років, пройшли медичний огляд, вступний інструктаж, інструктаж на робочому місці та інструктаж з пожежної безпеки.

При роботі над програмних проектом розробник-програміст має дотримуватись правил техніки безпеки, які розповсюджуються на всі категорії користувачів персональних комп’ютерів за трьома напрямками:

- перед початком роботи;

- в процесі виконання робіт;

- по закінченню робіт.

Перед початком робіт розробник повинен:

- оглянути і привести в порядок робоче місце, прибрати з робочого місця всі сторонні предмети, які можуть відволікати увагу і ускладнювати роботу;

- перевірити правильність установки столу, стільця, підставки під ніг, кут нахилу екрану монітора, положення клавіатури і, при необхідності, провести їх переустановку за вимогами до робочого місця;

- перевірити правильність і надійність заземлення обладнання;

- перевірити правильність розташування обладнання, джерел безперервного живлення, їх кабелів та подовжувачів;

- перевірити надійність підключення до системного блоку роз’ємів периферійного обладнання;

- переконатися у відсутності віддзеркалень і відблисків на екрані монітора, відрегулювати положення екрану;

- переконатися у відсутності пилу на екрані монітора, клавіатурі, столі та пристроях, при необхідності протерти їх спеціальною серветкою;

- включити обладнання робочого місця в порядку, встановленому інструкціями по експлуатації на обладнання [19].

В процесі виконання робіт розробник повинен:

- протягом робочого дня тримати в порядку і чистоті робоче місце;

- не закривати вентиляційні отвори комп’ютера та обладнання;

- при необхідності тимчасового припинення роботи коректно завершити всі активні задачі та обмежити доступ до файлів проекту;

- дотримуватися правил експлуатації обладнання;

- при роботі з кожною програмою вибирати найбільш оптимальне поєднання візуальних параметрів (колір і розмір символів, фон екрану, яскравість, контрастність тощо);

- дотримуватися встановлених режимом робочого часу регламентовані перерви в роботі, виконувати рекомендовані фізичні вправи в перервах.

В процесі виконання робіт розробнику забороняється:

- торкатися до задньої панелі системного блоку та інших пристроїв при включеному живленні;

- перемикати роз’єми інтерфейсних кабелів периферійних пристроїв при включеному живленні;

- закривати обладнання паперами і сторонніми предметами, а також допускати скупчення паперів на робочому місці;

- проводити відключення живлення під час виконання активної задачі;

- допускати потрапляння вологи на поверхні пристроїв;

- проводити самостійно розкриття і ремонт обладнання;

- розташовуватися при роботі на відстані менше 50 см від екрану монітору [19].

Тривалість безперервної роботи за персональним комп’ютером без регламентованого перерви не повинна перевищувати двох годин. Під час регламентованих перерв з метою зниження нервово-емоційного напруження, стомлення зорового апарату, усунення впливу гіподинамії і гіпокінезії, запобігання розвитку втоми необхідно виконувати комплекси фізичних вправ [19].

По закінченню роботи розробник повинен:

- провести закриття всіх виконуваних задач;

- вимкнути живлення у порядку, встановленому інструкціями з експлуатації на обладнання;

- прибрати зі столу робочі матеріали і привести в порядок робоче місце.

5.4 Виробнича санітарія

Облаштування робочих місць, обладнаних відеотерміналами, повинно забезпечувати:

- належні умови освітлення приміщення і робочого місця, відсутність відблисків;

- оптимальні параметри мікроклімату (температура, відносна вологість, швидкість руху та рівень іонізації повітря);

- врахування небезпечних і шкідливих факторів, як наявність шуму та вібрації, м’яке рентгенівське випромінювання, електромагнітне випромінювання, ультрафіолетове і інфрачервоне випромінювання, електричне поле між екраном і оператором, наявність пилу, озону, оксидів азоту та аероіонізації.

Відповідно персональні комп'ютери повинні розміщуватися не ближче одного метру від джерела тепла.

Приміщення з ЕОМ повинні мати природне і штучне освітлення відповідно до СНиП ІІ-4-79 «Естественное и искусственное освещение». Природне світло повинно проникати через бічні світло прорізи, зорієнтовані, як правило, на північ чи північний схід, і забезпечувати коефіцієнт природної освітленості (КПО) не нижче 1,5% [23].

При виробничій потребі дозволяється експлуатувати ЕОМ у приміщеннях без природного освітлення за узгодженням з органами державного нагляду за охороною праці та органами і установами санітарно-епідеміологічної служби.

Штучне освітлення приміщення з робочими місцями, обладнаними відео терміналами ЕОМ загального та персонального користування, має бути обладнане системою загального рівномірного освітлення.

Загальне освітлення має бути виконане у вигляді суцільних або переривчатих ліній світильників, що розміщуються збоку від робочих місць (переважно зліва) паралельно лінії зору працівників. Допускається застосовувати світильники таких класів світлорозподілу:

- світильники прямого світла;

- переважно прямого світла;

- переважно відбитого світла.

При розробці оптимальних умов праці програміста необхідно враховувати також рівень шуму: розташовувати робоче місце в шумоізольованих зонах, поза межами серверних та загальних приміщень.

Рівні шуму на робочих місцях осіб, що працюють з відеотерміналами та ЕОМ, визначені ДСанПіН 3.3.2-007-98 [20].

Для забезпечення нормованих рівнів шуму у виробничих приміщеннях та на робочих місцях застосовуються шумопоглинальні засоби, вибір яких обґрунтовується спеціальними інженерно-акустичними розрахунками.

Як засоби шумопоглинання повинні застосовуватися негорючі або важкогорючі спеціальні перфоровані плити, панелі, мінеральна вата з максимальним коефіцієнтом звукопоглинання в межах частот 31,5-8000 Гц, або інші матеріали аналогічного призначення, дозволені для оздоблення приміщень органами державного санітарно-епідеміологічного нагляду. Крім того, необхідно застосовувати підвісні стелі з аналогічними властивостями.

Рівні вібрації під час виконання робіт з ЕОМ у виробничих приміщеннях не повинні перевищувати допустимих значень, визначених в СН 3044-84 «Санитарные нормы вибрации рабочих мест» [20].

5.5 Електробезпека та заходи з пожежної безпеки

Одним з факторів дотримання правил охорони праці є уважне ставлення до протипожежної та електричної безпеки.

Електробезпека – це система організаційних та технічних заходів і засобів, що забезпечують захист людей від шкідливого та небезпечного впливу електричного струму, електричної дуги, електромагнітного поля і статичної електрики [20].

Пожежі можуть виникати від несправності електропроводки або неправильної експлуатації електромережі, яка використовується для живлення персональних комп’ютерів, а також порушення правил обслуговування периферійного обладнання.

Однією з причин пожеж, що виникають від електромереж, є короткі замикання. Якщо на місці короткого замикання будуть горючі матеріали і конструкції, вони миттєво спалахнуть. Тому необхідно стежити за справністю ізоляції проводів, не допускати кріплення їх цвяхами, які можуть порушити ізоляцію [23].

Основні заходи запобігання і правила пожежної безпеки при експлуатації електричних приладів, до яких відноситься і комп’ютерна техніка:

- забороняється вмикати в одну розетку одночасно кілька приладів з великою сумарною потужністю;

- необхідно слідкувати за щільністю контактів у місцях приєднання кабелю до приладів, розетки тощо;

- при порушенні електропроводки або оголенні електричного проводу, поломці електричної вилки або розетки не можна торкатися до оголених місць і взагалі використовувати несправні кабелі;

- не можна використовувати електричні прилади не за призначенням або користуватися ними, не ознайомившись попередньо з інструкцією;

- якщо в електротехніку, що працює, потрапив якийсь сторонній предмет, необхідно в першу чергу вимкнути прилад із мережі, а потім діставати цей предмет;

- забороняється тягнути за електричний шнур руками [23].

Перед закінченням роботи та закриттям приміщень особа, відповідальна за протипожежний стан приміщення, (працівник) зобов’язана перевірити протипожежний стан приміщень, відключити напругу з усіх електроустановок та електроприладів (вимірювальних, електронно-обчислювальних, паяльників, кондиціонерів, вентиляторів, радіоприймачів, комп’ютерів тощо), а також з мереж їх живлення, закрити вікна, кватирки. Виявлені порушення правил пожежної безпеки необхідно усунути до закриття приміщень [17].

В якості профілактичних заходів для забезпечення пожежної безпеки слід використовувати скриту електромережу, надійні розетки з пожежобезпечних матеріалів, силові мережі живлення устаткування виконувати кабелями, розрахованими на підключення в 3-5 разів більшого навантаження, включати й виключати живлення обладнання за допомогою штатних вимикачів.

Треба регулярно робити очистку внутрішніх частин комп’ютерів, іншого устаткування від пилу, розташовувати комп’ютери на окремих столах з матеріалу, що не піддається горінню. Для запобігання іскріння необхідно рідше встромляти і виймати штепсельні вилки з розеток.

Всі живильні кабелі та проводи повинні розташовуватися позаду комп’ютерів, периферійних пристроїв, мережевого устаткування. Їх розміщення в безпосередній робочій зоні користувача забороняється.

Забороняється проводити будь-які операції, пов’язані з підключенням, відключенням або переміщенням компонентів комп’ютерної системи без попереднього відключення живлення [19].

Комп’ютер не слід встановлювати поблизу електронагрівальних приладів і систем опалення.

Не бажано розташовувати на системному блоці, моніторах, периферійних пристроях, мережевому обладнанні сторонні предмети: книги, аркуші паперу, серветки, чохли для захисту від пилу. Це призводить до постійного або тимчасового блокування вентиляційних отворів.

Не слід допускати потрапляння сторонніх предметів в експлуатаційні або вентиляційні отвори компонентів комп’ютерної системи.

Обслуговування та ремонт комп’ютерів може проводити лише спеціаліст з допусками по електробезпеці та з відповідним досвідом.

5.6 Охорона навколишнього середовища

Охорона навколишнього середовища – це регулювання відносин у галузі охорони, використання і відтворення природних ресурсів, забезпечення екологічної безпеки, запобігання і ліквідації негативного впливу господарської та іншої діяльності на навколишнє природне середовище, збереження природних ресурсів, генетичного фонду живої природи, ландшафтів та інших природних комплексів, унікальних територій та природних об’єктів, пов’язаних з історико-культурною спадщиною [23].

Забрудненням навколишнього середовища є дії, які привнесли в екологічну систему не властиві їй живі або неживі компоненти, фізичні або структурні зміни, в результаті яких порушуються процеси круговороту і обміну речовин, а також відтоки енергії, унаслідок чого знижується продуктивність або руйнується дана екосистема [23].

Інформаційні технології сьогодні є екологічнішими за більшість інших видів активної людської діяльності, проте їх ще не можна назвати справді екологічними. Наприклад, ефективність інформаційних мереж напряму залежить від кількості користувачів, тобто, від кількості комп’ютерів, включених до мережі. Але для виготовлення одного звичайного персонального комп’ютера потрібно від 15 до 19 тонн матеріалів, що можна порівняти з 25 тоннами, потрібними для виготовлення автомобіля. На кожен функціонуючий комп’ютер (використовуваний в середньому протягом 4 років) припадає 1,5 комп’ютери вироблених. А близько третини комп’ютерів ніколи не продаються взагалі через швидкість, з якою вони втрачають технологічну актуальність.

У сучасному світі виробники створюють мережі по збору списаної техніки, будують заводи з її утилізації. Крім того, в конструкції обладнання максимально збільшується частка матеріалів, придатних для переробки.

Вся оргтехніка включає в свій склад як органічні складові (пластик різних видів, скло, матеріали на основі полівінілхлориду, фенолформальдегіда), так і майже повний набір металів: золото, срібло, залізо, мідь, алюміній.

Також звичайний комп'ютер містить як цінні метали, так і небезпечні, такі як кадмій, свинець, цинк, нікель, а тому при списанні та утилізації обладнання необхідно керуватися законодавством в області охорони навколишнього середовища [20].

Утилізація оргтехніки та комп’ютерів – це процес, який проводиться в кілька етапів. Найперше дію це списання обладнання безпосередньо з підприємства. Етап другий – це розбір техніки і сортування отриманих матеріалів. Якщо деталі здатні служити вихідною сировиною, наприклад, деталі, в складі яких є дорогоцінні метали, то їх відправляють на очищення.

Сутність етапу вторинної переробки сировини полягає в тому, що можна витягти з цієї сировини частку корисних та рідких матеріалів, таких як іридій, мідь та інші. Цей процес відтворити набагато легше аніж видобути таку ж кількість металів, яка міститься в тисячотонних гірських породах.

Практично жодне підприємство не зможе самостійно утилізувати комп’ютери та оргтехніку, так як цей процес вимагає сучасного обладнання та специфічних знань. Тому довірити таку роботу можна тільки професіоналам, які мають великий досвід у даній сфері.

Проблема утилізації використаних комп’ютерів, периферійного обладнання, стає гострішою з кожним роком. Обсяги виробництва продуктів інформаційно-телекомунікаційних технологій та частота їх заміни на нові моделі примушують компанії замислюватись над проблемою біодеградації. Успіхи в цій галузі допоможуть, серед іншого, компаніям-виробникам зменшити податки, котрі вони сплачують зараз за утилізацію застарілих моделей. Останнє тим більше важливо, оскільки робить екологізацію економічно вигідною, тож спрямовує у цю сферу дедалі більше зусиль дослідників та довгострокових капіталовкладень. Таким чином, подальше поширення інформаційних технологій не збільшить, а навпаки – зменшить техногенне навантаження на довкілля [20].

Висновки

При роботі над дипломним проектом було розроблено програмний мультимедійний комплекс для проведення інтерактивних заходів з підтримкою аудіо- та відеоконтенту.

Було проведено аналіз задачі, визначено вимоги до програмного проекту, проаналізовано шляхи розв’язання проблеми, створено опис алгоритму програми. На базі опису алгоритму було розроблено блок-схему роботи програми, перелік основних та допоміжних функцій, схему взаємодії модулів програми. За розробленими проектними даними створено та відлагоджено програмний код.

Прикладна програма для проведення інтерактивних заходів з підтримкою аудіо- та відеоконтенту написана на мові програмування С++ у середовищі програмування C++Builder з використанням бібліотеки візуальних компонентів.

Програма може використовуватися як прикладний програмний продукт для проведення інтерактивних заходів ігрового типу з використанням різних режимів демонстрації та з підтримкою віддаленого керування.

Для подальшого розвитку програми можна удосконалити інтерфейс додатку за допомогою нових елементів навігації та відображення, реалізувати підтримку різних форматів медіаданих, модернізувати систему масштабування зображень, додати еквалайзер та регулятори параметрів аудіо- та відеопотоків, розробити систему підтримки файлів конфігурації, реалізувати кросплатформеність програмного продукту для віддаленої роботи з мобільних пристроїв.

Під час роботи над дипломним проектом були отримані нові знання щодо стандартів та нормативів, здобуті навички оформлення проектної документації, поглиблені теоретичні знання з програмування, які були закріплені на практиці при роботі з мовою програмування високого рівня C++ та візуальним програмним середовищем C++Builder.

Перелік посилань

1 С++. Основи програмування. Теорія та практика : підручник / О.Г. Трофименко, Ю.В. Прокоп, І.Г. Швайко, Л.М. Буката та ін. ; за ред. О.Г. Трофименко. – Одеса: Фенікс, 2010. – 544 с.

2 Архангельский А. Я. Программирование в C++ Builder / А. Я. Архангельский – М: Бином-Пресс, 2010. – 1034 с.

3 Безруков В.А. Win32 API. Программирование. Учебное пособие / В.А. Безруков. – СПб: СПбГУ ИТМО, 2009. – 90 с.

4 Вступ до програмування мовою С++. Організація обчислень : навч. посіб. / Ю. А. Бєлов, Т. О. Карнаух, Ю. В. Коваль, А. Б. Ставровський. – К. : Видавничо-поліграфічний центр "Київський університет", 2012. – 175 с.

5 Говорущенко Т.О. Проектування інтерфейсів користувача: Електронний конспект лекцій для студентів спеціальностей 7.05010202 (спеціалісти) і 8.05010202 (магістранти) – «Системне програмування» денної форми навчання / Укл. Т.О. Говорущенко – Хмельницький: ХНУ, 2012. – 209 с.

6 Культин Н.Б. С++ Builder в задачах и примерах / Н.Б. Культин – СПб.: БХВ-Петербург, 2010. - 336 с.: ил.

7 Лавріщева К.М. Програмна інженерія / К.М.Лавріщева – К.: Академперіодика, 2008. – 319 с.

8 Липпман С.Б. Язык программирования C++. Базовый курс / Стенли Б. Липпман, Жози Лажойе, Барбара Э. Му – М: Вильямс, 2014. – 1120 с.

9 Поморова О.В. Проектування інтерфейсів користувача: [навчальний посібник для студентів вищих навчальних закладів] / О.В. Поморова, Т.О. Говорущенко. – Хмельницький : ХНУ, 2011. – 206 с.

10 Прата С. Язык программирования C++. Лекции и упражнения, 6-е изд.: Пер. с англ. / Стивен Прата – М.: ООО «И.Д. Вильямс», 2012. – 1248 с.

11 Страуструп Б. Программирование. Принципы и практика с использованием C++. Второе издание / Б. Страуструп. - М.: Издательский дом "Вильямс", 2016. - 1328 с.

12 Тидвелл Дж. Разработка пользовательских интерфейсов. 2-е издание / Дженнифер Тидвелл – СПб.: Питер, 2011. – 480 с.

13 C++Builder. Интеренет-учебник для всех [Електронний ресурс] / Главная – Interface – Multimedia - Универсальный проигрыватель MediaPlayer. – Режим доступу: https://cubook.supernew.org/interfece/multimedia/268-universalnyj-proigryvatel-mediaplayer

14 Femil.org.ua [Електронний ресурс] / C++ Builder: Работа с TStringList. – Режим доступу: http://femil.org.ua/2013/02/c-builder-rabota-s-tstringlist/

15 High Level Language C++ [Електронний ресурс] / Создание плеера. – Режим доступу: http://h-l-l.ru/publ/28-1-0-125

16 Indy Knowlege Base. [Електронний ресурс] / How do I use a TIdUDPClient with a SOCKS Proxy? – Режим доступу: http://www.indyproject.org/kb/ index.html?udpwithsocks.htm

17 InterШкола. Дистанційне навчання [Електронний ресурс] / Охорона праці користувачів персональних комп’ютерів (ПК) – Режим доступу: http:// inter.ptngu.com/головна/оксм/охорона-праці/охорона-праці-користувачів- персонал

18 Исходники.ru [Електронний ресурс] / А. Маховикова // Inno Setup – Режим доступу: http://www.sources.ru/win/inno\_win.shtml

19 КиберАрхив. Технологии [Електронний ресурс] / Условия труда инженера-программиста. – Режим доступу: http://www.shapovalov.org/publ/4-1-0-2

20 Навчальні матеріали онлайн [Електронний ресурс] / Безпека життєдіяльності людини та суспільства. Правила безпеки при роботі з комп’ютером. – Режим доступу: http://pidruchniki.com/18210712/bzhd/pravila\_ bezpeki\_pri\_roboti\_kompyuterom

21 Обучалка C++ Builder [Електронний ресурс] / Мультимедийное приложение. Компонент проигрыватель мультимедиа и его свойства. – Режим доступу: http://bigcamagan.ru/urok-2/

22 Программирование с C++ Builder [Електронний ресурс] / Компонент MediaPlayer. – Режим доступу: http://www.ci-builder.ru/04/Index03.htm

23 Техніка безпеки [Електронний ресурс] / Техніка безпеки при роботі з комп’ютером. – Режим доступу: http://tex-bezbeka.in.ua/vpobut/N\_nev. php?nev=11