Національний технічний університет України
«Київський політехнічний інститут імені Ігоря
Сікорського» Міністерства освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря
Сікорського» Міністерства освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

РВАЧ ДМИТРО ВЯЧЕСЛАВОВИЧ

УДК 004.627

ДИСЕРТАЦІЯ

АЛГОРИТМІЧНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ ЦИФРОВИХ ДВІЙНИКІВ МУЛЬСЕМЕДІЙНИХ ОБ’ЄКТІВ

121 Інженерія програмного забезпечення
12 Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Д. В. Рвач
(підпис)

Науковий керівник: Сулема Євгенія Станіславівна, доктор технічних наук, доцент

Київ – 2024

АНОТАЦІЯ

Рвач Д. В. Алгоритмічне та програмне забезпечення технології цифрових двійників мульсемедійних об’єктів. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії в галузі знань 12 Інформаційні технології за спеціальністю 121 Інженерія програмного забезпечення. – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, 2024.

Цифрові двійники є технологією, що надають змогу створити цифрову копію фізичного об’єкту для подальшого аналізу та симуляції його поведінки. Технологія цифрових двійників пропонує новий підхід до представлення та обробки динамічної цифрової моделі фізичного об'єкта або процесу, його минулого, теперішнього і майбутніх станів та поведінки. Цифровий двійник являє собою сукупність віртуальних інформаційних структур, які повністю описують майбутній або реально існуючий фізичний об'єкт від мікрорівня (одноелементний рівень) до макрорівня (загальний вигляд, загальні властивості об'єкта в цілому). Технологія цифрових двійників сполучає конкретну фізичну систему із комп'ютерною моделлю, яка відображає архітектуру, динаміку і фактичний стан цієї конкретної системи. Давачі, що дозволяють здійснювати безперервний моніторинг об’єкта, можуть бути використанні як джерела інформації для створення таких індивідуалізованих динамічних моделей.

Аналіз існуючих програмних систем для роботи з цифровими двійниками показує, що вони є вузькоспрямованими на створення цифрових двійників специфічних об’єктів для конкретної галузі та не здатні використовувати більш універсальний підхід до створення цифрової копії реального об’єкту. Також значною мірою виявлена проблема

відсутності уніфікації та стандартизації форматів обміну даними між фізичною та цифровою копією об’єкта.

Розв’язання цих завдань можливе шляхом створення уніфікованої, гнучкої та стандартизованої програмної системи, що надає змогу створювати цифрові копії фізичних об’єктів з різними параметрами і налаштувати роботу з ними згідно з вимогами.

Технологію цифрових двійників звичайно використовують у галузі виробництва та у інженерії. Натомість, практично немає прикладів застосування цієї технології у галузі освіти та у медичній галузі. Тим не менш, застосування технології цифрових двійників у цих галузях має великий потенціал, оскільки за рахунок її використання можна досягти нової якості освіти та медичних послуг. Проте, для адаптації технології цифрових двійників до вимог цих галузей, особливостю яких є безпосередня участь людини у процесах, процедурах та сервісах, доцільно враховувати новітні технічні можливості, що спрямовані на цифрову обробку даних про фізичні властивості об’єктів навколишнього світу, які здатна сприймати людина. Тому актуальною задачею є розширення технології цифрових двійників за рахнок застосування концепції мульсемедіа, яка робить можливим сворення нових інтерфейсів людино-машинної взаємодії.

Описані завдання та тенденції визначають актуальну науково-технічну задачу вдосконалення та розвитку теоретичних основ створення алгоритмічного та програмного забезпечення технології цифрових двійників, що вирішується у цій дисертаційній роботі.

Метою дисертаційної роботи є підвищення ефективності обробки даних про стан мульсемедійних об’єктів у програмних системах на основі концепції цифрового двійника.

У першому розділі дисертаційної роботи проаналізовано методи, підходи та програмне забезпечення для обробки даних цифрових двійників

мульсемедійних об’єктів. Досліджено технологію цифрових двійників та концепцію мульсемедіа. Вивчено поточний стан наявних програмних систем на основі технології цифрових двійників. Виявлено недостатню універсальність підходів до створення цифрового двійника у вказаних програмних системах. У результаті визначено функціональні та нефункціональні вимоги для цього класу програмного забезпечення.

У другому розділі розроблено алгоритмічне та програмне забезпечення процесів консолідації мульсемедійних даних на основі концепції мультиобразу. Запропоновано метод консолідації мульсемедійних даних, який передбачає обробку мульсемедійних даних з використанням алгебраїчної системи агрегатів. Запропоновано розширення формату JSON для створення синхронізованої у часі структури мульсемедійних даних – TJSON. Використано дискретні вейвлет-перетворення та алгоритми стеганографії для поєднання даних різних модальностей в єдиний цифровий об’єкт. Запропоновано використання паралельних обчислень для підвищення ефективності реалізації запропонованого методу.

У третьому розділі розглянуто наявні підходи до обробки даних про мульсемедійні об’єкти, один з яких ґрунтується на застосуванні мови програмування ASAMPL. Запропоновано шляхи вдосконалення мови програмування ASAMPL, а саме: оновлено синтаксис, що надало можливість поліпшити метрики коду при програмуванні мовою ASAMPL; уведено нові програмні конструкції, що розширило можливості мови, зокрема, зробило можливим застосування паралельних обчислень. Зазначені зміни включені до нової версії мови програмування – ASAMPL 2.0. Також запропоновано специфічні шаблони проєктування, що уніфікують та стандартизують підходи до роботи з мульсемедійними об’єктами.

Четвертий розділ присвячено розробленню узагальненої архітектури програмних систем для обробки даних цифрових двійників

мульсемедійних об’єктів. На основі запропонованої узагальненої архітектури розроблено архітектуру цифрового двійника пацієнта, яка призначена для використання у медичних програмних системах, та архітектуру програмної системи для тренінгів фахівців, чия професійна діяльність пов’язана з підвищеним ризиком.

У дисертаційній роботі отримано низку **нових наукових результатів**, зокрема **уперше** запропоновано метод консолідації мульсемедійних даних, характерними рисами якого є застосування: концепції мультиобразу, кількісних відношень дискретних інтервалів, операцій алгебраїчної системи агрегатів, принципів стеганографії, дискретного вейвлет-перетворення, паралельних обчислень, що робить можливим поєднання даних різних модальностей в єдиний цифровий об’єкт.

Удосконалено теоретичні засади обробки темпоральних мультимодальних даних, які полягають у тому, що запропоновані кількісні відношення дискретних інтервалів, які на відміну від відношень інтервальної алгебри Аллена та відношень дискретних інтервалів в алгебраїчній системі агрегатів, дають змогу встановлювати кількісні темпоральні властивості наборів даних різних модальностей.

Уперше запропоновано узагальнену архітектуру програмної системи для обробки даних цифрових двійників мульсемедійних об’єктів, характерними рисами якої є оперування наборами темпоральних мультимодальних даних, які представлені як комплексна структура даних – мультиобраз мульсемедійного об’єкта, та забезпечення взаємодії з мульсемедійним об’єктом через спеціалізовані програмно-апаратні засоби (сенсори, актуатори, симулятори, рендери), що спрощує процеси розроблення нового покоління програмних систем на основі концепції цифрових двійників – мульсемедійного програмного забезпечення.

Уперше запропоновано архітектурні шаблони проєктування, які, на відміну від відомих, призначені для оперування комплексними структурами мульсемедійних даних – мультиобразами мульсемедійних об’єктів, що спрощує процеси розроблення мульсемедійного програмного забезпечення.

Основні наукові результати дисертаційної роботи **опубліковано** у 8 наукових працях, зокрема у 4 наукових статтях опублікованих у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б», та у 4 матеріалах наукових конференцій.

Ключові слова: програмне забезпечення, інженерія програмного забезпечення, мови програмування, інформаційні технології, алгоритми, цифрові двійники, мульсемедіа, мульсемедійний об’єкт, інформаційний об’єкт, консолідація даних, обчислення.

SUMMARY

Rvach D. Algorithms and Software for Digital Twin Technology of Mulsemedia Objects. – Qualifying scientific work, manuscript.

PhD thesis in the field of knowledge 12 Information technologies in speciality 121 Software Engineering. – National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, 2024.

Digital twins are a technology that is purposed for creation a digital copy of a physical object for further analysis and simulation of its behavior. The technology of digital twins offers a new approach to the presentation and processing of a dynamic digital model of a physical object or process, its past, present and future states and behavior. A digital twin is a collection of virtual information structures that fully describe a future or actually existing physical object from the micro level (single-element level) to the macro level (general appearance, general properties of the object as a whole). Digital twin technology connects a specific physical system with a computer model that reflects the architecture, dynamics, and actual state of that specific system. Sensors that enable continuous monitoring of the object can be used as sources of information to create such individualized dynamic models.

Analysis of existing software systems for working with digital twins shows that they are narrowly focused on creating digital twins of specific objects for a specific industry and are not able to follow a more universal approach to creating a digital copy of a real object. The problem of lack of unification and standardization of data exchange formats between the physical and digital copy of the object was also identified to a large extent.

Solving these tasks is possible by creating a unified, flexible and standardized software system that allows creating digital copies of physical objects with different parameters and adjusting work with them according to the necessary requirements.

Digital twin technology is commonly used in manufacturing and engineering. On the other hand, there are practically no examples of the application of this technology in both education and healthcare. Nevertheless, the application of the technology of digital twins in these industries has great potential, as it is possible to achieve a new quality of education and healthcare services due to its use. However, to adapt the technology of digital twins to the requirements of these industries, the feature of which is the direct participation of a human in processes, procedures and services, it is reasonable to take into account the latest technical capabilities aimed at digital processing of data on the physical properties of real-world objects that a human is able to perceive. Therefore, an urgent task is to expand the technology of digital twins by applying the concept of mulsemedia, which makes it possible to create new interfaces for human-machine interaction.

The purpose of the dissertation is to increase the efficiency of mulsemedia objects state data processing in software systems based on the concept of a digital twin.

In the first section of the dissertation, methods, approaches, and software for processing data of digital twins of mulsemedia objects were analyzed. The technology of digital twins and the concept of mulsemedia were studied. The current state of existing software systems based on the technology of digital twins was studied. Insufficient universality of approaches to creating a digital twin in the specified software systems was revealed. As a result, functional and non-functional requirements for this class of software were formed.

In the second section, algorithmic and software solutions for the consolidation of mulsemedia data based on the concept of multi-image have been developed. A method of mulsemedia data consolidation is proposed, which involves processing mulsemedia data using an algebraic system of aggregates. An extension of the JSON format to create a time-synchronized mulsemedia data structure named TJSON is proposed. Discrete wavelet transforms and

steganography algorithms were used to combine data from different modalities into a single digital object. It is proposed to use parallel calculations to increase the effectiveness of the implementation of the proposed method.

In the third section, available approaches to data processing of mulsemedia objects are considered, one of which is based on the application of the ASAMPL programming language. Ways to improve the ASAMPL programming language are proposed, namely: the syntax has been updated, which made it possible to improve code metrics when programming in the ASAMPL language; new software designs were introduced, which expanded the capabilities of the language, in particular, made it possible to use parallel calculations. The specified changes are included in the new version of the programming language named ASAMPL 2.0. Specific design patterns are also proposed, which unify and standardize approaches to working with mulsemedia objects.

The fourth section is dedicated to the development of a generalized architecture for software systems processing data of digital twins of mulsemedia objects. Based on this developed generalized architecture, the architecture of a patient's digital twin was created, intended for use in medical software systems. Also, on the basis of the proposed generalized architecture, the architecture of the software system for the training of specialists whose professional activity is associated with increased risk has been developed.

In the dissertation, a series of **new scientific results** were obtained. In particular, **for the first time**, a method for mulsemedia data consolidation, a characteristic features of which is the use of the multi-image concept, quantitative relations between discrete intervals, operations of the algebraic system of aggregates, steganography principles, discrete wavelet transformation, and parallel computing, was proposed that enables data fusion of different modalities in a joint digital object.

The theoretical foundations of temporal multimodal data processing **have been improved**, which consist in the fact that the proposed quantitative relations between discrete intervals, unlike Allen's interval algebra relations and discrete intervals relationship in the algebraic system of aggregates, enable establishing quantitative features of multimodal data sets.

For the first time, a generalized architecture of software systems for mulsemedia object digital twin data processing was proposed, characteristic features of which are both handling temporal multimodal data sets, which are presented as a complex data structure – a mulsemedia object multi-image, and enabling the interaction with the mulsemedia object through specialized software and hardware (sensors, actuators, simulators, renders) that together simplify the development of the new generation of software systems based on the technology of digital twins – mulsemedia software.

For the first time, architectural software design patterns, which unlike the existing patterns enable handling a complex structure of mulsemedia data named mulsemedia object’s multi-image, are proposed that allows simplifying the development of mulsemedia software.

The main scientific results of the dissertation have been **published** in eight scientific works, including four articles were published in professional journals included in the list of scientific professional publications of Ukraine with the category "Б" assigned. Additionally, the work is featured in four materials of scientific conferences.

Keywords: software, software engineering, programming languages, information technologies, algorithms, digital twins, mulsemedia, mulsemedia object, information object, data consolidation, computing.

Список публікацій здобувача / List of publications of the applicant:

Статті у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б» / the articles published in scientific journals included in the list of scientific journals of Ukraine in category «Б»:

1. Sulema Ye., Rvach D. Models of computation for Digital Twins data processing. // Наукові вісті КПІ: міжнародний науково-технічний журнал. – 2020. – № 2 (129). – P. 44-81.
   DOI: 10.20535/kpi-sn.2020.2.205131
2. Dmytro R., Sulema Ye. Mulsemedia data consolidation method. // System technologies. – 2022. – Vol. 6, No. 143. – P. 69-79.
   DOI: 10.34185/1562-9945-6-143-2022-06
3. Sulema Ye., Rvach D. High-level architecture of mulsemedia software system // Вісник Хмельницького національного університету Серія: «Технічні науки». – 2023. – № 5 Т2 (143). – P. 121-125.
   DOI: 10.31891/2307-5732-2023-327-5-121-125
4. Sulema Ye., Rvach D. Formal specification of mulsemedia object's digital twin based on discrete intervals temporal relations // Computer Systems and Information Technologies. – 2023. – № 4. – P. 60-66.
   DOI: 10.31891/csit-2023-4-8

Публікації у матеріалах наукових конференцій / the publications in the proceedings of the scientific conferences:

5. Dychka, I., Sulema, Ye., Rvach, D., Drozdenko, L. Programming Language ASAMPL 2.0 for Mulsemedia Applications Development. // Lecture Notes on Data Engineering and Communications Technologies. – 2022. – Vol. 134. – P. 107-116.
   ISSN 2367-4512

6. Рвач Д. В. Технології та методи цифрових двійників // Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення. – 2022. – № 68. – С. 98-99.
   ISSN 2522-932X
7. Рвач Д. В. Аналіз даних для опису мульсемедійних об'єктів // Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення. – 2023. – № 74. – С. 183-186
   ISSN 2522-932X
8. Рвач Д. В., Сулема Є. С. Шаблони проєктування для розроблення мульсемедійного програмного забезпечення мовою ASAMPL 2.0 // Шістнадцята наукова конференція магістрантів та аспірантів «Прикладна математика та комп’ютинг (ПМК’2023)», Київ, 28 - 30 листопада 2023 р. Збірник тез доповідей Нац. техн. ун-т України «Київ. політехн. ін-т ім. Ігоря Сікорського», Київ: Просвіта. – 2021. – С. 580-584.
   ISBN 978-617-7010-14-1

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AR – Augmented reality (доповнена реальність)

ASA – Algebraic System of Aggregates (алгебраїчна система агрегатів, АСА)

ASAMPL – Algebraic System of Aggregates and Mulsemedia data Processing Language (мова алгебраїчної системи агрегатів та обробки мульсемедійних даних)

AWS – Amazon Web Services (веб-сервіси Амазон)

CSV – Comma Separated Values (значення розділені комою)

DCT – Discrete Cosine Transform (дискретне косинусне перетворення)

DCT – Discrete Fourier Transform (дискретне перетворення Фур'є)

DCT – Discrete Wavelet Transform (дискретне вейвлет перетворення)

EBNF – Extended Backus-Naur Form (розширена форма Бекуса-Наура)

HDWT – Haar Discrete Wavelet Transform (DWT Гаара)

IoT – Internet of Things (Інтернет речей)

JPEG – Joint Photographic Experts Group (Об'єднана експертна група з фотографій)

JSON – JavaScript Object Notation (запис об'єктів JavaScript)

MI – Multi-image (Мультиобраз, МО)

MPEG – Moving Pictures Experts Group (експертна група з питань рухомого зображення)

MPEG-V – Moving Pictures Experts Group Virtuality (експертна група з питань рухомого зображення – віртуальність)

MP3 – MPEG Audio Layer 3 (MPEG аудіо шар 3)

MP4 – MPEG-4 Part 14 (MPEG-4 частина 14)

MR – Mixed Reality (змішана реальність)

NASA – National Aeronautics and Space Administration (Національне управління з аеронавтики і дослідження космічного простору)

LSB – Least Significant Bits (найменш значущі біти)

PLM – Product Lifecycle Management (управління життєвим циклом продукту)

PSNR – Peak Signal to Noise Ratio (пікове співвідношення сигналу до шуму)

VR – Virtual Reality (віртуальна реальність)

SMI – Sub Multi-image (суб-мультиобраз)

SR – Syntaxes Rule (синтаксичне правило)

SSIM – Structure Similarity (індекс структурної подібності)

TJSON – Timeline JavaScript Object Notation (запис темпоральних об’єктів JavaScript)

UML – Unified Modeling Language (уніфікована мова моделювання)

ДІ – дискретний інтервал

ДСНС – Державна служба з надзвичайних ситуацій

МРТ – магнітно-резонансна томографія

ОС – операційна система

ВСТУП

Актуальність теми. Технологія цифрових двійників ґрунтується на використанні динамічних цифрових інформаційних моделей, які відображають фізичні об’єкти або процеси, їхній минулий, поточний та майбутній стан і поведінку. Цифровий двійник є комплексною моделлю, яка створюється відповідно до вимог і специфіки предметної галузі та реалізується програмним шляхом.

Технологія цифрових двійників застосовується у широкому спектрі галузей, включаючи виробництво, логістику, енергетику тощо. Застосування технології цифрових двійників надає змогу збирати, оброблювати та відображати значні обсяги даних з метою подальшого аналізу та прогнозування, що важливо для підвищення ефективності та оптимізації ресурсів. Наприклад, у сфері виробництва цифрові двійники можуть бути використані для моделювання виробничих процесів, оптимізації ліній збирання, підвищення якості продукції, а також для прогнозування та запобігання виробничих збоїв. У логістиці застосування технології цифрових двійників сприяє оптимізації ланцюгів постачання, покращуючи точність прогнозування та управління запасами.

Іншим важливим аспектом застосування технології цифрових двійників є забезпечення сталості та стійкості систем. Цифрові двійники надають можливість симулювати різні сценарії та оцінювати вплив на довкілля, що є важливим для екологічної безпеки та сталого розвитку.

Технологія цифрових двійників також має потенціал для застосування в освіті та галузі охорони здоров’я для забезпечення подільшої цифровізації та застосування нових інноваційних рішень, що розширять спектр та підвищать якість послуг. Специфікою цих галузей є їхня людиноцентрованість, оскільки саме людина є основним учасником усіх процесів та кінцевим споживачем послуг. Тому застосування

технології цифрових двійників у цих галузях вимагає її адаптації та вдосконалення.

Зі створенням концепції мульсемедіа (mulsemedia – multiple sensorial media) та розвитком технології віртуальної (VR) та доповненої реальності (AR) з'являються нові можливості для удосконалення технології цифрових двійників та розширення їхньої сфери застосування. Зокрема, запровадження нових підходів на основі концепції цифрових двійників мульсемедійних об'єктів у медицині сприяє переходу до більш ефективної телемедицини, де лікарі могли б, наприклад, планувати хірургічні втручання на відстані, використовуючи цифрові моделі пацієнтів. Такий підхід не лише підвищіть доступність медичних послуг, але й дозволить лікарям віддалено обмінюватися досвідом та знаннями, використовуючі нові цифрові технології. Крім того, використання цифрових двійників у поєднанні з VR/AR може підвищити точність медичних процедур шляхом забезпечення віртуального відображення фізичного стану пацієнтів.

У сфері освіти використання цих технологій надасть можливість створення імерсивних навчальних програм, які забезпечують студентам можливість набувати практичного досвіду в реалістичних, але контрольованих умовах. Це особливо важливо у таких галузях, як медицина, інженерія або фізика, де практичний досвід часто обмежений через високу вартість або ризикованість реальних експериментів. Імерсивні VR/AR-симуляції надають можливість студентам експериментувати та вчитися з помилок без реальних наслідків для об'єкту дослідження або ризику для самих студентів.

Проте, аналіз наявних програмних систем для роботи з цифровими двійниками показує, що вони є переважно спеціалізованими, корпоративними та не вирішують узагальнених завдань в контексті створення цифрових копій фізичних об'єктів різного роду. Недостатнім є ступінь універсальності підходів, адже кожна із наявних програмних

систем пропонує лише спеціалізований інструментарій для конкретної галузі і не може бути використана для створення цифрового двійника, що не є пов’язаним з цією галуззю.

Вирішення цієї проблеми можливе шляхом створення уніфікованої, гнучкої і стандартизованої архітектури програмної системи, що надає змогу створювати цифрові копії фізичних об’єктів, зокрема, мульсемедійних, з різними параметрами і налаштувати роботу з ними згідно з вимогами.

Таким чином, актуальною є науково-технічна задача вдосконалення теоретичних засад розроблення алгоритмічного та програмного забезпечення на основі концепції цифрових двійників, що призначене для обробки темпоральних мультимодальних даних про мульсемедійні об’єкти.

Зв’язок роботи з науковими програмами, планами, темами. Дослідження за темою дисертаційної роботи провадились у Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського» в рамках виконання держбюджетної науково-дослідної роботи «Математичні та програмні методи оброблення мультимодальних даних моніторингу медико-біологічних об’єктів для діагностики стану здоров’я пацієнтів» (номер державної реєстрації 0120U102134).

Мета і задачі дослідження. Метою дисертаційної роботи є підвищення ефективності обробки даних про стан мульсемедійних об’єктів у програмних системах на основі концепції цифрового двійника.

Відповідно до поставленої мети основними задачами дослідження є:

- аналіз програмного забезпечення на основі технології цифрових двійників;
- формування вимог до програмного забезпечення для створення цифрових двійників мульсемедійних об’єктів;
- розроблення алгоритмічного та програмного забезпечення процесів консолідації мульсемедійних даних;

- аналіз та вдосконалення мови програмування ASAMPL шляхом оновлення синтаксису та розширення функціональності;
- розроблення узагальненої архітектури програмної системи для обробки даних цифрових двійників мульсемедійних об’єктів;
- розроблення архітектурних шаблонів для оперування мультиобразами мульсемедійних об’єктів.

Об’єкт дослідження – процеси розроблення програмних систем на основі концепції цифрових двійників.

Предмет дослідження – методи розроблення алгоритмічного та програмного забезпечення мульсемедійних систем на основі концепції цифрових двійників.

Методи дослідження: теорія програмування, теорія програмних систем, теорія інформаційних систем, теорія алгоритмів, теорія паралельних і розподілених обчислень, теорія алгебраїчної системи агрегатів.

Наукова новизна одержаних результатів зводиться до наступного:

1. **Уперше** запропоновано метод консолідації мульсемедійних даних, характерними рисами якого є застосування: концепції мультиобразу, кількісних відношень дискретних інтервалів, операцій алгебраїчної системи агрегатів, принципів стеганографії, дискретного вейвлет-перетворення, паралельних обчислень, що робить можливим поєднання даних різних модальностей в єдиний цифровий об’єкт.
2. **Удосконалено теоретичні засади** обробки темпоральних мультимодальних даних, які полягають у тому, що запропоновані кількісні відношення дискретних інтервалів, які на відміну від відношень інтервальної алгебри Аллена та відношень дискретних інтервалів в алгебраїчній системі агрегатів, дають змогу встановлювати кількісні темпоральні властивості наборів даних різних модальностей.

3. **Уперше** запропоновано узагальнену архітектуру програмної системи для обробки даних цифрових двійників мульсемедійних об’єктів, характерними рисами якої є оперування наборами темпоральних мультимодальних даних, які представлені як комплексна структура даних – мультиобраз мульсемедійного об’єкта, та забезпечення взаємодії з мульсемедійним об’єктом через спеціалізовані програмно-апаратні засоби (сенсори, актуатори, симулятори, рендери), що спрощує процеси розроблення нового покоління програмних систем на основі концепції цифрових двійників – мульсемедійного програмного забезпечення.
4. **Уперше** запропоновано архітектурні шаблони проєктування, які, на відміну від відомих, призначені для оперування комплексними структурами мульсемедійних даних – мультиобразами мульсемедійних об’єктів, що спрощує процеси розроблення мульсемедійного програмного забезпечення.

Практичне значення одержаних результатів полягає у спрощенні процесів розроблення мульсемедійного програмного забезпечення на основі концепції цифрових двійників за рахунок застосування запропонованого у дисертаційній роботі алгоритмічно-програмного забезпечення, яке включає: метод консолідації мульсемедійних даних, архітектурні шаблони проєктування «Мульсемедійний конфігуратор», «Мульсемедійний патерн», «Мульсемедійний композитор», «Мульсемедійний адаптер» та мову програмування ASAMPL 2.0.

Особистий внесок здобувача. Усі основні результати дисертаційного дослідження, які представлені до захисту, одержані автором особисто. У публікаціях, написаних у співавторстві, здобувачеві належать наступні результати. У роботі [1] здобувачем виконано програмну реалізацію моделі обчислень для обробки даних цифрових двійників, а також запропоновано процедуру мультимодальної синхронізації даних. У

роботі [2] здобувачем запропоновано метод консолідації мульсемедійних даних та формат TJSON (розширення формату JSON). У роботі [3] здобувачем запропонована узагальнена архітектура для програмних систем, що обробляють дані цифрових двійників мульсемедійних об’єктів. У роботі [4] здобувачем запропоновано формальну специфікацію цифрового двійника мульсемедійного об’єкту на основі кількісних темпоральних відношень дискретних інтервалів.

Апробація результатів дисертації. Основні результати дисертаційного дослідження доповідалися та обговорювалися на наукових конференціях:

1. Міжнародна наукова конференція «International Conference on Computer Science, Engineering and Education Applications», 21 – 22 лютого 2022 р. Київ, Україна
2. Міжнародна наукова конференція «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення», 7 - 8 червня 2022 р. Тернопіль, Україна.
3. Міжнародна наукова конференція «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення», 6 - 7 лютого 2023 р. Тернопіль, Україна – Переворськ, Польща.
4. Шістнадцята наукова конференція магістрантів і аспірантів «Прикладна математика та комп’ютинг (ПМК’2023)», Київ, 28 - 30 листопада 2023 р. Київ, Україна.

Публікації. Основні наукові результати дисертаційної роботи опубліковано у 8 наукових працях, зокрема у 4 наукових статтях опублікованих у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б», та у 4 матеріалах наукових конференцій.

Структура роботи. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел, що включає

102 найменування, та чотири додатки. Загальний обсяг роботи становить 199 сторінок, у тому числі 144 сторінок основного тексту, 23 рисунки, 8 таблиць.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ, ПІДХОДІВ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОБРОБКИ ДАНИХ ЦИФРОВИХ ДВІЙНИКІВ МУЛЬСЕМЕДІЙНИХ ОБ'ЄКТІВ

З розвитком обчислювальних потужностей та появою різноманітних видів сенсорів, здатних фіксувати численні характеристики об'єкта в реальному часі, виникла можливість оцифрування фізичних об'єктів і створення їх цифрових копій – цифрових двійників. Зокрема, завдяки розробленню спеціалізованого апаратного забезпечення [1–30] стає можливим створювати *цифрові двійники мульсемедійних об'єктів*, які є моделями, що відтворюють фізичний об'єкт у такий спосіб, щоб користувач міг комплексно відчувати фізичні властивості об'єкта через свої органи чуття, причому для цього можуть бути задіяні не лише зір та слух, а й залучене відчуття дотику, кінестетичні та термосептичні відчуття, нюх тощо. Такий підхід значно розширює сферу застосування технології цифрових двійників та створює умови для започаткування нового класу прикладного програмного забезпечення – *мульсемедійних програмних систем*, які призначені для віддаленої взаємодії людини із фізичними об'єктами зі забезпеченням повноти сприйняття цих об'єктів через органи чуття. Застосування таких систем надасть нові можливості для телемедицини, дистанційного навчання, спеціалізованих тренінгів для фахівців ризикованих професій, а також досліджень та робіт, пов'язаних з небезпечними середовищами та об'єктами.

1.1. Аналіз технології цифрових двійників та мульсемедіа

Для проведення аналізу наявних методів, підходів та програмного забезпечення, що використовуються для обробки даних цифрових

двійників мульсемедійних об'єктів сформулюємо поняття цифрового двійника, мульсемедіа та мульсемедійного об'єкта.

Визначення 1.1. *Цифровий двійник* – це комплексна віртуальна модель фізичного об’єкта чи процесу, яка динамічно відображає його характеристики та стан у реальному часі, що дає змогу моделювати поведінку об'єкта та проводити аналіз параметрів його стану [31].

Визначення 1.2. *Мульсемедіа* – це технологія, яка реалізує людиноцентрованій підхід, що передбачає реєстрацію, подання, обробку та відтворення мультимодальної інформації про фізичні об’єкти, яка сприймається людиною через органи чуття [32].

Визначення 1.3. *Мульсемедійний об’єкт* – це фізичний об’єкт або група семантично-пов’язаних фізичних об’єктів, стан яких фіксується за допомогою набору сенсорів для формування темпорального мультимодального цифрового опису, який визначає цей об’єкт (групу об’єктів) настільки повно та комплексно, наскільки він може бути сприйнятий людиною через органи чуття [33].

Аналіз основних принципів, методів і характеристик програмного забезпечення технології цифрових двійників та мульсемедіа має надати розуміння того, як збирати, обробляти та аналізовувати дані про мульсемедійні об’єкти та як ефективно використовувати наявні інструменти для створення цифрових двійників мульсемедійних об’єктів, а також сформулювати вимоги для розроблення нових, більш точних та ефективних методів обробки даних, що відповідають сучасним вимогам і тенденціям у галузі цифрових технологій.

1.1.1. Технологія цифрових двійників

Концепція цифрового двійника була вперше сформульована Майклом Грівсом, який запропонував цифровий двійник як концептуальну модель,

що лежить в основі управління життєвим циклом продукції (PLM) [34]. Першим практичним застосуванням цієї концепції стало її використання NASA для поліпшення моделювання фізичних моделей космічних кораблів [35]. Подальше впровадження цифрових двійників стало інтегральною частиною четвертої промислової революції, сприяючи підвищенню ефективності бізнес-процесів, швидкому виявленню фізичних недоліків, точнішому прогнозуванню їх наслідків та покращенню якості продукції [36].

Цифровий двійник включає три основні складові – це поведінкова модель фізичного об’єкта, його візуальна модель та інформаційні потоки, які забезпечують зв’язок між цифровим двійником та фізичним об’єктом. Взаємозв’язок між фізичним та цифровим об’єктами реалізується через обмін даними, що дозволяє оптимізувати процеси в реальному середовищі. Цей зв’язок надає змогу точно відслідковувати та аналізовувати стан фізичного об'єкта, вносячи зміни у віртуальну модель у реальному часі.

Цифровий двійник можна визначити як динамічно модифіковану цифрову модель, що інкорпорує як історичні, так і сучасні дані про фізичний об’єкт чи процес. Основою цифрового двійника є масивний обсяг даних, зібраних через вимірювання різноманітних параметрів об'єкта у реальному середовищі. Подальший аналіз даних, отриманих від цифрового двійника, дає можливість ідентифікувати поведінкові аномалії у фізичному двійнику, що сприяє своєчасному попередженню потенційних аварійних ситуацій.

Візуальна та поведінкова моделі цифрового двійника [36] ґрунтуються на відповідних математичних моделях та забезпечують синхронізацію між реальним та віртуальним об’єктами через неперервний моніторинг даних від сенсорів, що були встановлені для постійного спостереження за досліджуваним фізичним об’єктом. Таким чином,

цифровий двійник є індивідуалізованою реалістичною віртуальною моделлю певного досліджуваного об’єкта – фізичного двійника (рис. 1.1).

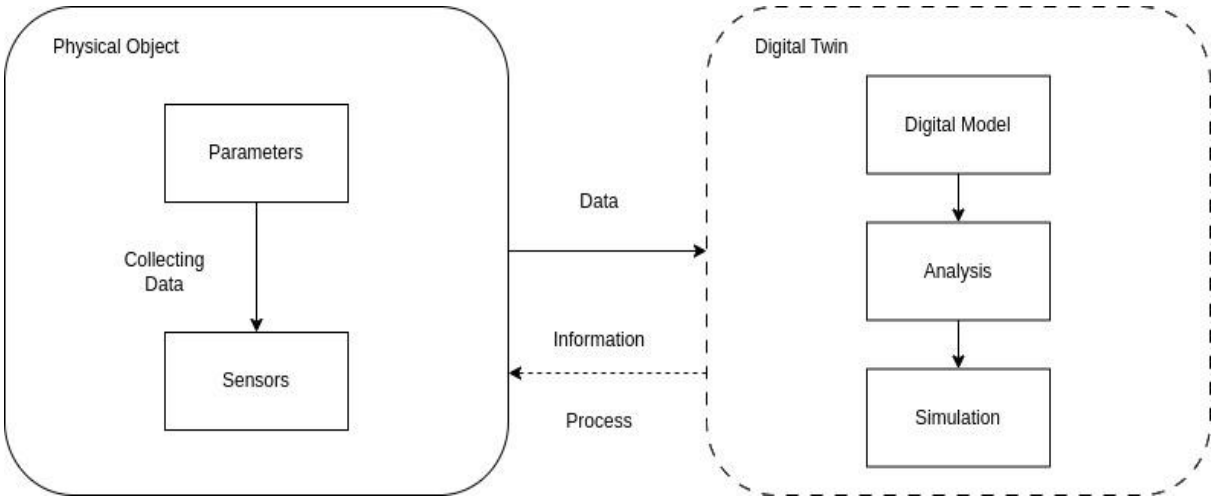

Рис 1.1 – Базовий принцип роботи цифрового двійника

За рівнем інтеграції цифрового та фізичного двійників розрізняють наступні види цифрових двійників [37]:

- цифрова модель (Digital Model);
- цифрова тінь (Digital Shadow);
- цифровий двійник (Digital Twin).

Цифрова модель – це цифрове подання наявного або запланованого фізичного об'єкта, яке не використовує жодної форми автоматизованого обміну даними між фізичним об'єктом та цифровим об'єктом. Ця модель передбачає використання імітаційних моделей, математичних моделей або інших видів моделювання фізичного об'єкта, коли автоматична інтеграція даних відсутня. Цифрові дані наявних фізичних систем можуть використовуватися для розроблення таких моделей, але обмін даними здійснюється вручну. Зміна стану фізичного об'єкта не має прямого впливу на цифровий об'єкт і навпаки [37].

Цифрова тінь – це цифрове подання фізичного об'єкта, при якому передбачено автоматизований односторонній потік даних між існуючим фізичним об'єктом та цифровим об'єктом. Зміна стану фізичного об'єкта призводить до зміни стану цифрового об'єкта, але не навпаки [37].

Цифровий двійник – це цифрове подання фізичного об'єкта, при якому потоки даних між існуючим фізичним об'єктом і цифровим об'єктом наявні в обох напрямках. Такий цифровий об'єкт може розглядатись як контрольний примірник фізичного об'єкта. Також можливе одночасне використання інших об'єктів, фізичних чи цифрових, які можуть викликати зміни стану в цифровому об'єкті. Зміна стану фізичного об'єкта безпосередньо призводить до зміни стану цифрового об'єкта і навпаки [37].

За рівнем складності візуальної моделі розрізняють цифрові двійники чотирьох рівнів (табл. 1.1) [37]:

- підготовчій цифровий двійник (Pre-Digital Twin);
- цифровий двійник (Digital Twin);
- адаптивний цифровий двійник (Adaptive Digital Twin);
- інтелектуальний цифровий двійник (Intelligent Digital Twin).

Аналіз класифікації цифрових двійників за рівнем інтеграції цифрового та фізичного двійників та / або за рівнем складності візуальної моделі надає змогу стверджувати, що цифрові двійники мульсемедійних об’єктів з точки зору їхнього можливого застосування можуть відноситись до таких типів:

- *цифрова тінь* та *цифровий двійник (2-й рівень складності)* – для використання у медицині (моделювання органів людини для підготовки до планових операцій складного рівня);
- *адаптивний цифровий двійник (3-й рівень складності)* – для використання у технічній та медичній освіті у дистанційному форматі (моделювання та взаємодія з технічними об’єктами або моделювання та симуляція взаємодії з біологічними об’єктами), а також для

використання у практичних тренінгах фахівців ризикованих професій (моделювання та симуляція небезпечних середовищ та об’єктів);

- *інтелектуальний цифровий двійник (4-й рівень складності)* – для використання у наукових дослідженнях (моделювання та взаємодія з фізично віддаленими та небезпечними об’єктами).

Таблиця 1.1 – Рівні складності цифрових двійників

Рівень цифрового двійника	Складність моделі	Зв’язок з фізичним двійником	Застосування машинного навчання
1-й рівень. Підготовчий цифровий двійник	Спрощена віртуальна модель	Не передбачено	Відсутнє
2-й рівень. Цифровий двійник	Віртуальна модель фізичного двійника	Пакетні дані	Відсутнє
3-й рівень. Адаптивний цифровий двійник	Віртуальна модель фізичного двійника з адаптивним користувацьким інтерфейсом	Дані у реальному часі	Частково
4-й рівень. Інтелектуальний цифровий двійник	Віртуальна модель фізичного двійника з адаптивним користувацьким інтерфейсом та застосуванням алгоритмів навчання з підкріпленням	Пакетні дані / дані у реальному часі	Наявне

Цифрові двійники мульсемедійних об’єктів також можуть бути застосовані для моделювання складних систем і сценаріїв для визначення стратегії інноваційного розвитку в багатьох галузях людської діяльності, включаючи медицину, науку, освіту, виробництво, сферу послуг тощо.

1.1.2. Технологія мульсемедіа та її застосування

Технологія мульсемедіа, що передбачає інтеграцію розширеного спектру сенсорних даних, надає нові перспективи в інтерактивному сприйнятті інформації, значно поглиблюючи досвід користувачів та забезпечуючи більш комплексну та багатогранну взаємодію з контентом. На відміну від технологій мультимедіа, що передбачає використання звуку, зображень та тексту, технологія мульсемедіа передбачає інтеграцію до ціфрового контенту додаткових сенсорних відчуттів, таких як запах, смак, тактильні відчуття тощо [39]. Це забезпечує перетворення стандартного цифрового контенту на комплексне цифрове подання інформації про об’єкт (рис 1.2).

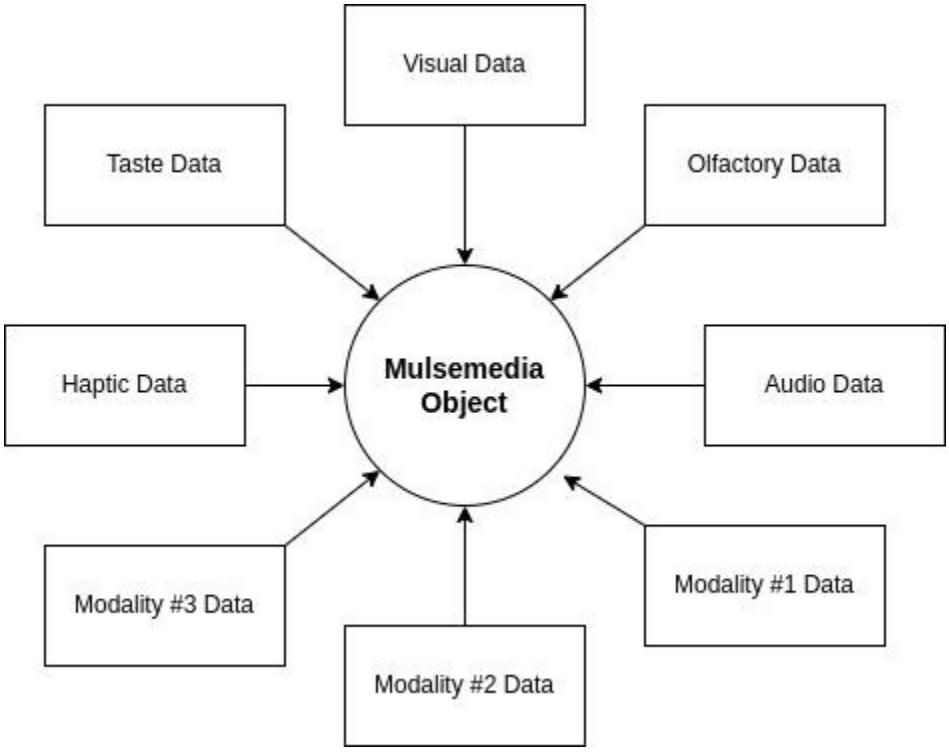

Рис 1.2 – Базова концепція мульсемедіа

Застосування мульсемедійних технологій надає значні можливості в різних сферах, включаючи освіту, медицину та телекомунікації. Наприклад,

у галузі медичної освіти технологія мульсемедіа може використовуватися для створення реалістичних тренажерів, що допомагають майбутнім медичним працівникам вдосконалювати майстерність у проведенні медичних процедур. У сфері інженерної освіти, інтерактивні мульсемедійні матеріали можуть підвищити залученість студентів та покращити процес навчання, надаючи можливість взаємодії з навчальним матеріалом на новому рівні. Корпоративний сектор може використовувати мульсемедійні технології для підвищення ефективності тренінгів та презентацій, забезпечуючи високий рівень взаємодії та залученості співробітників.

З огляду на широкий спектр застосування мульсемедійні системи вимагають можливості застосування спеціалізованого апаратного забезпечення [1-30] для відтворення тактильних, смакових, запахових та інших сенсорних ефектів. Також має бути можливою інтеграція з іншими сучасними технологіями [38], серед яких технології віртуальної та доповненої реальності, зокрема, з використанням VR-шоломів, що оснащені сенсорами руху та трекерами погляду, які забезпечують глибоке занурення в цифрове середовище. Доповненням до цього можуть бути тактильні рукавички, які дозволяють користувачам «відчувати» віртуальні об’єкти, надаючи новий рівень реалістичності взаємодії з віртуальним світом.

Наприклад, застосування VR-шоломів та тактильних рукавичок у телемедицині надає нові перспективи для медичної допомоги на відстані. Тактильні рукавички, оснащені сенсорами, дозволятимуть хірургу відчувати фізичні властивості тканин пацієнта, а також здійснювати точні рухи, необхідні під час операції завдяки високоточним сенсорам, які передають детальну інформацію про рухи рук хірурга. Додатково, інтеграція температурних сенсорів може надавати інформацію про

температуру тіла пацієнта чи інструментів, що використовуються під час операції.

Мульсемедійні системи можуть бути налаштовані та оптимізовані для різних застосувань, від простих освітніх програм до складних індустріальних застосувань, забезпечуючи користувачам не лише доступ до інформації, але й сенсорні відчуття. Таким чином, мульсемедіа є не тільки актуальною, але й перспективною технологією, здатною допомогти оптимізувати процеси та запропонувати нові підходи в різних сферах діяльності та перевести їх на новий рівень.

1.2. Аналіз програмних систем для створення цифрових двійників

Розглянемо програмні системи для роботи з технологією цифрових двійників, які вже існують та наявні у відкритому доступі. Ці системи призначені для створення точних, інтерактивних та функціонально багатих віртуальних моделей. Аналіз цих систем має на меті визначення можливості їхнього застосування для створення цифрових двійників мульсемедійних об’єктів.

Сучасні розробки за технологією цифрових двійників включають різноманітні платформи та рішення, які забезпечують широкий спектр можливостей для створення та використання цифрових двійників. Однією з ключових характеристик цих систем є їх гнучкість у відтворенні складних систем, здатність інтегрувати великі обсяги даних, а також можливість використання аналітичних інструментів для аналізу отриманих моделей.

Серед найвідоміших та актуальних систем, які зараз застосовуються у різних галузях, можна виділити наступні: Azure Digital Twins, AWS IoT TwinMaker, NVIDIA Omniverse та Supply Chain Twin. Кожна з цих платформ має свої унікальні особливості та можливості, а також недоліки.

1.2.1. Платформа Azure Digital Twins

Платформа Azure Digital Twins компанії Microsoft втілює концепцію інтегрованих цифрових репрезентацій фізичних об'єктів і процесів, що надає можливість використовувати принципово нові підходи в різних секторах, від міського планування до промислового виробництва [39].

Однією з ключових характеристик Azure Digital Twins є її здатність об'єднувати реальні дані з фізичного середовища та динамічно відображати їх у віртуальних моделях. Такий підхід дозволяє користувачам візуалізовувати складні системи й проводити аналіз різних сценаріїв, оптимізувати процеси та передбачати можливі проблеми. Платформа надає інструменти збору даних та аналітики, які допомагають у прийнятті обґрунтованих рішень на основі точних моделей [39].

Платформа Azure Digital Twins забезпечує гнучкість у створенні та масштабуванні цифрових двійників. Платформа Azure Digital Twins підтримує велику кількість IoT-сенсорів та пристроїв, що забезпечує високу точність та актуальність даних у цифрових моделях [39].

Одним з основних обмежень платформи Azure Digital Twins є її спеціалізація на певних типах бізнес-концепцій та обмежена здатність до створення цифрових двійників для широкого діапазону фізичних об'єктів. Це особливо критично у сферах застосування технології цифрових двійників, де необхідне детальне моделювання фізичних властивостей, таких як біомедична інженерія або розроблення складних механічних систем.

Іншим значним недоліком Azure Digital Twins є її залежність від екосистеми Microsoft, що може створювати перешкоди для інтеграції з системами та інструментами, що ґрунтуються на інших платформах. Це може обмежити можливість масштабування проєктів, які потребують взаємодії з різноманітними джерелами даних або програмним

забезпеченням, що не належить до продуктів Microsoft. Така ізоляція від інших технологічних рішень може призвести до додаткових витрат на розроблення спеціальних адаптерів або інтерфейсів для забезпечення сумісності. Отже, хоча Azure Digital Twins надає потужні можливості для створення цифрових двійників, вона може вимагати більше ресурсів та зусиль для інтеграції у складні мультиплатформні екосистеми.

Таким чином, складність інтеграції з іншими системами, висока залежність від продуктів Microsoft та висока вартість можуть ускладнювати впровадження та ефективне використання платформи Azure Digital Twins. Також, обмежена універсальність платформи в контексті створення цифрових двійників різних фізичних об'єктів вимагає розгляду альтернативних рішень для створення цифрових двійників мульсемедійних об’єктів.

1.2.2. Платформа AWS IoT TwinMaker

AWS IoT TwinMaker компанії Amazon розроблена з метою спрощення процесу створення цифрових двійників, тому надає інтуїтивно зрозумілі інструменти для візуалізації, моніторингу та оптимізації різних систем та процесів. Використовуючи потужність хмарних обчислень Amazon Web Services, AWS IoT TwinMaker дозволяє інтегрувати дані з різних джерел, включаючи IoT-сенсори та існуючі бізнес-системи, для створення точних та інтерактивних цифрових моделей [40].

Основною перевагою AWS IoT TwinMaker є її здатність обробляти великі обсяги даних, що надходять з різноманітних джерел, та інтегрувати ці дані в єдину віртуальну модель. Платформа також пропонує розширені можливості для налаштування, дозволяючи користувачам створювати унікальні рішення, які відповідають специфічним потребам [9].

AWS IoT TwinMaker відрізняється гнучкістю у використанні, пропонуючи інтеграцію з широким спектром інших сервісів AWS, що відкриває можливості для розроблення комплексних цифрових двійників. Використання хмарних технологій також забезпечує масштабованість та високу доступність платформи, що робить її хорошим вибором для проєктів будь-якого розміру. Завдяки цим функціям, AWS IoT TwinMaker може бути використаний у широкому діапазоні галузей, від промисловості до управління міською інфраструктурою [40].

Одним з ключових недоліків платформи AWS IoT TwinMaker є її спеціалізація на певних секторах бізнесу, що обмежує її універсальність у створенні цифрових двійників для широкого спектру фізичних об'єктів. Хоча ця платформа може ефективно використовуватись у задачах, де потрібен високий рівень інтеграції даних з IoT-пристроїв, вона може не відповідати потребам проєктів, які вимагають детального моделювання більш складних або специфічних фізичних об'єктів, як-от цифрові двійники мульсемедійних об’єктів.

Іншою проблемою AWS IoT TwinMaker є потенційні проблеми, пов’язані з її інтеграцією з системами, які не є частиною екосистеми Amazon Web Services. Незважаючи на гнучкість та відкритість платформи, інтеграція з продуктами або послугами, які функціонують поза AWS, може вимагати додаткових ресурсів та технічних зусиль. Це може стати перешкодою для компаній, що використовують різноманітні технології та платформи, та прагнуть до створення єдиної інтегрованої системи управління.

Отже, як і у випадку з Azure Digital Twins, використання платформи AWS IoT TwinMaker відзначається наявністю обмежень з точки зору універсальності та складності інтеграції, що може ускладнювати або унеможливлювати її використання для створення цифрових двійників мульсемедійних об’єктів.

1.2.3. Платформа NVIDIA Omniverse

Платформа NVIDIA Omniverse надає можливості для створення, редагування та спільної роботи в реальному часі над віртуальними моделями. Особливістю Omniverse є забезпечення високої продуктивності рендерингу, підтримка великої кількості форматів даних та можливість легкої інтеграції з іншими програмами та системами [41].

NVIDIA Omniverse забезпечує широкі можливості для дизайнерів, інженерів та художників, дозволяючи створювати складні інтерактивні сцени та моделі з високим рівнем деталізації. Завдяки інтуїтивно зрозумілому інтерфейсу та інструментам рендерингу, Omniverse надає нові можливості для креативної роботи [41]. Важливою особливістю NVIDIA Omniverse є її підтримка роботи в реальному часі та колаборативні функції, які дозволяють командам спільно дистанційно працювати над проєктами. Крім того, застосування алгоритмів машинного навчання розширює можливості платформи, роблячи її перспективним інструментом для різних областей використання [41].

Проте, застосування платформи NVIDIA Omniverse не є універсальним для всіх видів фізичних об'єктів. Ця платформа оптимально підходить для створення деталізованих візуальних моделей, що вимагають складного рендерингу та візуалізації, але може виявитися обмеженою при роботі з об'єктами, які потребують точного моделювання фізичних властивостей та взаємодій. Така специфіка робить Omniverse менш придатною для проєктів, що вимагають глибокого аналізу фізичної поведінки об'єктів або складних інженерних симуляцій. Отже, попри свої можливості у візуалізації, Omniverse має обмеження щодо створення цифрових двійників мульсемедійних об'єктів.

1.2.4. Платформа Supply Chain Twin

Supply Chain Twin є спеціалізованою платформою, призначеною для створення цифрових двійників для ланцюгів поставок. Ця платформа забезпечує унікальний підхід до візуалізації, моніторингу та аналізу ланцюгів поставок, дозволяючи компаніям оптимізувати свої операції за допомогою точного моделювання та симуляцій. Завдяки здатності інтегрувати різноманітні дані з різних джерел, Supply Chain Twin дозволяє керівникам ланцюгів поставок отримувати глибокий аналітичний огляд своїх операцій, виявляючи потенційні вузькі місця та можливості для поліпшення сервісу [42].

Особливістю Supply Chain Twin є її зосередженість на специфіці ланцюгів поставок, що включає в себе аналіз руху товарів, управління запасами, оптимізацію логістики та моніторинг показників продуктивності. Ця платформа дозволяє не тільки візуалізувати ланцюги поставок у вигляді деталізованих моделей, але й проводити різноманітні симуляції для передбачення наслідків змін у цих ланцюгах, у тому числі з використанням алгоритмів машинного навчання [42].

Загалом, оскільки платформа Supply Chain Twin є вузько спеціалізованою, вона не може бути застосована для створення цифрових двійників мульсемедійних об’єктів. Хоча вона надає значні переваги у визначеній області застосування – логістиці, її функціональність є обмеженою порівняно з більш універсальними платформами, такими як Azure Digital Twins, AWS IoT TwinMaker та NVIDIA Omniverse.

Таким чином, зазначені платформи не можуть бути ефективно використані для створення цифрових двійників мульсемедійних об’єктів, тому задача створення нових методів та програмних засобів для розширення галузі застосування технології цифрових двійників є актуальною.

1.3. Аналіз вимог до програмного забезпечення для створення цифрових двійників мульсемедійних об’єктів

Аналіз вимог до програмного забезпечення для створення цифрових двійників мульсемедійних об'єктів охоплює важливі аспекти, такі як визначення функціональних вимог, що задають ключові характеристики та можливості системи, та нефункціональні вимоги, які стосуються якості та ефективності системи, включаючи продуктивність, надійність, сумісність та безпеку [43-45].

Функціональні вимоги визначають конкретні функції або характеристики, які програмна система чи продукт повинні виконувати з метою задоволення певних потреб користувача або досягнення цілей проєкту. Ці вимоги звичайно описують дії чи поведінку системи у відповідь на певні вхідні дані або у визначених умовах, включаючи специфічні задачі, які система має виконати, та результати, які вона повинна надати. Функціональні вимоги є ключовими для проєктування системи, оскільки вони безпосередньо впливають на архітектуру, дизайн, тестування та валідацію продукту. Чітке формулювання функціональних вимог є вирішальним для успішної реалізації проєкту, оскільки вони встановлюють критерії для оцінки продуктивності та функціональності системи, а також забезпечують відповідність очікуванням користувачів [43].

Функціональні вимоги до програмного забезпечення для створення цифрових двійників мульсемедійних об'єктів наведено у табл. 1.2. Пріоритет вимоги визначено за шкалою: «Обов’язково» – «Бажано» – «Можливо».

Таблиця 1.2. Базові функціональні вимоги до мульсемедійного програмного забезпечення на основі технології цифрових двійників

Код	Вимога	Пріоритет
1	2	3
ФВ1	Забезпечення отримання файлових даних визначеної модальності зі сховища даних	Обов’язково
ФВ2	Забезпечення отримання файлових даних довільної модальності зі сховища даних	Бажано
ФВ3	Забезпечення отримання потокових даних визначеної модальності зі зовнішнього джерела (сенсора) визначеного типу	Обов’язково
ФВ4	Забезпечення отримання потокових даних довільної модальності зі зовнішнього джерела (сенсора) визначеного типу	Бажано
ФВ5	Забезпечення конвертування даних до визначеного формату	Обов’язково
ФВ6	Забезпечення визначення темпоральних характеристик набору даних довільної модальності	Обов’язково

Табл. 1.2 (продовження)

1	2	3
ФВ7	Забезпечення синхронізації наборів даних різних модальностей	Обов’язково
ФВ8	Забезпечення консолідації синхронізованих мультимодальних даних	Обов’язково
ФВ9	Забезпечення підтримки визначених протоколів комунікації зі зовнішніми джерелами даних (сенсорами) та приймачами даних (актуаторами)	Обов’язково
ФВ10	Забезпечення підтримки довільних протоколів комунікації зі зовнішніми джерелами даних (сенсорами) та приймачами даних (актуаторами)	Бажано
ФВ11	Забезпечення реалістичного візуального представлення мульсемедійного об’єкта	Обов’язково
ФВ12	Забезпечення моделювання визначених характеристик мульсемедійного об’єкта	Обов’язково
ФВ13	Забезпечення моделювання довільних характеристик мульсемедійного об’єкта	Бажано

ФВ14	Забезпечення цільової обробки темпоральних мультимодальних даних про мульсемедійний об’єкт	Обов’язково
ФВ15	Забезпечення інтеграції та обміну даними з зовнішнім програмним забезпеченням	Можливо

Нефункціональні вимоги визначають критерії, за якими оцінюється якість роботи системи, але не її конкретні поведінкові характеристики. Вони охоплюють аспекти, такі як продуктивність, надійність, масштабованість, безпека, сумісність, і зручність користування. Нефункціональні вимоги часто визначаються як обмеження або стандарти, яким має відповідати система. Вони є важливими, оскільки безпосередньо впливають на користувацький досвід та загальну задоволеність користувачів, а також на можливість системи задовольнити специфічні умови використання [44].

Нефункціональні вимоги до програмного забезпечення для створення цифрових двійників мульсемедійних об'єктів наведено у табл. 1.3. Пріоритет вимоги визначено за шкалою: «Обов’язково» – «Бажано» – «Можливо».

Таблиця 1.3. Базові нефункціональні вимоги до мульсемедійного програмного забезпечення на основі технології цифрових двійників

Код	Вимога	Пріоритет
НВ1	Забезпечення часу відгуку системи, оптимального для реалістичної візуалізації моделі мульсемедійного об’єкта	Обов’язково
НВ2	Забезпечення можливості обробки великих об'ємів даних	Обов’язково
НВ3	Забезпечення можливості масштабування залежно від обсягу оброблюваних даних	Бажано
НВ4	Забезпечення конфіденційності та захисту даних	Можливо
НВ5	Забезпечення точності моделювання	Обов’язково

	мульсемедійного об’єкта	
НВ6	Забезпечення реалізації інтуїтивно зрозумілого інтерфейсу користувача	Можливо
НВ7	Забезпечення сумісності з з іншими системами та додатками для інтеграції платформ цифрових двійників	Можливо

Сформульовані функціональні та нефункціональні вимоги мають бути враховані при розробленні прикладних програмних систем на основі концепції цифрових двійників мульсемедійних об’єктів.

1.4. Висновки до першого розділу

У цьому розділі проаналізовано ключові аспекти технології цифрових двійників та технології мульсемедіа та вивчено можливості розширення сфер їхнього прикладного застосування. Цифрові двійники, що є віртуальними відображеннями реальних об'єктів, дають змогу аналізувати параметри фізичних об'єктів, відстежувати їхній стан та моделювати різні сценарії взаємодії. Технологія мульсемедіа спрямована на створення комплексного досвіду взаємодії користувача з цифровим контентом із залученням тактильних, кіностетичних, термосептичних та інші видів відчуттів. Комбінування цих двох технологій у вигляді технології цифрових двійників мульсемедійних об’єктів надасть змогу розширення можливостей цифрової взаємодії користувачів з досліджуваними об’єктами, що створює умови для розроблення програмного забезпечення нового класу – мульсемедійних програмних систем, які призначені для застосування, у першу чергу, у галузі медицини та освіти.

У ході дослідження проаналізовано наявні платформи для створення цифрових двійників, такі як Azure Digital Twins, AWS IoT TwinMaker,

NVIDIA Omniverse та Supply Chain Twin. Кожна з цих платформ має свої специфічні особливості та галузі застосування, проте жодна з них не відповідає в повному обсязі потребам щодо створення цифрових двійників мульсемедійних об'єктів.

У результаті проведеного дослідження сформульовано функціональні та нефункціональні вимоги до програмного забезпечення на основі концепції цифрових двійників мульсемедійних об'єктів.

РОЗДІЛ 2. РОЗРОБЛЕННЯ МЕТОДУ КОНСОЛІДАЦІЇ МУЛЬСЕМЕДІЙНИХ ДАНИХ

Консолідація темпоральних мультимодальних даних про об’єкт спостереження, що є мульсемедійним об’єктом, є ключовою задачею, що вирішується у процесі функціонування мульсемедійної програмної системи.

Визначення 2.1. *Мульсемедійна програмна система* – це програмне забезпечення, яке оброблює темпоральні мультимодальні дані, що характеризують стан мульсемедійного об’єкта.

У процесі моніторингу мульсемедійного об’єкта генеруються значні обсяги даних. Ці дані отримуються з набору сенсорів та мають різні формати. Семантично ці дані відображають інформацію різної модальності, оскільки мульсемедійний об’єкт визначається різнобічно (мультимодально) з метою формування його комплексного цифрового опису. Важливою властивістю інформації про мульсемедійний об’єкт, окрім її *мультимодальності*, є *темпоральність* цієї інформації, тобто прив’язка до моменту реєстрації стану об’єкта. У наслідок цього виникає потреба в *одночасному* використанні даних *різних модальностей*. Це вимагає специфічних методів обробки даних для забезпечення їхньої консолідації. Поєднання інформації з різних джерел може значно поліпшити якісь аналізу даних, виявляючи складні залежності між даними та забезпечуючи більш глибоке занурення у явище чи об’єкт, що вивчається. Проте традиційні підходи до обробки мультимодальних даних часто є неефективними для мульсемедійних даних через інтермодальні розбіжності, тобто відмінності у представленні інформації різних модальностей.

Основна проблема при обробці мульсемедійних даних полягає в тому, що інформація з різних модальностей може мати різний характер,

структуру та семантику. Вирішення цієї проблеми вимагає розроблення нових підходів до консолідації таких даних з урахуванням їхніх специфічних особливостей.

У цьому розділі висвітлюється розроблення методу консолідації мульсемедійних даних на основі концепції мультиобразу, а також формування уніфікованої структури даних для представлення мультимодальної інформації, що буде консолідована та синхронізована у часі на єдиному часовому проміжку, який визначається життєвим циклом мульсемедійного об’єкта.

2.1. Концепція мультиобразу як основа методу консолідації мульсемедійних даних

Концепція мультиобразу [46] передбачає комплексне подання у цифровому вигляді інформації про об’єкт спостереження, яка реєструється у декількох модальностях та надходить з різних джерел (сенсорів). Це дозволяє синхронізувати та інтегрувати різні види сенсорних даних у цілісне представлення. Щоб досягти такої консолідації, важливо враховувати специфіку кожної модальності, включаючи різницю у підходах до отримання, обробки та збереження даних.

Основним підходом концепції мультиобразу є забезпечення семантичного взаємозв’язку між різними модальностями, що дозволяє користувачеві отримувати більш цілісний досвід від взаємодії з цифровим контентом. Наприклад, при використанні імерсійного середовища користувач може не лише чути та бачити події на екрані, але й відчувати запахи, температуру, дотики, що відтворюються за допомогою спеціального обладнання [47].

Ключовими принципами концепції мультиобразу є наступні.

1. *Інтегрованість*. Полягає у поєднанні різних типів сенсорної інформації, включаючи візуальну, аудіальну, тактильну, смакову тощо.
2. *Адаптивність*. Полягає у здатності системи динамічно змінювати та оптимізовувати подання інформації залежно від потреб користувача та контексту використання.
3. *Консистентність*. Полягає у забезпеченні логічності та цілісності подання інформації незалежно від модальності інформації.

Як математична концепція мультиобраз є непустим агрегатом, компонентами якого є кортежі значень темпоральних мультимодальних даних [48]. Семантика мультиобразу у контексті цього дослідження ґрунтується на тому, що ці значення темпоральних мультимодальних даних визначають властивості об'єкта спостереження.

Будь-який фізичний об’єкт виявляє свою сутність через набір властивостей, які можуть бути виміряні. У багатьох випадках ці властивості є взаємопов’язаними, адже вони подають стан одного і того ж об’єкту та є різними проявами його поведінки, що змінюється у часі. Комплексне подання даних, що відображають характеристики досліджуваного об'єкта, сприяє кращому розумінню поведінки цього об’єкта [49].

Основною задачею при застосуванні концепції мультиобразу для створення цифрового двійника мульсемедійного об’єкта є розроблення методу консолідації мульсемедійних даних, який забезпечуватиме гомогенність та семантичну зв’язність сукупності цих темпоральних мультимодальних даних. Тому, незалежно від природи таких даних та їх відмінностей між собою, вони повинні бути приведені до єдиної форми представлення, а також бути логічно пов’язані між собою, забезпечуючи зрозумілість та цілісність інформаційного контенту.

2.2. Теоретичні засади процесу консолідації мульсемедійних даних

Для забезпечення консолідації даних необхідно вдосконалити наявний математичний та логічний апарат, що застосовується у рамках концепції мультиобразу, для забезпечення вищезазначених принципів.

Основою дослідження, представленого в цьому розділі, є алгебраїчна система агрегатів (АСА), носієм якої є довільний набір певних структур – агрегатів [48].

Визначення 2.2. *Агрегат D* – це кортеж довільних кортежів, елементи якого належать до визначених множин:

$$D = [\![M_j | \langle d_i^j \rangle_{i=1}^{n_j}]\!]_{j=1}^{N} = [\![\{D\} | \langle D \rangle]\!], \quad (2.1)$$

де $\{D\}$ – це кортеж множин Mj, $\langle D \rangle$ – кортеж елементів $\langle d_i^j \rangle_{i=1}^{n_j}$, що відповідає кортежу множин $(d_i^j \in M_j)$.

У більш широкому сенсі агрегат можна розглядати як складну структуру даних для консолідованого представлення темпоральних мультимодальних наборів даних, які визначають той самий об'єкт спостереження.

Як алгебраїчна система АСА складається з трьох множин: носій (непорожня множина), множина операцій і множина відношень. Це дослідження зосереджено на застосуванні відношень АСА, зокрема, відношень між дискретними інтервалами.

2.2.1. Відношення між дискретними інтервалами

Визначення 2.3. *Дискретний інтервал* (ДІ) – це кортеж, елементи якого є унікальними значеннями, розташованими у порядку зростання або

спадання [50].

У прикладному сенсі ДІ є кортежом значень, які визначають моменти часу, коли необхідно виміряти характеристики об'єкта спостереження.

В АСА визначені наступні відношення між двома ДІ [53].

Відношення *передує* означає, що перший ДІ ($\overline{t^1}$) закінчується до початку другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \leftarrow \overline{t^2} \; if \; t^1_{n_1} < t^2_1. \tag{2.2}$$

Відношення *настає після* означає що перший ДІ ($\overline{t^1}$) починається після закінчення другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \rightarrow \overline{t^2} \; if \; t^1_1 > t^2_{n_2}. \tag{2.3}$$

Відношення *збігається* означає, що два ДІ ($\overline{t^1}$ і $\overline{t^2}$) починаються та закінчуються одночасно:

$$\overline{t^1} \leftrightarrow \overline{t^2} \; if \; t^1_1 = t^2_1 \; and \; t^1_{n_1} = t^2_{n_2} \; and \; n_1 = n_2. \tag{2.4}$$

Відношення *стикається з початком* означає, що перший ДІ ($\overline{t^1}$) завершується в той самий момент часу, в який другий ДІ ($\overline{t^2}$) починається:

$$\overline{t^1} \leftarrow\!\!| \; \overline{t^2} \; if \; t^1_{n_1} = t^2_1. \tag{2.5}$$

Відношення *стикається з кінцем* означає, що перший ДІ ($\overline{t^1}$) починається в той самий момент часу, коли другий ДІ ($\overline{t^2}$) завершений:

$$\overline{t^1} \mapsto \overline{t^2} \; if \; t^2_{n_2} = t^1_1. \tag{2.6}$$

Відношення *перекриває* означає, що другий ДІ ($\overline{t^2}$) починається під час першого ДІ ($\overline{t^1}$) і звершується після завершення першого ДІ:

$$\overline{t^1} \hookleftarrow \overline{t^2} \; if \; t^1_1 < t^2_1 \; and \; t^1_{n_1} < t^2_{n_2} \; and \; t^2_1 < t^1_{n_1}. \tag{2.7}$$

Відношення *перекривається* означає, що перший ДІ ($\overline{t^1}$) починається під час другого ДІ ($\overline{t^2}$) і завершується після завершення другого ДІ:

$$\overline{t^1} \hookrightarrow \overline{t^2} \; if \; t^2_1 < t^1_1 \; and \; t^2_{n_2} < t^1_{n_1} \; and \; t^1_1 < t^2_{n_2}. \tag{2.8}$$

Відношення *відбувається під час* означає, що перший ДІ ($\overline{t^1}$) починається після початку другого ДІ ($\overline{t^2}$) і завершується до завершення другого ДІ:

$$\overline{t^1} \curvearrowleft \overline{t^2} \; if \; t^1_1 > t^2_1 \; and \; t^1_{n_1} < t^2_{n_2}. \tag{2.9}$$

Відношення *містить* означає, що перший ДІ ($\overline{t^1}$) починається до початку другого ДІ ($\overline{t^2}$) та завершується після завершення другого ДІ:

$$\overline{t^1} \sim \overline{t^2} \; if \; t^1_1 < t^2_1 \; and \; t^1_{n_1} > t^2_{n_2}. \tag{2.10}$$

Відношення *починає* означає, що перший ДІ ($\overline{t^1}$) починається в той самий момент часу, що й другий ДІ ($\overline{t^2}$) і він завершується до завершення другого ДІ:

$$\overline{t^1} \leftarrow \overline{t^2} \; if \; t^1_1 = t^2_1 \; and \; t^1_{n_1} < t^2_{n_2}. \tag{2.11}$$

Відношення *починається* означає, що перший ДІ ($\overline{t^1}$) починається в той самий момент часу, що й другий ДІ ($\overline{t^2}$) та завершується після завершення другого ДІ:

$$\overline{t^1} \rightarrowtail \overline{t^2} \; if \; t_1^1 = t_1^2 \; and \; t_{n_1}^1 > t_{n_2}^2. \tag{2.12}$$

Відношення *закінчує* означає, що другий ДІ ($\overline{t^1}$) починається після початку другого ДІ ($\overline{t^2}$) і завершується в той самий момент часу, що й другий ДІ:

$$\overline{t^1} \hookleftarrow \overline{t^2} \; if \; t_1^1 > t_1^2 \; and \; t_{n_1}^1 = t_{n_2}^2. \tag{2.13}$$

Відношення *закінчується* означає, що перший ДІ ($\overline{t^1}$) починається перед початком другого ДІ ($\overline{t^2}$) та завершується в той самий момент часу, що й другий ДІ:

$$\overline{t^1} \twoheadrightarrow \overline{t^2} \; if \; t_1^1 < t_1^2 \; and \; t_{n_1}^1 = t_{n_2}^2. \tag{2.14}$$

Наведені вище відношення є якісними. Проте, якісні відношення не дають змогу встановити чисельне відношення у часі подій, що визначаються дискретними інтервалами. Тому пропонується новий підхід, який ґрунтується на модифікації цих відношень, щоб зробити їх кількісними.

2.2.2. Кількісні відношення між дискретними інтервалами

У цьому дослідженні пропонуються наступні кількісні відношення,

що дають змогу точно (кількісно) визначити часовий взаємозв'язок між двома дискретними інтервалами [50].

Відношення *кількісно передує* означає, що перший ДІ ($\overline{t^1}$) завершується за τ моментів часу до початку другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \overleftarrow{\tau} \overline{t^2} \; if \; t_1^2 = t_{n_1}^1 + \; \tau. \tag{2.15}$$

Відношення *кількісно настає після* означає, що перший ДІ ($\overline{t^1}$) починається через τ моментів часу після завершення другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \vec{\tau} \overline{t^2} \; if \; t_1^1 = t_{n_2}^2 + \; \tau. \tag{2.16}$$

Відношення *кількісно перекриває* означає, що другий ДІ ($\overline{t^2}$) починається через τ_s моментів часу після початку першого ДІ ($\overline{t^1}$) і завершується через τ_f моментів часу після завершення першого ДІ ($\overline{t^1}$):

$$\overline{t^1} \underset{(\tau_s, \tau_f)}{\overleftarrow{\;\;}} \overline{t^2} \; if \; t_1^2 = t_1^1 + \; \tau_\mathrm{s} \; and \; t_{n_2}^2 = t_{n_1}^1 + \tau_f \; . \tag{2.17}$$

Відношення *кількісно перекривається* означає, що перший ДІ ($\overline{t^1}$) починається через τ_s моментів часу після початку другого ДІ ($\overline{t^2}$) та завершується після τ_f моментів часу після завершення другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \underset{(\tau_s, \tau_f)}{\overrightarrow{\;\;}} \overline{t^2} \; if \; t_1^1 = t_1^2 + \; \tau_\mathrm{s} \; and \; t_{n_1}^1 = t_{n_2}^2 + \tau_f. \tag{2.18}$$

Відношення *кількісно відбувається під час* означає, що перший ДІ ($\overline{t^1}$) починається через τ_s моментів часу після початку другого ДІ ($\overline{t^2}$) та

завершується за τ_f моментів часу до завершення другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \overset{\curvearrowleft}{(\tau_s, \tau_f)} \overline{t^2} \; if \; t_1^1 = t_1^2 + \; \tau_{\mathrm{s}} \; and \; t_{n_2}^2 = t_{n_1}^1 + \tau_f. \tag{2.19}$$

Відношення *кількісно містить* означає, що перший ДІ ($\overline{t^1}$) починається за τ_s моментів часу до початку другого ДІ ($\overline{t^2}$) та завершується після τ_f моментів часу після завершення другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \overset{\curvearrowright}{(\tau_s, \tau_f)} \overline{t^2} \; if \; t_1^2 = t_1^1 + \; \tau_{\mathrm{s}} \; and \; t_{n_1}^1 = t_{n_2}^2 + \tau_f. \tag{2.20}$$

Відношення *кількісно починає* означає, що перший ДІ ($\overline{t^1}$) в той самий момент часу, що і другий ДІ ($\overline{t^2}$), але завершується за τ моментів часу перед завершенням другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \overset{\leftarrowtail}{(\tau)} \overline{t^2} \; if \; t_1^1 = t_1^2 \; and \; t_{n_2}^2 = t_{n_1}^1 + \tau. \tag{2.21}$$

Відношення *кількісно починається* означає, що перший ДІ ($\overline{t^1}$) починається в той самий момент часу, що і другий ДІ ($\overline{t^2}$), але завершується через τ моментів часу після завершення другого ДІ ($\overline{t^2}$):

$$\overline{t^1} \overset{\rightarrowtail}{(\tau)} \overline{t^2} \; if \; t_1^1 = t_1^2 \; and \; t_{n_1}^1 = t_{n_2}^2 + \tau. \tag{2.22}$$

Відношення *кількісно закінчує* означає, що перший ДІ ($\overline{t^1}$) починається через τ моментів часу після початку другого ДІ ($\overline{t^2}$) та завершується в той самий момент, що і другий ДІ ($\overline{t^2}$):

$$\overline{t^1} \underset{(\tau)}{\overset{\leftarrow\rho}{}} \overline{t^2} \; if \; t_1^1 = t_1^2 + \tau \; and \; t_{n_1}^1 = t_{n_2}^2. \tag{2.23}$$

Відношення *кількісно закінчується* означає, що перший ДІ ($\overline{t^1}$) починається за τ моментів часу до початку другого ДІ ($\overline{t^2}$) і завершується в той самий момент, що і другий ДІ ($\overline{t^2}$):

$$\overline{t^1} \underset{(\tau)}{\overset{\rightsquigarrow}{}} \overline{t^2} \; if \; t_1^2 = t_1^1 + \tau \; and \; t_{n_1}^1 = t_{n_2}^2. \tag{2.24}$$

Відношення *збігається, стикається з початком* та *стикається з кінцем* не мають якісних і кількісних варіантів, оскільки ці відношення завжди визначаються конкретними моментами часу і, таким чином, можуть розглядатися як якісні і як кількісні залежно від контексту.

Запропоновані кількісні відношення між дискретними інтервалами можуть бути корисними для формальної специфікації складної структури даних на основі концепції мультиобразу [48].

Визначення 2.4. *Мультиобраз* – це комплексне подання темпоральних мультимодальних наборів даних, що описують об’єкт.

У математичному сенсі мультиобраз (МО) є агрегатом, перший кортеж якого є непорожнім кортежом значень часу.

$$MI = \left[\!\left[T, M_1, .., M_N \mid \langle t_k \rangle_{k=1}^{\tau}, \left\langle d_{k_1}^1 \right\rangle_{k_1=1}^{n_1}, .., \left\langle d_{k_N}^1 \right\rangle_{k_N=1}^{N} \right]\!\right], \tag{2.25}$$

де T це набір часових значень; $\tau \geq n_i, i \in [1, \dots, N]$.

Таким чином, T є математичним поданням загальної шкали часу, що визначає весь період спостереження за об’єктом. Проте, для розроблення алгоритму обробки мультиобразу об’єкта може бути корисним представити

наочним способом взаємозв'язок різних модальностей даних у різних проміжках часу. Математичне визначення мультиобразу (2.25) не забезпечує такого представлення. Скористаємося кількісними відношеннями, щоб розширити визначення мультиобразу.

Спочатку необхідно визначити суб-мультиобраз (SMI_i), що представляє темпоральні модальності даних як:

$$SMI_i = \left[\!\left[T_i, M_i \vee \left\langle t_{k_i}^i \right\rangle_{k_i=1}^{\tau_i}, \left\langle d_{k_i}^i \right\rangle_{k_i=1}^{n_i} \right]\!\right], \tag{2.26}$$

де T_i – набір значень часу, що визначають моменти часу $\left\langle t_{k_i}^i \right\rangle_{k_i=1}^{\tau_i}$ коли дані $\left\langle d_{k_i}^i \right\rangle_{k_i=1}^{n_i}$ модальностей мають бути виміряні (модальність визначена набором M_i); $i \in [1, \dots, N]$.

Тоді, мультиобраз (*MI)* може бути визначений як:

$$MI = SMI_{i_1}\left(R_{i_1 i_2}\right) SMI_{i_2}, i_1 \neq i_2, \forall i_1, i_2 \in [1, \dots, N], \tag{2.27}$$

де $R_{i_1 i_2}$ – кількісне відношення між дискретними інтервалами $\left\langle t_{k_i}^{i_1} \right\rangle_{k_i=1}^{\tau_{i_1}}$ і $\left\langle t_{k_i}^{i_2} \right\rangle_{k_i=1}^{\tau_{i_2}}$ суб-мультиобразів SMI_{i_1} та SMI_{i_2} відповідно.

Інтерпретація концепції мультиобразу, визначеного формулою (2.27), дає змогу встановити темпоральні зв'язки між наборами даних різних модальностей. У свою чергу, це може надати можливість спростити формальну специфікацію складної структури даних для опису цифрового двійника мульсемедійного об'єкта.

2.2.3. Темпоральна специфікація мультиобразу

Візуалізація формальної специфікації на основі модифікованого визначення мультиобразу може бути представлена як орієнтований граф. Цю специфікацію називатимемо *темпоральною специфікацією мультиобразу*. Наступний приклад демонструє, як можна визначити темпоральну специфікацію.

Нехай задача полягає у створені освітнього фільму з мульсемедійним контентом для його відтворення в імерсійному навчальному середовищі. Мульсемедійним об'єктом у цьому фільмі є реальна природна сцена, у якій передбачене комплексне спостереження з метою вивчення флори та фауни. Спостереження виконується за допомогою двох відеокамер для фіксації візуальних характеристик середовища, двох мікрофонів для фіксації звуків природи та чотирьох спеціалізованих тактильних сенсорів, що надає змогу відчувати фізичні характеристики об'єктів, на яких вони знаходяться; дані, які вони реєструють можуть бути відтворені за допомогою спеціалізованого апаратного забезпечення, яке дозволить користувачеві віддалено відчути відповідні тактильні відчуття.

Згідно із запропонованим підходом, зазначений мульсемедійний об'єкт має бути описаний з використанням темпоральної специфікації з метою створення відповідного мульсемедійного продукту – мульсемедійного освітнього фільму.

Визначимо наступні суб-мультиобрази, що будуть використані для специфікації мульсемедійного об'єкту: SMI_1 – перше відео; SMI_2 – друге відео; SMI_3 – перше аудіо; SMI_4 – друге аудіо; SMI_5 – перший тактильний сенсор; SMI_6 – другий тактильний сенсор; SMI_7 – третій тактильний сенсор; і SMI_8 – четвертий тактильний сенсор. Кожен із цих SMI має власний ДІ, що відображає, коли були отримані дані модальності цього SMI.

Час реєстрації даних відповідних модальностей показано на рис. 2.1.

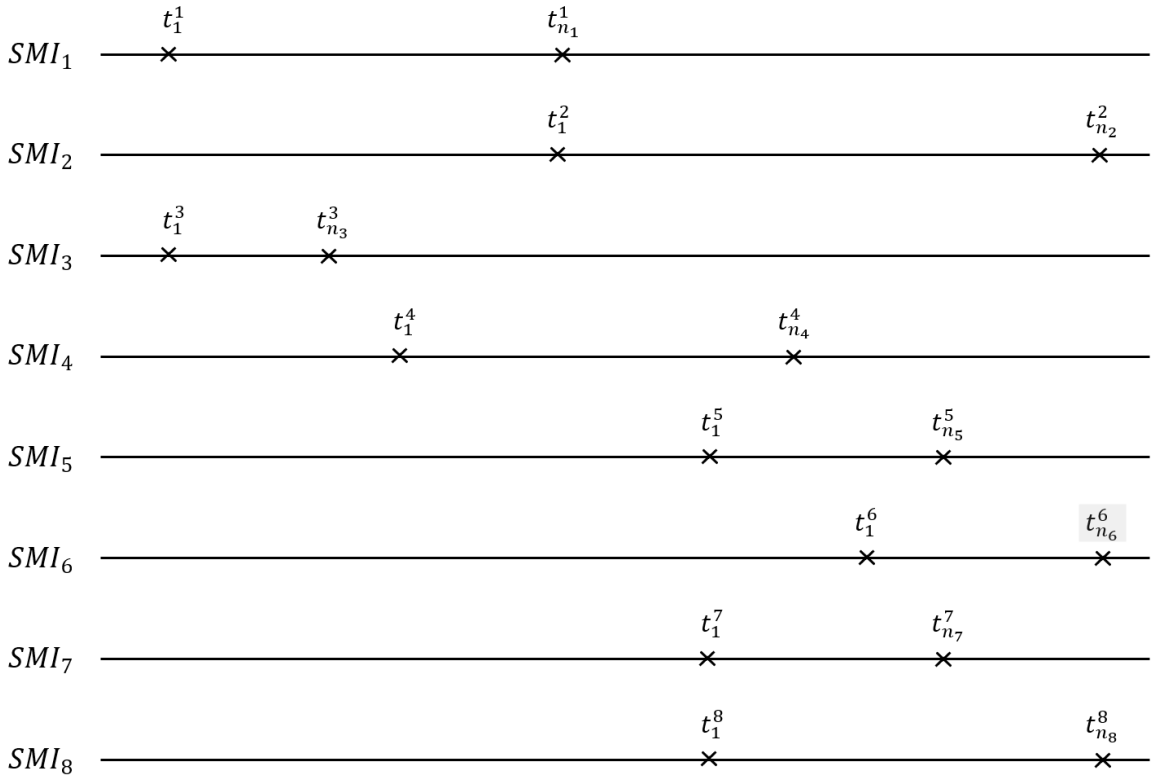

Рис. 2.1 – Схема хронометражу отримання даних модальностей

Ця схема синхронізації модальностей дозволяє розрахувати значення часу, які необхідні для визначення якісних зв’язків між ДІ, згідно із системою рівнянь (2.28).

$$\begin{cases} w_1 = t_{n_1}^1 - t_{n_3}^3 \\ w_2 = t_1^4 - t_{n_3}^3 \\ w_3 = t_1^4 - t_1^1 \\ w_4 = t_{n_4}^4 - t_{n_1}^1 \\ w_5 = t_1^5 - t_1^4 \\ w_6 = t_{n_5}^5 - t_{n_4}^4 \\ w_7 = t_1^5 - t_1^2 \\ w_8 = t_{n_2}^2 - t_{n_5}^5 \\ w_9 = t_1^6 - t_1^5 \\ w_{10} = t_{n_6}^6 - t_{n_5}^5 \\ w_{11} = t_1^6 - t_1^2 \\ w_{12} = t_{n_8}^8 - t_{n_7}^7 \\ w_{13} = t_{n_8}^8 - t_{n_5}^5 \end{cases} \quad (2.28)$$

Тоді мультиобраз для наведеного прикладу можна визначити темпоральною специфікацією, зображеною на рис. 2.2.

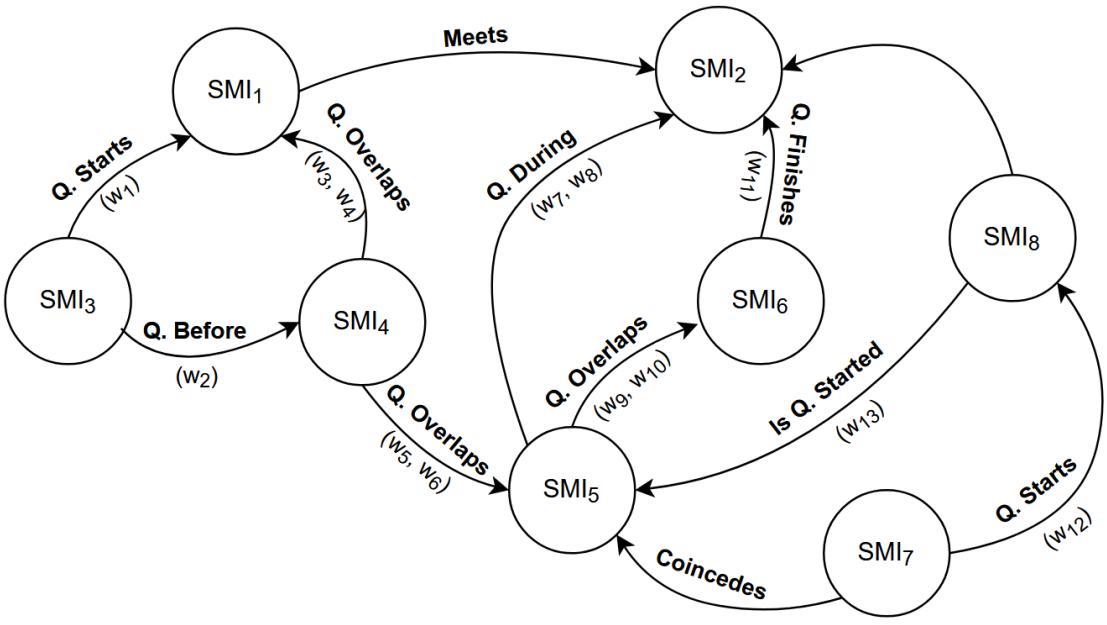

Рис. 2.2 – Темпоральна специфікація мультиобразу

(«Q» означає кількісний)

Наступним кроком є реалізація отриманої темпоральної специфікації у програмному коді. Це можна зробити за допомогою мови програмування загального призначення або спеціалізованої мови програмування ASAMPL 2.0 [51]. Перевага використання ASAMPL 2.0 полягає у тому, що ця мова програмування оптимізована для обробки темпоральних мультимодальних даних. Для забезпечення подальшого спрощення реалізації алгоритмів темпоральної мультимодальної обробки даних для розроблення мульсемедійного програмного забезпечення, запропоновані вище кількісні відношення між дискретними інтервалами мають бути реалізовані у синтаксисі мови програмування ASAMPL 2.0.

2.3. Основні етапи процесу консолідації мульсемедійних даних

Аналіз відомих методів обробки та синхронізації мульсемедійних даних [52, 53] дає змогу сформулювати вимоги, які має допомогти забезпечити метод консолідації таких даних:

- можливість використання даних з локальних баз даних, хмарних сховищ даних або мережевих потоків даних;
- можливість обробки даних різних модальностей;
- формування єдиної часової шкали для мультимодальних даних, що консолідуються;
- використання особливостей метаданих файлів мультимодальних даних для кращої синхронізації.

Програмна реалізація такого методу консолідації мульсемедійних даних має передбачати:

- багатопоточність (обробка даних кожної модальності в окремому потоці);
- можливість використання мультимодальних даних, що зберігаються у хмарних сховищах.

Метод консолідації мульсемедійних даних, що розроблений у цьому дослідженні, складається з таких основних етапів.

1. Визначення темпоральної специфікації мультиобразу об’єкта.
2. Отримання та форматування вхідних файлів.
3. Аналіз метаданих файлів даних різних модальностей.
4. Формування часової шкали процесу дослідження об’єкта спостереження.
5. Визначення структури даних та створення файлу консолідованих мульсемедійних даних.
6. Кодування даних різних модальностей у вихідному мульсемедійному файлі з використанням стеганографічних методів.

2.3.1. Визначення темпоральної специфікації мультиобразу об’єкта спостереження

Визначення темпоральної специфікації мультиобразу об’єкта спостереження є ключовим етапом у процесі створення цифрового двійника мульсемедійного об’єкту. Цей процес включає у себе серію кроків, які надають можливість встановити та структурувати темпоральні характеристики об’єкта, що є важливим для точного та ефективного подання мульсемедійних даних.

Основні кроки формування темпоральної специфікації є наступними.

1. *Ідентифікація об’єкта спостереження*. На цьому етапі визначається об’єкт спостереження, його ключові атрибути та характеристики. Важливо зібрати всю необхідну інформацію про об’єкт, яка впливає на його темпоральні властивості.
2. *Аналіз часових параметрів*. Проводиться детальний аналіз часових рамок, пов’язаних з життєвим циклом об’єкта. Це може включати історичні дані, заплановані події, циклічні процеси тощо. Важливо визначити всі відповідні часові моменти та періоди, які мають значення для функціонування об’єкта спостереження.
3. *Встановлення темпоральних взаємозв’язків*. На цьому етапі розробляється структура темпоральних взаємозв’язків між різними елементами чи подіями, пов’язаними з об’єктом. Для формалізації цих взаємозв’язків використовуються теоретичні засади алгебраїчної системи агрегатів.
4. *Створення темпоральної моделі*. Розробляється темпоральна модель об’єкта, що включає всі зібрані дані та встановлені взаємозв’язки. Модель має враховувати можливі зміни та доповнення у майбутньому.

Темпоральна специфікація мультиобразу об’єкта спостереження є фундаментальним етапом, який забезпечує точне та ефективне подання мульсемедійного контенту. Від її якості залежить якість реалізації цифрового двійника мульсемедійного об’єкта.

2.3.2. Збереження, форматування та аналіз метаданих

Мульсемедійні дані, які потрібно консолідувати, можуть бути збережені як локально, так і у хмарних сховищах. Також вони можуть бути збережені у різних форматах, навіть у межах однієї модальності, або передаватися як мережевий потік даних.

Після отримання вхідних даних та файлів, у тому числі, з можливістю збереження локально у файл потокових даних різних модальностей, потрібно звести наявні файли до єдиного переліку форматів. Пропонується використати наступні формати для збереження даних відповідних модальностей: MP4 контейнер – для збереження відеофайлів, MP3 – для збереження аудіофайлів, CSV – для збереження текстових або/та числових значень, що мають часові мітки.

Коли всі дані отримано, збережено та форматовано до визначених вище форматів, розпочинається етап аналізу даних для виявлення темпоральної інформації, необхідної для консолідації та синхронізації наявних мульсемедійних даних.

Окрім власне даних про об’єкт спостереження, які потрібно синхронізувати, вхідні дані, що організовані у вигляді файлу, включають метадані, а саме: тип файлу, його розмір, дата створення, дата останнього редагування, місцезнаходження у момент створення, технічна інформація про пристрій, яким файл було створено. Для відео та аудіо файлів до складу метаданих також входять: тривалість, бітрейт, кодек, кількість

каналів, розмір кадру, тощо. Додатковими метаданими є назва файлу, його опис, інформація про авторів тощо.

Для коректної консолідації мульсемедійних даних необхідно виявити метадані, які називатимемо *темпоральними метаданими*. Такими метаданими є: дата створення файлу, дата його останнього редагування та тривалість запису. Аналіз темпоральних метаданих дає змогу отримати інформацію, яка потрібна для подальшої обробки даних, а саме, для їх сортування, поєднання та синхронізації з метою комплексного подання мульсемедійних даних [56], що всебічно описують об’єкт спостереження.

Наприклад, якщо файли з даними різних модальностей створені у приблизно однаковий момент часу, то аналіз темпоральних метаданих дозволяє виявити, які дані почали надходити першими та визначити тривалість їхнього надходження, що дає змогу виконати синхронізацію цих даних з даними інших модальностей в одному часовому проміжку.

У файлах різного типу темпоральні метадані можуть зберігатися у різних форматах, а також бути відсутніми. Крім того, мітки часу можуть зберігатись не у заголовку файлу, а в основній частині файлу, тобто не у метаданих, а у даних. Тому задача виявлення темпоральних метаданих не є тривіальною.

Першим етапом вирішення цієї задачі є аналіз структури файлу. Залежно від типу файлу, часові мітки можуть зберігатися в різних місцях і форматах. Наприклад, для мультимедійних файлів, таких як відео або аудіо, часові мітки часто інтегровані безпосередньо в потік даних і можуть бути представлені у формі часових кодів. У випадку документів, таких як логи подій або XML-файли анотацій, часові мітки можуть бути включені як частина тексту або атрибутів тегів. Тому для початку аналізу потрібно проаналізувати формат файлу і визначити, де і в якому вигляді можуть бути збережені часові мітки.

Другий етап – це безпосередньо виявлення та екстракція часових міток. Це може вимагати застосування спеціалізованих скриптів для читання та аналізу файлу. Наприклад, можуть бути використані інструменти для аналізу метаданих мультимедійних файлів або парсери XML для витягування даних з анотацій. Важливо забезпечити точність виявлення, оскільки неправильно інтерпретовані або відсутні часові мітки можуть спотворити загальну картину синхронізації даних.

Для прискорення процесу консолідації мульсемедійних даних доцільно забезпечити багатопотоковість алгоритму виявлення темпоральних даних для одночасної обробки файлів різних модальностей. Алгоритм багатопотокової обробки файлів для виявлення темпоральних метаданих наведений на рис. 2.3.

2.3.3. Формування часової шкали

Результатом виконання першого та другого етапу консолідації мульсемедійних даних є виявлення темпоральних метаданих – часових значень, що визначають тривалість та час початку і закінчення відтворення даних певної модальності. Проте ці часові значення можуть бути визначені у будь-яких одиницях виміру часу: мілісекундах, годинах, місяцях тощо, що залежить від частоти отримання відповідних даних з сенсорів, камер та інших пристроїв. Для консолідації мульсемедійних даних необхідно привести всі значення в одну систему виміру часу та визначити єдиний спільний часовий проміжок для узгодженого представлення даних різних модальностей — часову шкалу життєвого циклу об’єкта спостереження. Ця шкала (рис. 2.4) формується на основі отриманого набору темпоральних метаданих, який є масивом часових міток, що позначають ключові моменти дослідження об’єкта спостереження та сприйняття людиною-дослідником мульсемедійної інформації про цей об’єкт.

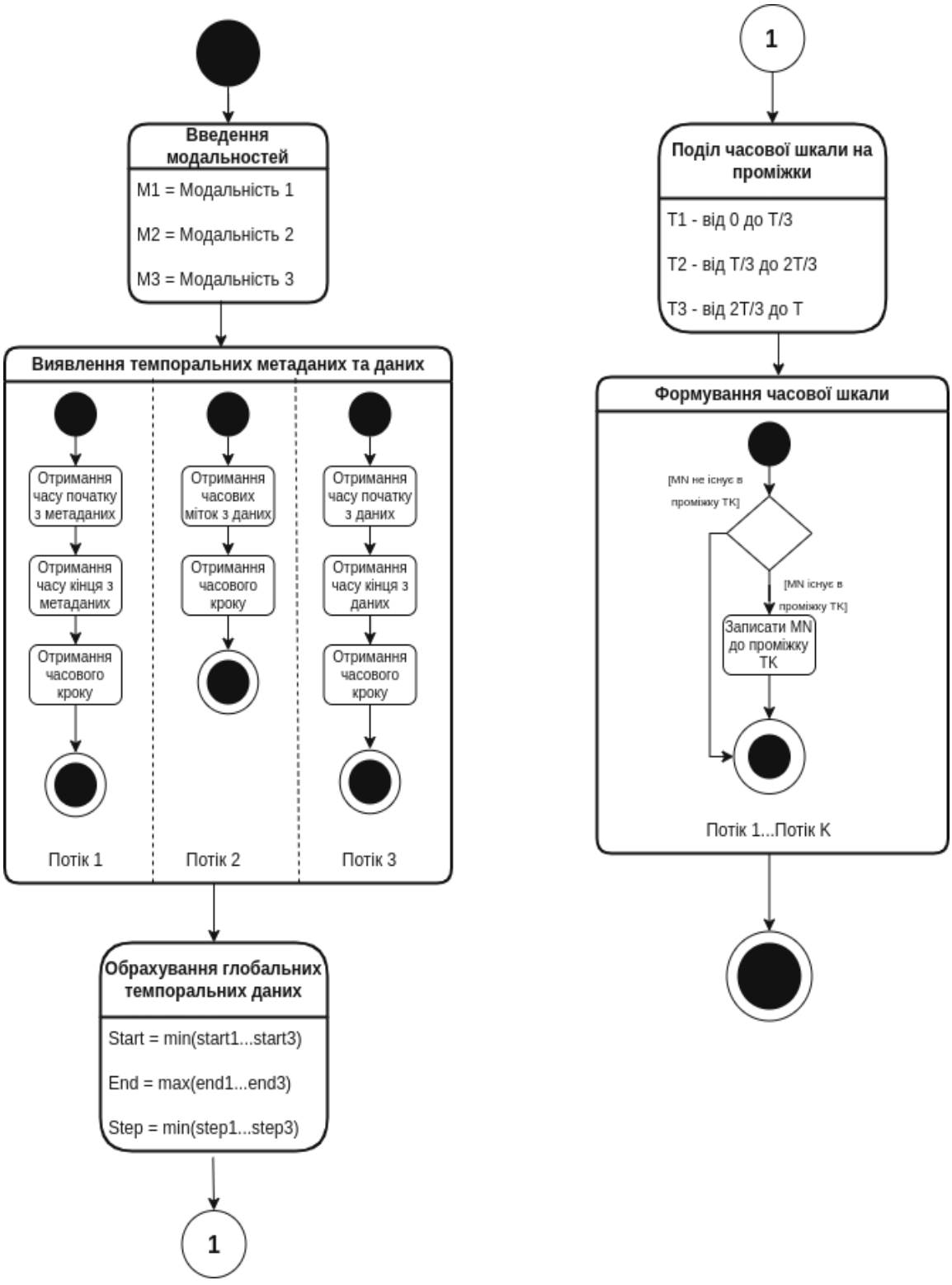

Рис. 2.3 – Алгоритм багатопотокової обробки мульсемедійних файлів для виявлення темпоральних метаданих

Рис. 2.4 – Приклад візуалізації часової шкали життєвого циклу об’єкта спостереження

Часова шкала як вектор значень часу визначається згідно з формулою (2.29).

$$\overline{T} = \left[\left(\left(i = 1 \middle| N\overline{t}_i\right) \uparrow_T\right) \|_T\right], \tag{2.29}$$

де $\overline{T}$ – часова шкала життєвого циклу об’єкта спостереження;

$\overline{t}_i$ – кортеж темпоральних значень, які визначають моменти часу, у які існує (має сенс для дослідника) інформація i-ї модальності;

$\uparrow_T$ – операція сортування [52] об’єднаного кортежу темпоральних значень для формування шкали T;

$\|_T$ – операція проріджування [52] об’єднаного кортежу темпоральних значень для формування шкали T.

2.3.3. Визначення структури даних та створення файлу консолідованих мульсемедійних даних

Синхронізовані мульсемедійні дані мають бути збережені у вигляді файлу. Можливим форматом такого файлу може бути JSON (JavaScript Object Notation) [57], оскільки він надає змогу структуровано зберігати

мультимодальні дані. Формат JSON, завдяки своїй структурованості та інтегрованості, зарекомендував себе як універсальний стандарт у сферах передачі та зберігання інформації. Проте мульсемедійні дані є темпоральними, тому для забезпечення подальшої коректної обробки файлу таких даних формат файлу має бути модифікований з урахуванням темпоральності даних. Назвемо такий модифікований формат *TJSON* (*Timeline JSON*). Цей формат включає основні переваги JSON, такі як читабельність, інтеграційні можливості та гнучкість у структурованому представленні даних, але додатково розширюється засобами для роботи із темпоральними даними. Приклад TJSON-об’єкту наведено у Лістингу 2.1.

Лістинг 2.1 – Специфікація TJSON-об’єкта

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "object",
  "properties": {
    "name": {
      "type": "string"
    },
    "description": {
      "type": "string"
    },
    "duration": {
      "type": "number"
    },
    "mediaFormats": {
      "type": "array",
      "items": {
        "type": "string"
      }
    },
```

```
    "step": {
      "type": "number"
    },
    "stepMeasurement": {
      "type": "string",
      "enum": ["ns, Ojs, ms, s, min, h"]
    },
    "data": {
      "type": "object",
      "patternProperties": {
        "^[0-9]+(\\.[0-9]+)?$": {
          "type": "object",
          "properties": {
            "modalityData": {
              "type": "string"
            }
          },
          "additionalProperties": false
        }
      },
      "additionalProperties": false
    }
  },
  "required": ["name", "description", "duration", "mediaFormats",
"step", "stepMeasurement", "data"],
  "additionalProperties": false
}
```

Використання запропонованого формату дозволяє створити деталізовану темпоральну структуру, що дає змогу синхронізувати дані різних модальностей у межах єдиної часової шкали. Також відкритість формату JSON, на якому ґрунтується запропонований формат, дозволяє

спростити інтеграцію з існуючими інструментами, що можуть бути реалізовані за допомогою різних мов програмування.

Після визначення TJSON-об’єкту, тобто структури даних для збереження мульсемедійних даних, відбувається створення мульсемедійного файлу. Пропонується завжди використовувати як тип файлу, що створюється, *mp4* контейнер – навіть за відсутності відео- чи аудіоданих. Текстові, числові та спеціалізовані формати зберігаються у тілі TJSON-об’єкту та кодуються у вихідному *mp4* файлі за допомогою алгоритмів стеганографії.

Вихідний файл консолідованих мульсемедійних даних містить дані, впорядковані згідно з визначенням TJSON-об’єкта. Цей файл може бути відтворений за допомогою відповідного програмного та апаратного забезпечення [58].

Також мульсемедійні дані можуть бути попередньо оброблені відповідно до задачі дослідження. При цьому доцільно застосувати операції впорядкування або логічні операції на основі АСА [52], оскільки TJSON-об’єкт є програмною реалізацією агрегата – математичної абстракції, визначеної в АСА. Таким чином, консолідовані мульсемедійні дані можна представити у вигляді набору агрегатів, над якими виконуються логічні операції та операції впорядкування.

Якщо операції впорядкування можуть виконуватися у межах одного набору даних, то для логічних операцій наборів даних має бути два, тому логічні операції можуть виконуватися або між двома наборами із різних, можливо, сусідніх часових проміжків, або за наявності іншого такого TJSON-об’єкта. У разі, якщо необхідно виконати операції по всьому TJSON-об’єкту або між двома TJSON-об’єктами, можна застосувати багатопотоковий підхід для збільшення ефективності обчислень. При цьому для забезпечення багатопотоковості часова шкала поділяється на

фрагменти. Мульсемедійні дані, що визначені у певному фрагменті шкали, оброблюються в окремому потоці.

2.4. Кодування TJSON-об'єкта у вихідному файлі за допомогою алгоритмів стеганографії

Стеганографія – це техніка приховування інформації у певному носії для забезпечення того, щоб інформація могла бути передана призначеним одержувачам без її публічного відображення. На відміну від криптографії, де інформація захищається шляхом шифрування, стеганографія надає можливість приховувати інформацію в іншому носії інформації, наприклад, у зображенні, аудіо- чи відеофайлі, таким чином, що неможливо виявити її присутність [60]. Базова модель стеганографії наведена на рис. 2.5.

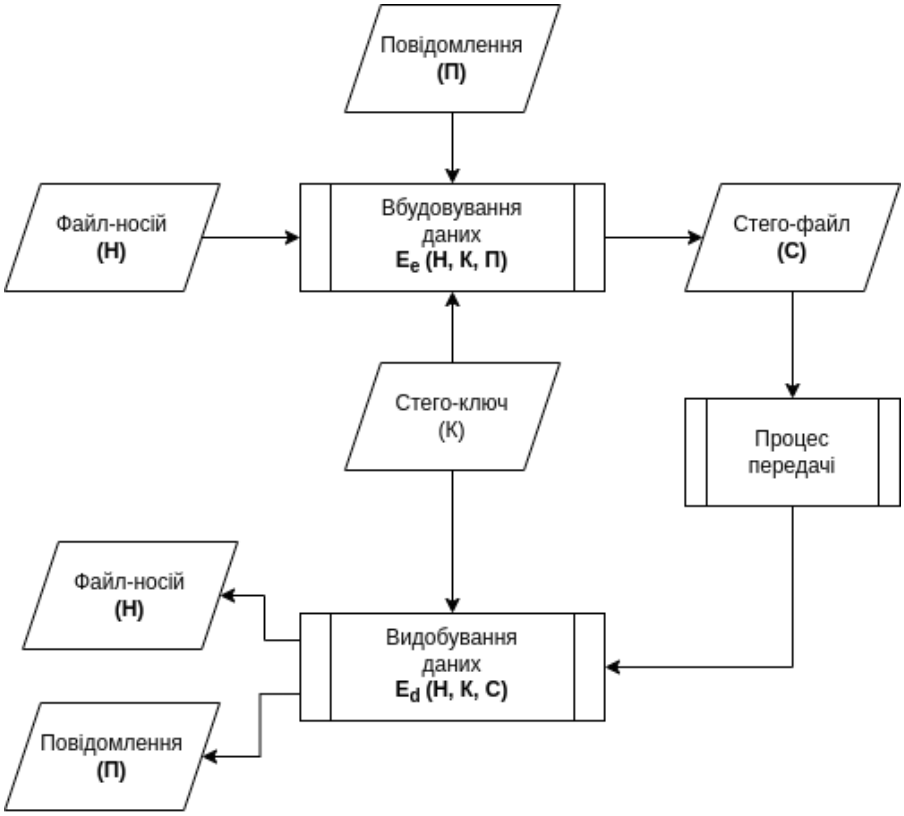

Рис 2.5 – Базова модель стеганографії

Приховування даних здійснюється шляхом застосування алгоритму вбудовування E_e повідомлення П до файлу-носія Н з використанням ключа К з метою ґенерування стегофайлу С. Після передавання стегофайлу С через канал зв'язку на приймальну сторону повідомлення П видобувається шляхом виконання алгоритму видобування даних E_d. Таким чином, процес вбудовування даних можна описати формулою (2.30).

$$E_e\,(Н, К, П) \rightarrow С, \qquad (2.30)$$

де E_e – алгоритм вбудовування, Н – файл-носій, К – ключ стеганографії, П – повідомлення, С – згенерований під час процесу вбудовування стегофайл, символ → означає власне сам процес ґенерування стегофайлу.

Процес видобування даних E_d, у свою чергу, можна визначити формулою (2.31).

$$E_d\,(Н, К, С) \approx П, \qquad (2.31)$$

де E_d – процес видобування, Н – файл-носій, К – ключ стеганографії, С – стегофайл, П – закодоване повідомлення.

Символ ≈ у формулі (2.31) показує, що при застосуванні деяких методів стеганографії функція видобування E_d може лише приблизно відновити оригінальне повідомлення П, оскільки вхідне повідомлення може зазнати певних спотворень у процесі стеганографічного вбудовування
[61-62].

Оскільки, у контексті застосування принципу стеганографії у методі консолідації мульсемедійних задача приховування даних відсутня, а лише є задача їхнього кодування з метою поєднання, то необхідно обрати серед

відомих алгоритмів такий, що дозволить закодовувати дані без спотворень для вбудовування у файл-носій, який теж не має зазнавати спотворень.

2.4.1. Вибір методу стеганографії для оптимізації способу збереження даних TJSON-об'єкту

Найбільш широко використовуваними методами стеганографії є метод LSB [63-64], метод DCT [66], метод DFT [63, 68] та метод DWT [70-71].

Метод LSB є одним з найпростіших і найпоширеніших методів стеганографії. Він полягає у приховуванні інформації у найменш значущих бітах пікселів зображення або інших мультимедійних даних. Наприклад, якщо зображення містить 8-бітні пікселі, то метод LSB може використовувати 3 найменш значущі біти для вбудовування даних [65].

Метод DCT ґрунтується на дискретному косинусному перетворенні. Він використовується для перетворення зображення або інших мультимедійних даних у дискретний спектр. Метод DCT можна використовувати для приховування інформації у надлишкових даних спектру [66-67].

Метод DFT ґрунтується на дискретному перетворенні Фур'є. Він подібний до методу DCT, але використовує дискретне перетворення Фур'є замість дискретного косинусного перетворення [68-69].

Метод DWT ґрунтується на дискретному вейвлет-перетворенні. Він подібний до методу DCT і DFT, але використовує вейвлет-аналіз [70-71].

Зазначені методи відрізняються між собою наступним:

- методом вбудовування даних;
- типами та кількістю підтримуваних носіїв;
- об'ємами інформації, що може бути прихована;
- ступенем можливого спотворення вихідного файлу;

- складністю реалізації;
- стійкістю до атак.

Серед перелічених критеріїв можна знехтувати стійкістю до атак, адже головна задача полягає не у приховуванні даних, а в їх поєднанні для утворення єдиного мульсемедійного файлу, що містить у собі дані різних модальностей.

Переваги та недоліки кожного із зазначених методів для застосування у методі консолідації мульсемедійних даних наведені у табл. 2.1.

Таблиця 2.1 – Порівняння методів стенографії

Метод	Спосіб вбудовування	Переваги	Недоліки
LSB	Модифікація найменш значущих бітів	- Простота реалізації, - сумісність з багатьма типами носіїв	- Можливе погіршення якості носія, - вбудовує невеликі обсяги даних, - чутливий до шуму
DCT	Переміщення частот	- Вбудовує більші обсяги даних	- Складність реалізації, - можливе погіршення якості носія
DFT	Переміщення частот	- Вбудовує ще більші обсяги даних	- Складність реалізації, - можливе погіршення якості носія
DWT	Переміщення частот	- Більш гнучкий, - вбудовує найбільший обсяг даних,	- Висока складність реалізації

		– надає можливість мінімізувати спотворення	

За результатами аналізу зазначених методів пропонується використовувати для створення мульсемедійного файлу метод DWT. Розглянемо його докладніше.

DWT – це математичний метод, що використовується для перетворення зображення із просторової області, де сигнал представлений у вигляді дискретних точок, у частотну, де сигнал представлений у вигляді набору частотних компонентів. Перетворення виконується шляхом розкладання зображення на набір вейвлет-коефіцієнтів. Кожен вейвлет-коефіцієнт є значенням, що вказує наскільки значно поточний вейвлет присутній у зображенні. Вейвлет-коефіцієнти можуть бути використані для ущільнення зображення шляхом видалення вейвлет-коефіцієнтів, які відповідають низькочастотним компонентам цього зображення [70-71].

Вейвлети – це математичні функції, що дають змогу аналізувати різні частотні компоненти даних. Графік функції вейвлета має вигляд хвилі або хвилеподібних коливань з амплітудою, що зменшується до нуля, віддаляючись від початку координат. Ці функції можуть бути використані для представлення зображень, які мають як локальні, так і глобальні частотні характеристики. Вейвлет-коефіцієнти, у свою чергу, визначаються інтегральним перетворенням сигналу. Отримані вейвлет-спектрограми відрізняються від звичайних спектрів Фур'є тим, що мають додаткову чітку прив'язку різних особливостей сигналів до часу.

Для обробки зображень вейвлети використовуються як базові функції. Ці функції отримують шляхом розширення та декодування материнського вейвлета $\psi\,(x)$ на величину s та τ, як визначено формулою (2.32).

$$\psi_{t,s}(x) = \psi\frac{x-\tau}{s}, \qquad (2.32)$$

де $\psi_{t,s}(x)$ – вейвлет-функція; $\psi\ (x)$ – материнський вейвлет; x – змінна, що вказує на час або простір у залежності від природи сигналу; τ – параметр зсуву, що вказує величину, на яку потрібно звинути материнський вейвлет $\psi\ (x)$ вздовж осі x; s – параметр масштабування, що вказує на ширину вейвлет-функції.

Перетворення такого роду використовуються при ущільненні зображень, наприклад, у форматі JPEG 2000, який відрізняється від класичного JPEG тим, що при тих же розмірах файлу завдяки вейвлет-перетворенням зображення має менше втрат та більш глибоку деталізацію, чіткість й гладкість. Також DWT використовується для зменшення шумів в аудіо- і відеофайлах [60].

Оскільки дискретне вейвлет-перетворення передбачає розкладення зображення на різні масштаби та частоти, це надає можливість закодовувати інформацію на різних рівнях деталізації зображення, обираючи ті рівні, де спотворення буде менше помітно або непомітно зовсім. Більш високі рівні деталізації містять більше інформації та менше впливають на загальний вигляд зображення, що дозволяє зберегти високу якість важливих областей зображення та вбудовувати інформацію до менш важливих. Дотримуючись певного балансу, можна вбудовувати досить великі обсяги інформації, при цьому не втрачаючи початкової якості та деталізації зображення, що задовольняє поставлену задачу у повному обсязі. Отже, у кожному кадрі вихідного mp4 файлу мають бути закодовані дані, які не входять до класу аудіо-візуальних даних. Також метод DWT завдяки своїй гнучкості надає змогу уникнути спотворення вихідного

зображення, а також зберігати та видобувати закодовану інформацію без жодних змін.

2.4.2. Процедура збереження даних TJSON-об'єкта на основі дискретного вейвлет-перетворення

Існує декілька варіантів дискретного вейвлет-перетворення, які відрізняються між собою підходом до обробки сигналу, що впливає на складність реалізації. Одним з найбільш широко вживаних вейвлет-перетворень є перетворення Гаара [72-73].

Дискретне вейвлет-перетворення Гаара (HDWT) є дискретним перетворенням, яке розкладає сигнал на два компоненти: коефіцієнти апроксимації та коефіцієнти деталізації. Вейвлет-функція Гаара визначається на інтервалі [0,1] формулою (2.33).

$$\begin{cases} 1, & \text{якщо } 0 \le t < 0.5, \\ -1, & \text{якщо } 0.5 \le t \le 1, \\ 0, & \text{в інших випадках.} \end{cases} \quad (2.33)$$

Обчислення коефіцієнтів апроксимації cA для скалярної декомпозиції сигналу $f(t)$ за допомогою вейвлета Гаара визначається формулою (2.34).

$$cA[k] = \frac{1}{2}\sum f\,(t)\phi(2t-k), \quad (2.34)$$

де $cA[k]$ – коефіцієнт апроксимації на рівні деталізації k;

$f(t)$ – вхідний сигнал або послідновність;

$\phi(t)$ – функція масштабування, яка для вейвлета Гаара визначається формулою (2.35).

$$\phi(t) = \begin{cases} 1, \text{якщо } 0 \le t < 1, \\ 0, \text{в інших випадках.} \end{cases} \quad (2.35)$$

Обчислення коефіцієнтів деталізації cD визначається формулою (2.36).

$$cD[k] = \frac{1}{2}\sum f\,(t)\psi(2t-k), \quad (2.36)$$

де $cD[k]$ – коефіцієнт деталізації на рівні деталізації k;

$\psi(t)$ – вейвлет-функція Гаара [72].

Для роботи із зображеннями використовують двовимірне дискретне вейвлет-перетворення Гаара, сутність якого полягає в обчисленні коефіцієнтів апроксимації та коефіцієнтів деталізації для кожного рядка по горизонталі та, при повторному застосуванні, по вертикалі. Таким чином отримується набір із чотирьох піддіапазонів зображення (рис. 2.6) [74-75], а саме:

- LL відображає нижчі частоти (апроксимацію) за обома вимірами;
- LH представляє нижчі частоти по горизонталі та вищі по вертикалі;
- HL представляє вищі частоти по горизонталі та нижчі по вертикалі;
- HH відображає вищі частоти за обома вимірами.

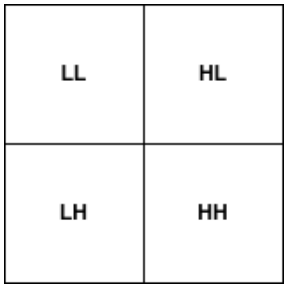

Рис 2.6 – Чотири піддіапазони зображення після застосування дискретного вейвлет-перетворення

Для двовимірного дискретного вейвлет-перетворення коефіцієнт апроксимації для пікселю (i, j) визначається за формулою (2.37).

$$cA[i,j] = \frac{1}{4}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} f\,[k,l]\phi(2k-i)\phi(2l-j). \qquad (2.37)$$

Коефіцієнти деталізації для вейвлет-функції Гаара можуть бути обчислені за допомогою формули (2.38).

$$cD[k] = \frac{1}{4}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} f\,[k,l]\psi(2k-i)\phi(2l-j), \qquad (2.38)$$

Тоді алгоритм вбудовування інформації у зображення за допомогою дискретного вейвлет-перетворення може бути сформульований наступним чином.

1. Прочитати зображення-носій.
2. Прочитати інформацію, яку необхідно вбудувати.
3. Виконати двійкове перетворення інформації для вбудовування.
4. Розбити зображення на канали.
5. Застосувати дискретне вейвлет-перетворення Гаара для отримання чотирьох піддіапазонів: LL, HL, LH і HH.
6. Вбудувати двійкові дані до піддіапазону HH, що містить найвищі частоти зображення для вбудовування інформації.
7. Виконати зворотнє дискретне вейвлет-перетворення для всіх піддіапазонів.
8. Створити стегозображення.

Приклад розкладання кадру на чотири піддіапазони навдено на рис. 2.7, де у нижньому правому куті знаходиться піддіапазон HH з найвищими частотами, до яких вбудовуватиметься інформація. Після

вбудовування, зображення відновлюється за допомогою зворотнього вейвлет-перетворення до свого початкового вигляду. Це стегозображення містить у собі закодовану інформацію, яку можна відновити шляхом повторного розкладання зображення на піддіпазони та отримання її з найвищих частот й перетворення з двійкового вигляду у потрібний формат.

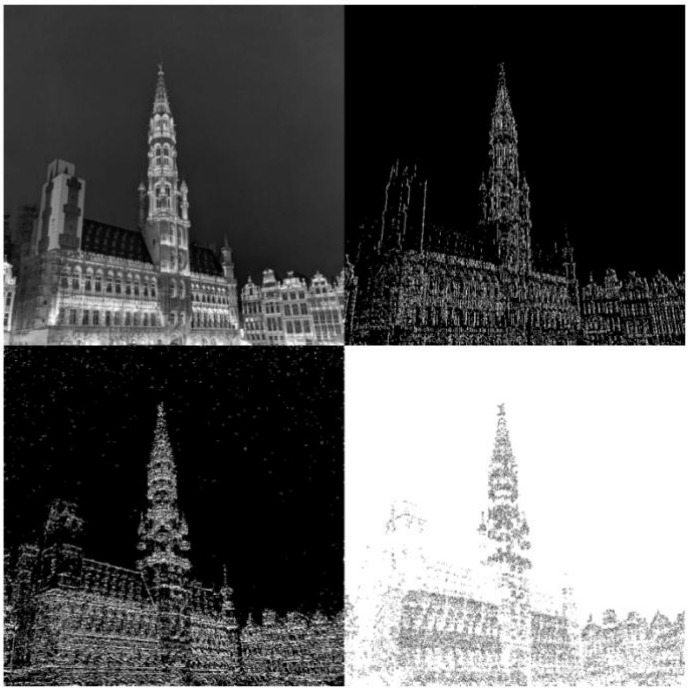

Рис. 2.7. Приклад розкладення кадру на чотири діапазони після дискретного вейвлет-перетворення Гаара

Тоді алгоритм отримання інформації зі стегозображення може бути сформульований наступним чином.

1. Прочитати стегозображення.
2. Розбити стегозображення на канали.
3. Отримати коефіцієнти області перетворення за допомогою дискретного вейвлет-перетворення Гаара для кожного із піддіапазонів.
4. Видобути двійкові дані із піддіапазону НН.

5. Сформувати та зберегти отриману інформацію у потрібному форматі.

Для того, щоб перевірити, що запропонований алгоритм повною мірою виконує поставлену задачу та результуюче стегозображення не зазнає значних спотворень у порівнянні з початковим зображенням-носієм, скористаємося метриками PSNR та SSIM [76-78].

Метрика PSNR визначається як відношення максимальної потужності сигналу до потужності шуму, що додає спотворення та вимірюється у децибелах згідно з формулою (2.39) [77].

$$PSNR = 10 \cdot log_{10}\left(\frac{MAX_I^2}{MSE}\right), \quad (2.39)$$

де $MAXI$ – максимально можливе значення пікселя на зображення,
MSE – середньоквадратичне відхилення між оригінальним зображенням та відтвореним, що визначається за формулою (2.40).

$$MSE = \frac{1}{M \times N}\sum_{i=1}^{M}\sum_{j=1}^{N}[I(i,j) - K(i,j)]^2, \quad (2.40)$$

де *I(i, j)* — піксель оригінального зображення в позиції *i, j*;
K(i,j) — піксель відтвореного зображення в позиції *i, j*;
M, N — розміри зображення.

Метрика SSIM надає можливість виміряти відмінність між двома зображеннями шляхом оцінювання абсолютних похибок, як визначено формулою (2.41 [78]).

$$SSIM(x,y) = \left(2\mu_x\mu_y + c_1\right)\frac{\left(2\sigma_{xy}+c_2\right)}{\left(\mu_x^2+\mu_y^2+c_1\right)}\left(\sigma_x^2 + \sigma_y^2 + c_2\right), \quad (2.41)$$

де μx та μy – середні значення інтенсивності пікселів для зображень x та y відповідно; σ_x^2 та σ_y^2 – варіації інтенсивності пікселів для зображень x та y;

σ_{xy} – коваріація між зображеннями x та y; c_1 та c_2 – константи, що додаються для запобігання діленню на нуль.

З метою виявлення спотворень зображення при вбудовуванні даних різних об'ємів у цьому дослідженні проведено серію експериментів. Для експериментів обрано дані наступних обсягів: 1кБ, 10кБ, 20кБ та 30кБ. Результати цих експериментів наведено у табл. 2.2.

Таблиця 2.2 – Результати експериментів із вбудовування даних різних обсягів та вимірювання метрик PSNR та SIMM для результуючих стегозображень

Обсяг даних	Метрика PSNR	Метрика SIMM
1 кБ	56.0322 дБ	0.9998
10 кБ	54.7245 дБ	0.9991
20 кБ	53.4931 дБ	0.9982
30 кБ	53.3004 дБ	0.9980

Отримані результати показують, що зображення майже не зазнають спотворень відносно оригінального зображення, адже значення SIMM наближається до одиниці. Також досить високі значення PSNR позначають відсутність шумів, які б також могли спотворити зображення. На рис. 2.8 наведені оригінальне зображення та зображення, у яке вбудовано 30 кБ даних.

Рис 2.8 – Порівняння оригінального (ліворуч) та стегозображення

У запропонованому методі після виконання попередніх етапів є вже сформований TJSON-об’єкт, який містить у собі розбивку у часі з вказаним кроком, тобто частотою. На етапі вбудовування інформація про модальності, відмінні від аудіовізуальних, закодовуються у кадри *mp4* файлу з відповідною частотою, як визначено вище.

Отже, алгоритм вбудовування інформації у кадри *mp4* файлу має вигляд, показаний на рис. 2.9.

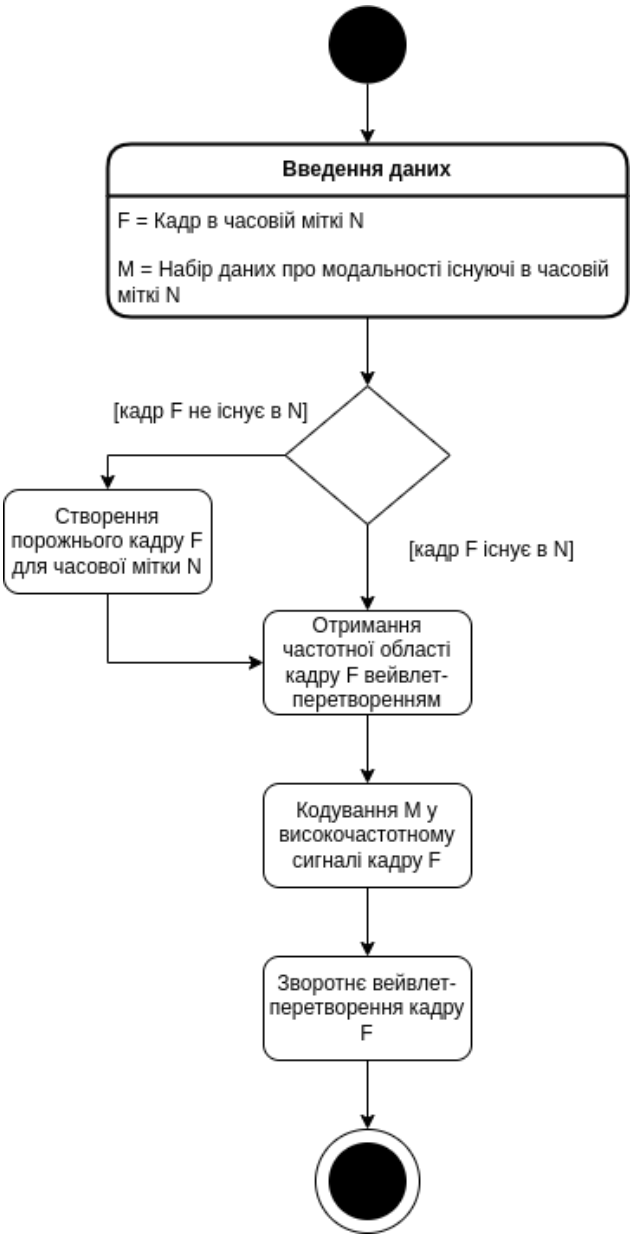

Рис. 2.9 – Алгоритм вбудовування інформації у кадри *mp4* файлу

2.5. Дослідження ефективності застосування паралельних обчислень у запропонованому методі

Для дослідження доцільності використання багатопотоковості виконано реалізацію запропонованого методу та протестовано розроблений програмний код на експериментальних даних. Метою

експерименту є визначення швидкості виконання обробки мульсемедійних даних при різних кількостях потоків.

Вхідними даними у цьому експерименті є файли наступних типів: відео-, аудіо- та CSV-файл. Тривалість відео складала 61 секунду. Тривалість аудіо – 84 секунди. Враховуючи зсув, що виник внаслідок різниці часу початку аудіо- та відео файлу, загальна часова шкала має тривалість 107 секунд. CSV-файл містив таблицю двох значень «ключ»-«значення», де «ключем» є часова мітка, а «значенням» – числове значення, що надходить із сенсора, подане у текстовому форматі.

Експеримент виконаний чотири рази зі застосуванням різної кількості потоків. Заміри виконані окремо для кожного підпроцесу запропонованого методу. Варто звернути увагу на те, що процес формування мульсемедійного файлу включає у себе також процес вбудовування даних у кадр з розрахунком у 30 кадрів на секунду. Результати визначення швидкості виконання обробки мульсемедійних даних у секундах при різних кількостях потоків наведені у табл. 2.3.

Експеримент проводився на апаратному забезпеченні з наступними характеристиками:

1. ЦП – Intel Core i5-8300H (4 ядра, 8 потоків, базова частота 2,3 ГГц, максимальна частота 4 ГГц);
2. ОЗП – 16 ГБ (частота 2667 МГц);
3. Твердотільний накопичувач – SSD NVMe (максимальна швидкість зчитування 3400 МБ/с, максимальна швидкість запису 2180 МБ/с).

Таблиця 2.3 – Результати експерименту з паралельних обчислень

Процедура	Один потік, с	Два потоки, с	Три потоки, с	Чотири потоки, с
Створення часової шкали та TJSON файлу	0,7848	0,5336	0,3139	0,2511

Об’єднання	0,0413	0,0281	0,0165	0,0132
Переріз	0,0345	0,0235	0,0138	0,011
Різниця	0,0292	0,0199	0,0117	0,0094
Симетрична різниця	0,035	0,0238	0,014	0,0112
Виключний переріз	0,0486	0,0331	0,0194	0,0155
Розміщення	0,045	0,0306	0,018	0,0144
Сортування	0,065	0,0442	0,026	0,0211
Проріджування	0,052	0,0354	0,0208	0,0166
Видалення	0,049	0,0333	0,0196	0,0157
Вставлення	0,0588	0,0411	0,0235	0,0188
Вбудовування даних в кадр	0,4305	0,2927	0,1722	0,1378
Формування мульсемедійного файлу (включає процес вбудовування даних)	1592,965	1142,4047	744,1065	619,2074

Аналіз отриманих результатів надає можливість підтвердити твердження, що зі збільшенням кількості потоків швидкість оброблення мульсемедійних даних зростає. Проте починаючи з 3-4 потоків зростання швидкості для обробки того об’єму даних, що був запропонований для експерименту, є незначним. При використанні мов програмування низького рівня, а також за наявності більш продуктивного апаратного забезпечення, можливе збільшення швидкості обробки мульсемедійних даних.

2.6. Висновки до розділу 2

Мульсемедійні дані являють собою складну структуру даних, що може містити дані довільних модальностей. Наразі існує проблема коректного злиття даних різних модальностей для їхньої консолідованої

обробки та зберігання. Запропонований у цьому розділі метод дає змогу вирішити проблему поєднання даних різних модальностей, а також їхньої синхронізації у межах єдиної часової шкали, що робить можливим створення мульсемедійних файлів для подальшої програмної обробки. Для збільшення ефективності запропонованого методу доцільно використати багатопотоковість на етапі аналізу метаданих та етапі створення вихідного файлу.

Для створення вихідного мульсемедійного файлу, запропоновано використати методи стеганографії, а саме, метод на основі дискретного вейвлет-перетворення, з подальшим кодуванням даних у високих частотах зображення. Такий підхід надає можливість поєднувати дані різних модальностей на одному часовому проміжку, незважаючи на різницю у природі та структурі цих даних.

Проведено серію експериментів для перевірки доцільності запропонованих підходів. Перевірено кореляцію коефіцієнтів спотворення зображення з об’ємом даних, що кодуються у ньому, а також швидкість виконання процесів для запропонованого методу залежно від кількості потоків, що використовуються для виконання обчислень. Результати експериментів підтвердили ефективність запропонованого методу.

РОЗДІЛ 3. ВДОСКОНАЛЕННЯ ІНСТРУМЕНТАРІЮ ДЛЯ ОБРОБКИ ДАНИХ ПРО МУЛЬСЕМЕДІЙНІ ОБ'ЄКТИ

Теоретичний апарат, запропонований для консолідації темпоральних мультимодальних даних про мульсемедійні об'єкти, а також алгоритми для вирішення прикладних задач, що пов'язані із реалізацією цифрових двійників мульсемедійних об'єктів, вимагають певного інструментарію для обробки даних про мульсемедійні об'єкти. До такого інструментарію відносяться: мова програмування, архітектурні шаблони проєктування, стандарти обробки мульсемедійних даних тощо.

У цьому розділі висвітлюються наявні та можливі підходи до обробки мульсемедійних даних. Значна увага приділяється вдосконаленню спеціалізованої доменно-орієнтованої мови програмування ASAMPL, яка призначена для спрощення процесів розроблення мульсемедійних застосунків. Також у цьому розділі пропонуються архітектурні шаблони проєктування, спрямовані на стандартизацію обробки мульсемедійних даних та спрощення процесів розроблення мульсемедійного програмного забезпечення.

3.1 Підходи до обробки даних про мульсемедійні об'єкти

Розглянемо два основних підходи до обробки даних про мульсемедійні об'єкти. Перший підхід ґрунтується на інструментарії, який надається стандартом MPEG-V, що визначає мову Sensory Effect Description Language (SEDL), глосарій Sensory Effect Vocabulary (SEV) та дискриптори Sensory Effect Metadata (SEM). Другий підхід спрямований на використання спеціалізованої мови програмування ASAMPL для розроблення мульсемедійного програмного забезпечення.

3.1.1 Обробка даних на основі стандарту MPEG-V

Основним стандартом у галузі мульсемедійних технологій і інтерактивних систем є стандарт MPEG-V [79-81]. Цей стандарт дає змогу інтегрувати широкий спектр сенсорних даних і віртуальних елементів у реальні сценарії, забезпечуючи більш повне та багатогранне сприйняття користувачами віртуального контенту. Стандарт MPEG-V визначає принципи обміну даними між віртуальними та реальними середовищами. Це включає стандартизацію форматів даних, протоколів обміну інформацією, а також методів інтеграції сенсорних відчуттів (тактильних, нюхових, смакових) у віртуальні середовища [82].

Стандарт MPEG-V складається з декількох частин, які визначають наступне [79-81].

1. *Архітектура та загальні принципи*: визначено основні поняття та загальні принципи організації взаємодії між віртуальними та реальними середовищами.
2. *Контроль та взаємодія*: визначено методи для управління віртуальними елементами та взаємодії між ними і реальним світом, що включає сенсорні відчуття, такі як зорові, слухові, тактильні та інші.
3. *Репрезентація сенсорної інформації*: стандартизовано формати для подання та кодування сенсорної інформації, що дозволяє відтворювати відчуття за допомогою апаратного забезпечення мультимедіа.
4. *Інтеграція віртуальних та реальних даних*: визначено механізми для інтеграції даних з реального світу у віртуальне середовище та навпаки, що є ключовим для створення інтерактивних мульсемедійних застосунків.

Основними компонентами стандарту MPEG-V є: Sensory Effect Description Language (мова опису сенсорних ефектів), Sensory Effect Vocabulary (словник сенсорних ефектів) та Sensory Effect Metadata

(метадані сенсорних ефектів), кожен з яких відіграє важливу роль у процесі створення, опису, та управління сенсорними ефектами в мульсемедійних системах
[83-88].

Sensory Effect Description Language (SEDL) – це мова, яка ґрунтується на форматі XML, що призначена для деталізованого опису сенсорних стимулів, асоційованих з аудіовізуальним контентом. Структура SEDL включає визначення сенсорних ефектів та їхніх характеристик, таких як тип ефекту, інтенсивність, тривалість та інші параметри [83].

Визначення типів ефектів у SEDL відбувається через XML-теги [89], що дозволяє стандартизувати описи та забезпечити їх сумісність з різними сенсорними пристроями та системами. Кожен сенсорний ефект асоціюється з відповідними параметрами, що визначають його поведінку та спосіб взаємодії з користувачем.

Тривалість та синхронізація сенсорних ефектів з аудіовізуальним контентом є важливими аспектами в SEDL. Система дозволяє точно прив'язати час активації та тривалість кожного ефекту до моментів часової шкали аудіовізуального контенту, що забезпечує синхронне відтворення сенсорних вражень.

Масштабованість SEDL забезпечується через модульну структуру XML, що дозволяє вводити нові типи сенсорних ефектів та параметри без необхідності зміни основної структури мови. Ця властивість робить SEDL адаптивною до нових вимог та технологічних інновацій у сфері мульсемедіа.

Сумісність SEDL з різними платформами та сенсорними пристроями надає широкі можливості для розробників контенту та систем інтеграції. Використання стандартизованої мови для опису сенсорних ефектів сприяє однозначності інтерпретації та відтворення сенсорних досвідів.

Sensory Effect Vocabulary (SEV) сприяє стандартизації процесів створення імерсійних мульсемедійних застосунків шляхом надання структурованого та стандартизованого набору термінів для опису характеристик та властивостей сенсорних ефектів. SEV використовується спільно з SEDL для точного опису та інтеграції сенсорних стимулів з аудіовізуальним контентом.

Кожен термін у SEV чітко визначає конкретний сенсорний ефект, описуючи його параметри, такі як інтенсивність, тривалість, частота та інші ключові атрибути. Ця стандартизація забезпечує універсальність і сумісність у створенні та відтворенні сенсорних ефектів, забезпечуючи взаємодію між різними пристроями та платформами.

Sensory Effect Vocabulary (SEV) включає різні терміни, кожен з яких описує конкретний тип сенсорного ефекту або атрибут. Наведемо приклади деяких термінів, які можуть використовуватися у SEV для опису сенсорних ефектів [87-88].

1. Типи сенсорних ефектів:
   - Light: світловий ефект;
   - Sound: звуковий ефект;
   - Vibration: тактильний ефект у формі вібрації;
   - Scent: запах;
   - Temperature: зміна температури;
   - Wind: створення потоків повітря або вітру;
   - Smoke: ефект диму або туману;
   - Water Spray: ефект розпилювання води.
2. Параметри сенсорних ефектів:
   - Intensity: інтенсивність ефекту;
   - Duration: тривалість ефекту;
   - Frequency: частота, зокрема для звукових або вібраційних ефектів;
   - Color: колір для світлових ефектів, зазвичай у форматі RGB або HEX;

- Temperature Level: рівень температури для теплових ефектів;
- Direction: напрямок, з якого приходить ефект (наприклад, для вітру) ;
- Scent Type: тип запаху.

3. Характеристики сенсорних ефектів:
   - Fade-In: час, протягом якого ефект поступово наростає;
   - Fade-Out: час, протягом якого ефект поступово зменшується;
   - Trigger Condition: умова або подія, що активує ефект.

Цей перелік не є вичерпним, адже SEV може бути розширений та адаптований для підтримки нових сенсорних ефектів і технологій, що виникають у процесі розвитку мульсемедійних систем і пристроїв.

Sensory Effect Metadata (SEM) є структурованим набором метаданих, що використовується для детального опису та управління сенсорними ефектами. SEM включають інформацію, необхідну для інтеграції, синхронізації та контролю сенсорних ефектів з аудіовізуальним контентом. Структура SEM розроблена для того, щоб забезпечити точне відтворення сенсорних стимулів у відповідності з задумом розробника контенту [84-86].

Основні характеристики SEM включають наступне.

1. *Синхронізація з аудіовізуальним контентом*: визначення часових міток для активації сенсорних ефектів, що забезпечує синхронізацію сенсорних стимулів з визначеними моментами або сегментами відео або аудіо.
2. *Деталізація сенсорних ефектів*: включає визначення параметрів, таких як інтенсивність, тривалість, частота та інші характеристики, що дозволяють точно описати та контролювати кожен сенсорний ефект.
3. *Умови активації ефектів*: визначення логічних умов або залежностей, які визначають, коли і як мають бути активовані сенсорні ефекти; це може включати залежності від інших ефектів, дій користувача або певних подій у контенті.

4. *Гнучкість та розширюваність*: SEM дозволяють легко додавати нові типи ефектів або змінювати існуючі параметри, забезпечуючи адаптивність до нових технологічних можливостей та творчих концепцій.
5. *Сумісність з пристроями та системами*: SEM розроблені з урахуванням широкої сумісності, що дозволяє інтегрувати сенсорні ефекти з різноманітними мульсемедійними системами та пристроями, забезпечуючи однорідність та консистенцію відтворення ефектів.
6. *Контроль над досвідом користувачів*: надання розробникам можливості точно налаштовувати параметри сенсорних ефектів для оптимізації користувацького досвіду, адаптації до конкретних умов відтворення та врахування індивідуальних вподобань користувачів.

Приклад лістингу XML-коду з використанням SEDL, SEV та SEM відображено в лістингу 3.1.

Лістинг 3.1 – Приклад XML-коду з використанням SEDL, SEV та SEM

```
<?xml version="1.0" encoding="UTF-8"?>
<sedl:Effects>
    <sedl:Effect>
        <sedl:EffectType>Light</sedl:EffectType>
        <sedl:StartTime>00:01:17</sedl:StartTime>
        <sedl:Duration>00:00:22</sedl:Duration>
        <sedl:Intensity>0.7</sedl:Intensity>
        <sedl:Color>#132760</sedl:Color>
    </sedl:Effect>

    <sedl:Effect>
        <sedl:EffectType>Vibration</sedl:EffectType>
        <sedl:StartTime>00:01:13</sedl:StartTime>
        <sedl:Duration>00:00:07</sedl:Duration>
        <sedl:Intensity>0.6</sedl:Intensity>
```

```
        <sedl:Frequency>5Hz</sedl:Frequency>
    </sedl:Effect>

    <sedl:Effect>
        <sedl:EffectType>Scent</sedl:EffectType>
        <sedl:StartTime>00:02:00</sedl:StartTime>
        <sedl:Duration>00:00:44</sedl:Duration>
        <sedl:Intensity>0.6</sedl:Intensity>
        <sedl:ScentType>Forest</sedl:ScentType>
    </sedl:Effect>
</sedl:Effects>
```

Хоча XML, який використовується даною технологією, довгий час відігравав роль стандартного інструмента для структурування та обміну даними у веб-застосунках та багатьох інших системах, на сьогодні він вважається застарілим через свою відносну громіздкість та складність. Сучасні формати, такі як JSON (JavaScript Object Notation), пропонують більш зручну альтернативу XML, зокрема завдяки своїй простоті, читабельності для розробника та легкості аналізу та генерації. Зокрема, JSON є зручним для використання в мережевих застосунках, особливо тих, які використовують RESTful API, що надає широкі можливості для розробників, підвищуючи продуктивність систем та покращуючи користувацький досвід, а також забезпечуючи більш ефективну взаємодію між різними компонентами розподілених систем.

Суттєвим недоліком інструментарію, який пропонується стандартом MPEG-V, є відсутність прямих механізмів обробки темпоральних мультимодальних даних. Іншим недоліком є неможливість розроблення повнофункціональних мульсемедійних програмних систем.

3.1.2 Програмна обробка даних за допомогою мови програмування ASAMPL

Мова програмування ASAMPL (від англ. *Algebraic System of Aggregates* – алгебраїчна система агрегатів і *Mulsemedia data Processing Language* – мова обробки мульсемедійних даних) – це спеціалізована мова програмування, призначена для роботи з темпоральними мультимодальними даними. Ця мова програмування орієнтована на обробку та аналіз даних з урахуванням їхніх темпоральних та модальних характеристик, що відповідає принципам парадигми програмування мультиобразів [90]. Темпоральні характеристики вказують на те, що кожен елемент таких даних пов’язаний з конкретним моментом часу, коли він був отриманий (записаний, сформований, виміряний тощо). Модальні характеристики даних вказують на природу їх походження, сумісність з іншими даними, а також можливий взаємозв’язок з іншими модальностями.

ASAMPL має низку унікальних особливостей, які відрізняють її від традиційних мов програмування. Ключовою концепцією у цій мові програмування є концепція мультиобразу, яка ґрунтується на математичному апараті алгебраїчної системи агрегатів (АСА) [52]. Вона надає можливість обробляти агрегати даних, що представляють мультиобрази об’єктів, які включають дані різних модальностей, і використовує визначені в АСА правила для їхньої обробки. Для подання мультиобразу об’єкта у цій мові програмування передбачені спеціальні структури даних – кортежі та агрегати. Обробка агрегатів виконується відповідно до правил, призначених в АСА. Це дає змогу створювати детальне та повне представлення об’єктів у мульсемедійних додатках.

ASAMPL дозволяє виконувати синхронізацію та агрегацію мульсемедійних даних, що є важливим для забезпечення правильного

подання інформації про об’єкт. Мова забезпечує можливість роботи з даними, що надходять з різних джерел, і підтримує їхню обробку в реальному часі. Така обробка є важливою для створення імерсійних середовищ та інтерактивних мульсемедійних додатків.

Основні особливості мови ASAMPL, які конкретизують принципи парадигми програмування мультиобразів є наступними [48, 90]:

- зв’язок процесу обробки даних з часовою шкалою;
- синхронізація даних різних модальностей;
- комплексне представлення мультимодальних даних за допомогою агрегатів та кортежів;
- орієнтація на одночасне використання різних джерел мультимодальних даних та різних форматів (потоки даних з віддалених сенсорів, мультимедійні файли з хмарних сховищ, тощо);
- використання апарату АСА для обробки мультиобразів та їхніх компонентів.

Ці принципи можна реалізувати також за допомогою інших мов програмування з більш універсальним застосуванням, проте це буде більш складним та ресурсозатратним процесом, адже потребуватиме більш складної логіки, більше дій та, як наслідок, більше коду.

Програмування мовою ASAMPL є відносно простим процесом завдяки наявності у мові операторів, які дозволяють розробнику виконувати базові операції над мультимодальними даними: синхронізовувати дані різних модальностей, динамічно їх замінювати та завантажувати. Особливу увагу у мові приділено такому типу даних, як часові дані. Час є особливою сутністю для мультимодальних даних, оскільки всі мульсемедійні та мультимедійні дані є темпоральними. Обчислювальна модель парадигми програмування мультиобразів може бути реалізована мовою програмування ASAMPL шляхом визначення

агрегатів темпоральних мультимодальних даних з синхронізацією потоків даних із застосуванням часового оператора.

Мультиобраз можна визначити агрегатом у блоці коду AGGREAGATES, як показано у лістингу 3.2 [91].

Лістинг 3.2 – Визначення мультиобразу об’єкта мовою ASAMPL

```
AGGREGATES {
     MultiImage = [time, data_modality_1, …, data_modality_N];
}
```

Незважаючи на низку специфічних властивостей, які роблять мову програмування ASAMPL оптимальним інструментом для розроблення мульсимедійного програмного забезпечення, вона потребує подальшого вдосконалення.

3.2 Характеристика мови програмування ASAMPL 2.0

Оскільки синтаксис мови програмування ASAMPL є доволі громіздким та складним у порівнянні з іншими сучасними мовами програмування, доцільно його вдосконалити. Оновлення синтаксису ASAMPL сприятиме прискоренню процесів розроблення, поліпшенню метрик коду та зробить мову більш привабливою для програмістів. Оновлені синтаксичні правила мови програмування ASAMPL 2.0 наведені у Додатку Б.

Базова версія мови [90] орієнтована на розроблення коротких програм, які спрямовані на створення мультиобразів об’єктів, проте мова ASAMPL має значний потенціал для розроблення також складних програмних систем для обробки темпоральних мультимодальних даних та організації взаємодії зі спеціалізованим апаратним забезпеченням, тобто

для розроблення мульсемедійного програмного забезпечення. Тому, крім оновлення синтаксису, мова ASAMPL потребує розширення функціональності. Розглянемо нові можливості, реалізовані у мові програмування ASAMPL 2.0.

3.2.1 Користувацькі функції

Як вже зазначено вище, мова програмування ASAMPL розроблена для створення невеликих «сервісних» програм для забезпечення взаємодії зі специфічним апаратним забезпеченням для мульсемедіа. У версії ASAMPL 2.0 пропонується розширити функціональність мови за рахунок уведення користувацьких функцій. Цей надасть можливість розроблювати складні програмні системи для реалізації імерсійних технологій та мульсемедіа. Користувацькі функції можуть бути викликані багаторазово у будь-якому місці в коді, що знижує ризик дублювання коду та сприяє використанню рекурсивних підходів у програмуванні. Підхід, що дозволяє описувати окремі логічні частини програми через функції, спрощує структуру коду і збільшує його читабельність. Крім того, впровадження можливості застосування рекурсії є удосконаленням, що розширює функціональні можливості мови ASAMPL [51].

У користувацьких функціях ASAMPL 2.0 застосовано оператор *return*, який надає змогу повертати обчислені значення безпосередньо у ту частину коду, звідки було ініційовано виклик функції. Використання оператора *return* також передбачає можливість його застосування без зазначення конкретних значень для завчасного припинення виконання функції, що додає додаткову гнучкість у керуванні логікою програми [51].

Для визначення граматики мови програмування ASAMPL 2.0, використаємо розширену форму Бекуса-Наура (EBNF) [92]. Тоді

декларування користувацької функції, а також її виклик можна описати синтаксичними правилами [SR1, SR2] відповідно.

користувацька_функція = function , *ідентифікатор* , [SR1]
"(" , *ідентифікатор* | *кортеж* | *набір_ідентифікаторів* |
набір_кортежів | *набір_ідентифікаторів_та_кортежів* , ")" ,
"{" , *оператор* [return , [*ідентифікатор* | *значення*] , ";"] , "}"

виклик_користувацької_функції = [*ідентифікатор* , "="] [SR2]
ідентифікатор , "(" , *ідентифікатор* | *кортеж* |
набір_ідентифікаторів | *набір_кортежів* |
набір_ідентифікаторів_та_кортежів , ")" , ";"

Діаграму станів користувацької функції зображено на рис. 3.1.

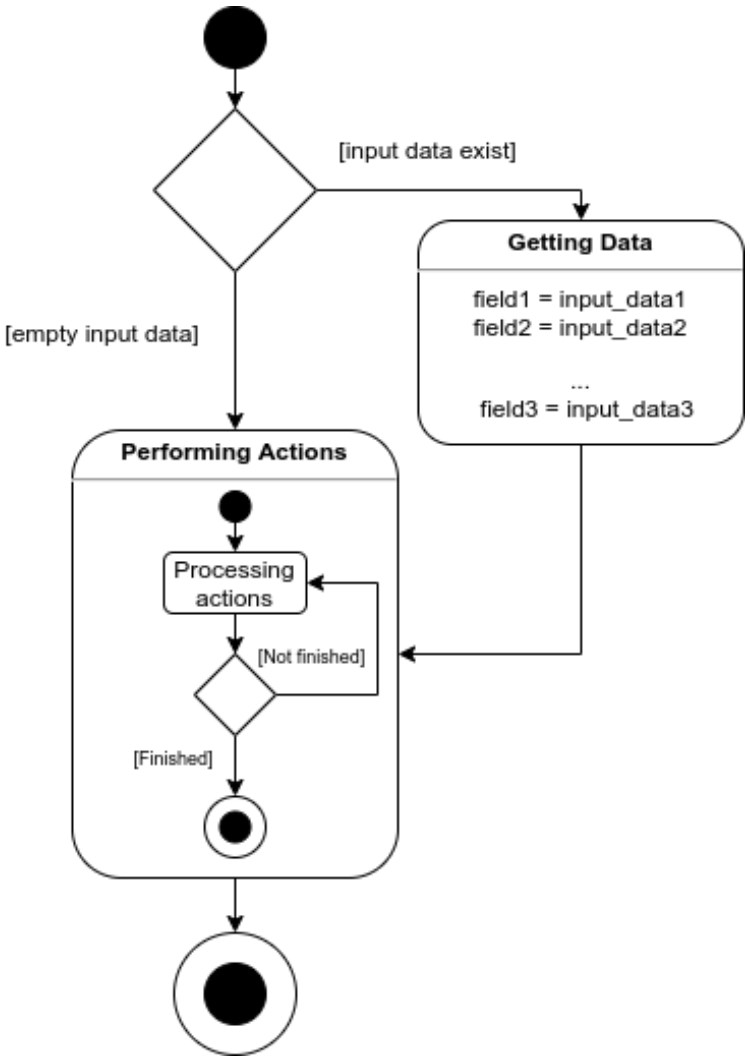

Рис 3.1 Діаграма станів користувацької функції

Використання користувацьких функцій у мові програмування ASAMPL 2.0 забезпечує більшу гнучкість, модульність та ефективність у розробленні мульсемедійного програмного забезпечення.

3.2.2. Темпоральна обробка мультимодальних даних

Одною з особливостей ASAMPL є можливість використання темпоральних операторів для координації обробки даних у часі. Це включає використання часових відміток, синхронізацію даних різних модальностей та їх агрегацію. Такий підхід дозволяє ефективно обробляти сценарії, де динамічні дані з різних джерел потрібно аналізувати та представляти у спільному темпоральному контексті.

У мові ASAMPL також велика увага приділяється агрегації та представленню мультимодальних даних. Метою цього є те, що дані з різних джерел можуть бути зібрані та оброблені разом, щоб сформувати єдине цілісне цифрове представлення об’єкта. Синтаксис мови ASAMPL включає спеціалізовані часові оператори, які дозволяють розробникам маніпулювати даними на основі часових параметрів. Ці оператори дають змогу визначати часові інтервали, протягом яких повинна відбуватися обробка даних, що дозволяє більш точно контролювати потік даних і їхню взаємодію у межах програми.

У мові програмування ASAMPL для синхронізації певного набору операцій у визначеному часовому проміжку використовується оператор *timeline* [51]. На відміну від попередньої версії мови програмування ASAMPL, у версії 2.0 запропоновано, щоб часовий оператор *timeline* мав лише одну варіацію свого застосування. Його можна виразити за допомогою синтаксичного правила [SR3].

часовий_оператор = Timeline , "(" , *ідентифікатор* | [SR3]
значення_часу , "," , *ідентифікатор* | *значення_часу* , "," ,
ідентифікатор | *значення_часу* , ")" | "(" , *кортеж_значень_часу* ,
")" | "(" , *ідентифікатор* | *логічний_вираз* , ")" , "{" , *оператор* , "}" ;

Оператор *timeline* у мові програмування ASAMPL є ключовим елементом для обробки темпоральних даних, що дозволяє розробникам здійснювати контроль над виконанням операцій у межах визначених часових рамок. Він впроваджує концепцію часової синхронізації операцій, що є особливо корисним у сценаріях, де потрібно координувати різні процеси або події, що відбуваються одночасно або послідовно у часі.

Схематично принцип роботи синхронізації операцій у часі за допомогою оператору *timeline* зображено на рис. 3.2.

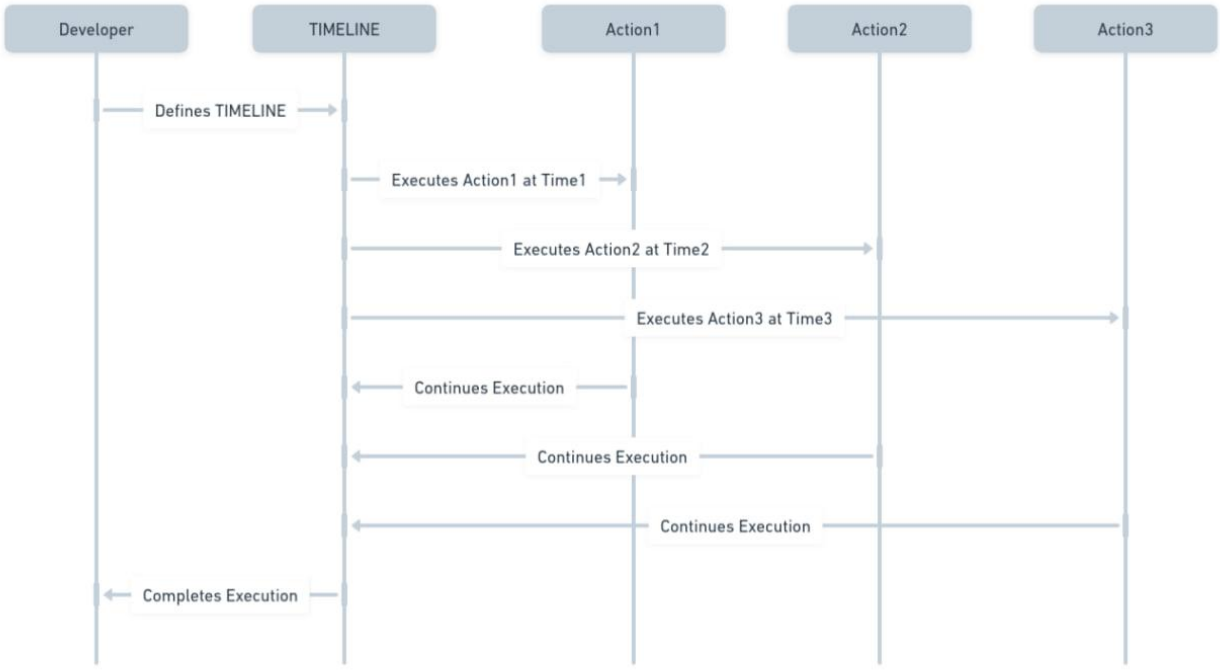

Рис. 3.2 – Принцип синхронізації операцій у часі за допомогою *timeline*

Кількість дій, виконання яких потрібно синхронізувати, може бути довільною. Оператор *timeline* надає гнучкість у визначенні часових інтервалів, дозволяючи розробнику вказувати як фіксовані часові моменти, так і умови для завершення виконання дій [51].

3.2.3 Паралельні обчислення

З метою підвищення ефективності обробки мульсемедійних даних пропонується доповнити мову ASAMPL 2.0 можливістю виконання паралельних обчислень. Це надасть можливість значно зменшити час обробки мульсемедійних даних, які звичайно характеризуються великими обсягами. Ефективне використання паралелізму в ASAMPL 2.0 не тільки зменшує час відгуку систем, але й оптимізує використання обчислювальних ресурсів, забезпечуючи високу пропускну здатність та масштабованість обробки даних.

Додавання можливості паралельних обчислень до мови програмування ASAMPL 2.0 вимагатиме низки змін та розширень на рівні синтаксису. Для забезпечення більш гнучкого застосування та ліпшої читабельності коду, запропоновано використовувати користувацькі функції, які були запропоновані в цьому розділі, як контейнер для операцій, що мають виконуватися паралельно в окремому потоці. З таким підходом синтаксис мови програмування ASAMPL 2.0 не буде обтяжений додатковими структурами та блоками коду, які б значно збільшили кількість символів коду і негативно вплинули б на його читабельність.

Таким чином, реалізацію паралельних обчислень у мові програмування ASAMPL 2.0 запропоновано розділити на наступні етапи.

1. Декларування нового окремого потоку та присвоєння йому вже реалізованої користувацької функції.
2. Старт цього потоку в потрібному місці в коді та за потрібних умов.
3. За необхідності, блокування спільних даних, які можуть опрацьовуватися одночасно в декількох потоках та бути перезаписані, що в результаті може призвести до втрати очікуваного результату.
4. Приєднання цього окремого потоку до основного потоку виконання.

Для першого етапу запропонованої імплементації паралельних обчислень запропоноване наступне синтаксичне правило [SR4], що передбачає створення потоку та присвоєння йому функції.

новий_потік = Thread , *ідентифікатор* , "=" , [SR4]
ідентифікатор_користувацької_функції, ";"

Наступний етап передбачає відгалуження нового потоку від основного потоку виконання програми. Момент, в який відбувається вказане відгалуження, визначається розробником у коді програми та виражене наступним синтаксичним правилом [SR5].

відгалуження_потоку = *ідентифікатор_потоку* , "." , start() , ";" [SR5]

Паралельні обчислення є складними та непередбачуваними через пряму залежність від механізмів планування потоків, які використовуються операційними системами. Така недетермінованість у плануванні потоків вимагає ретельного використання синхронізаційних механізмів для забезпечення коректної роботи програми в багатопотоковому середовищі. Через це, однією з найбільш складних задач є управління доступом до спільних ресурсів. Спільні ресурси, такі як загальні змінні, файли або з'єднання з базами даних чи мережевими ресурсами, можуть бути доступні декільком потокам або процесам одночасно. Однак, без належного контролю, синхронний доступ кількох потоків до одного ресурсу може призвести до конфліктів, втрати даних або некоректної поведінки програми. Ці проблеми в паралельному програмуванні звичайно називають «race conditions», коли результат виконання програми залежить від того, у якому порядку різні потоки отримують доступ до спільного ресурсу [93-94].

Блокування (locking) є одним з основних механізмів для забезпечення безпечного доступу до спільних ресурсів. Це включає використання синхронізаційних примітивів, що дають змогу контролювати доступ до ресурсу, гарантуючи, що лише один потік може використовувати ресурс у певний момент часу. Таким чином, блокування ефективно запобігає проблемі «race conditions» та іншим пов'язаним з ними проблемам [95-96].

Для розв’язання цієї проблеми у мові програмування ASAMPL 2.0 пропонується використання синхронізаційного примітиву *lock*, за допомогою функцій якого можна вказувати початок та кінець роботи з спільними ресурсами у межах одного окремого потоку, для запобігання одночасного доступу до нього з декількох окремих потоків. Пріоритетність блокування визначається присвоєнням натурального числа цьому примітиву, де більше значення вказаного числа відповідає вищому пріоритету. Цей додатковий механізм синхронізації необхідний для забезпечення більшої гнучкості паралельних обчислень, наприклад, коли розробнику важливо визначити, який із потоків є пріоритетнішим та має першочергове право на використання спільного ресурсу. Декларування запропонованого синхронізаційного примітиву визначається синтаксичним правилом [SR6].

синхронізаційний_примітив = Lock , *ідентифікатор* , [SR6]
"=", *значення_пріорітету* ";"

Після ініціалізації всіх необхідних потоків та синхронізаційних примітивів та запуску цих потоків у тілі кожного з потоків використовуються зазначені синхронізаційні примітиви для управління доступом до спільних ресурсів. Це реалізується за допомогою викликів функцій *acquire* та *release*, що застосовуються для блокування доступу до спільного ресурсу перед його використанням та для вивільнення ресурсу

після завершення роботи з ним. Ці операції забезпечують синхронізацію між різними потоками, запобігаючи конфліктам і гарантуючи консистентність даних при їх одночасному доступі до спільних ресурсів, та можуть бути виражені синтаксичними правилами [SR7] та [SR8].

блокування_доступу = *ідентифікатор_примітиву*, "." , acquire() , ";" [SR7]

вивільнення_доступу = *ідентифікатор_примітиву*, "." , release() , ";" [SR8]

Останнім етапом після створення та запуску додаткових потоків у мультипотоковій програмі є їхнє приєднання до основного потоку програми за допомогою функції *join*. Ця операція виконується у головному потоці та дозволяє забезпечити, щоб основний потік очікував завершення всіх дочірніх потоків перед продовженням свого виконання. Така організація потоків дає змогу розробнику контролювати точки синхронізації в програмі та уникати випадків, коли основний потік завершує свою роботу раніше, ніж дочірні потоки, що може призвести до непередбачуваних результатів або помилок. Відповідальність за правильне застосування приєднання відгалужених потоків покладається на розробника, який повинен правильно визначити оптимальні точки у програмному коді для виконання цієї операції, що залежить від логіки його програми забезпечуючи коректну координацію та послідовність виконання потоків. Операцію приєднання потоку до основного потоку програми, аналогічно до відгалуження, можна виразити синтаксичним правилом [SR9].

відгалуження_потоку = *ідентифікатор_потоку* , "." , join() , ";" [SR9]

Отже, для правильного та ефективного використання запропонованих паралельних обчислень в мові програмування ASAMP 2.0, розробник має чітко вказувати точки відгалуження та злиття окремих потоків з основним потоком програми, згідно з логікою його проєкту, а також забезпечити правильний порядок доступу до спільних ресурсів, шляхом їх блокування та вивільнення в коді потоків, для уникнення помилок та коректної синхронізації даних, що опрацьовуються паралельно.

Приклад застосування парадигми паралельних обчислень у мові програмування ASAMPL 2.0 наведено у лістингу 3.3.

Лістинг 3.3 – Паралельне програмування в мові ASAMPL 2.0

```
Lock lock1 = 2;
Lock lock2 = 0;
Lock lock3 = 1;
// Функція для першого потоку
function threadFunction1() {
    lock2.acquire();
    // виконання операцій зі спільними ресурсами в потоці t1
    lock2.release();
}
function threadFunction2() {
    lock3.acquire();
    // виконання операцій зі спільними ресурсами в потоці t2
    lock3.release();
}
lock1.acquire();
// виконання операцій зі спільними ресурсами в головному потоці
lock1.release();
// Декларування та відгалуження першого потоку
Thread t1 = threadFunction1;
t1.start();
```

```
// Декларування та відгалуження другого потоку
Thread t2 = threadFunction2;
t2.start();
t2.join()
lock1.acquire();
// виконання операцій зі спільними ресурсами в головному потоці
lock1.release();
t1.join()
```

У запропонованому коді реалізовано механізм паралельного виконання функцій із використанням блокування спільних ресурсів, де пріоритетність блокування задається за допомогою числових значень, привласнених синхронізаційним примітивам.

Схематично виконання паралельних процесів показано на рис. 3.3.

Синхронізаційний примітив із найвищим пріоритетом використовується у головному потоці для синхронізації доступу до спільних ресурсів. Перед виконанням операцій із спільними ресурсами головний потік намагається захопити спільні ресурси, звільнивши його після виконання операцій, що дозволяє іншим потокам отримати доступ до цих ресурсів. Така модель гарантує, що операції в основному потоці мають вищий пріоритет доступу до спільних ресурсів порівняно з іншими потоками.

Паралельно з основним потоком, створюються та запускаються два додаткові потоки з нижчими рівнями пріоритетності, проте другий потік є пріоритетнішим за перший. Це дозволяє кожному потоку безпечно взаємодіяти зі спільними ресурсами, уникаючи конфліктів та гарантуючи синхронізацію процесів. Після запуску, другий потік приєднується до головного потоку, після виконання всіх вказаних в ньому операцій, гарантуючи, що головний потік не завершить своє виконання до того, як даний потік завершить свою роботу. Аналогічна операція проводиться для

першого потоку, забезпечуючи організацію та координацію роботи всіх потоків у системі.

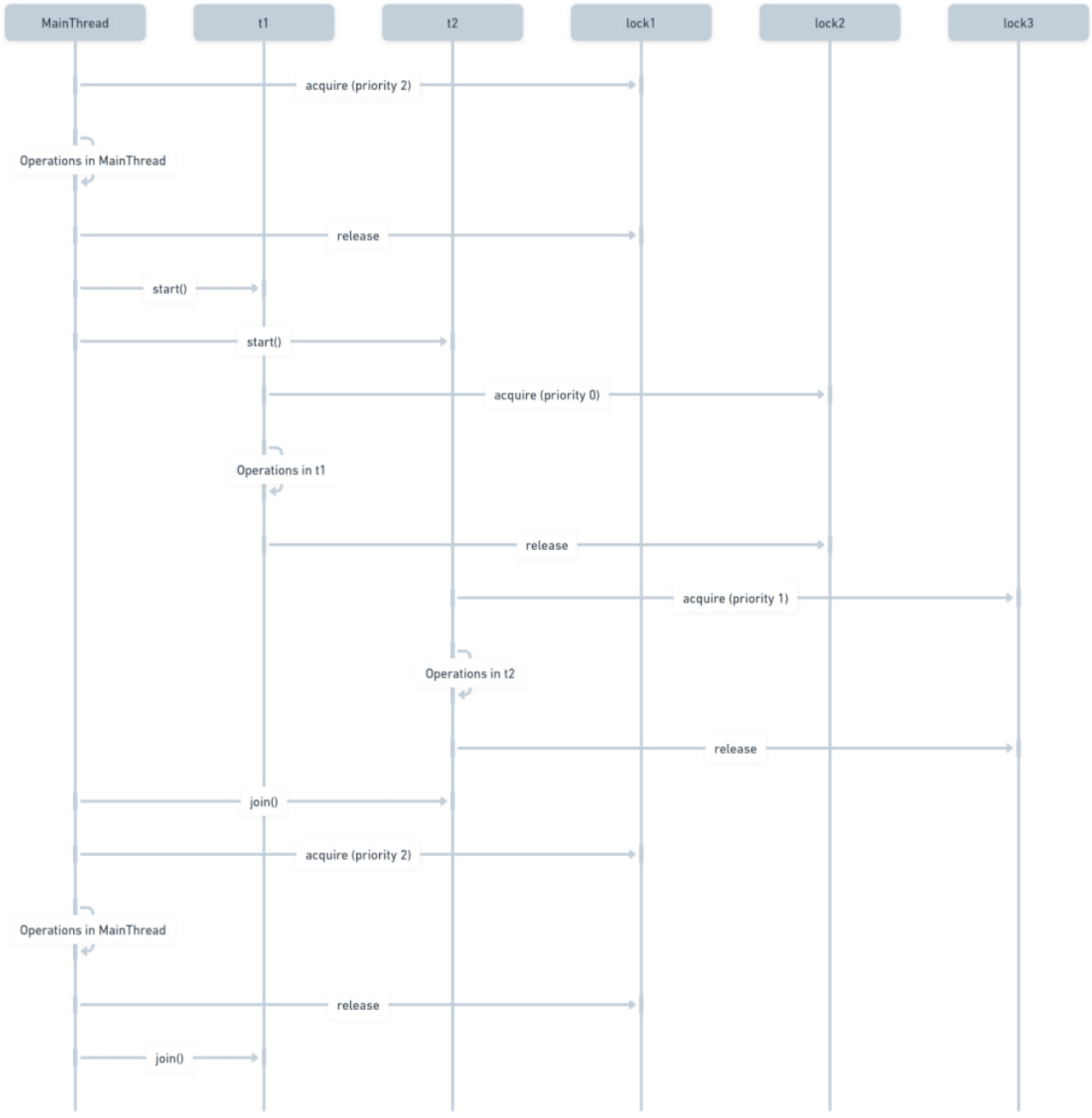

Рис. 3.3 – Схема виконання паралельних процесів у двох потоках

3.2.4. Аналіз метрик програмного коду мовою ASAMPL 2.0

Для оцінки ефективності та зручності розробленої мови програмування ASAMPL 2.0, потрібно виміряти та проаналізувати метрики програмного коду. Це дозволить об'єктивно оцінити якісні показники різних версій мови, виходячи з їхньої продуктивності та легкості використання при розробленні програмного забезпечення.

Отже, для аналізу та оцінки ефективності мови програмування ASAMPL 2.0 у цьому дослідженні застосовані три метрики: цикломатична складність, обсяг програмного коду за Холстедом та індекс зручності підтримки програмного коду.

Цикломатична складність [97] є однією з метрик складності потоку керування програм. Цей показник надає змогу оцінити складність програмного коду шляхом визначення кількості лінійно незалежних маршрутів через код програми, тим самим надаючи індикатор кількості тестових випадків, необхідних для повного тестування. Оцінка цикломатичної складності дає змогу розробникам ідентифікувати складні частини коду, які можуть вимагати додаткового рефакторингу або більш ретельного тестування. Код з високою цикломатичною складністю часто вважається більш схильним до помилок та важким для розуміння, що може впливати на якість та надійність програмного продукту.

Для визначення цикломатичної складності для певного фрагменту коду потрібно побудувати для нього граф керуючої логіки. Граф керуючої логіки є орієнтованим графом, вузли якого визначають елементарні керуючі конструкції, з яких складається цей програмний код. Два вузли у графі керуючої логіки з’єднуються ребром, якщо відповідні цим вузлам керуючі конструкції виконуються послідовно. Цикломатична складність може бути обчислена за формулою (3.1) на основі графа керуючої логіки, приклад якого наведено на рис 3.4.

$$\Psi_c = \upsilon_{edge} - \upsilon_{node} + 2 \cdot \upsilon_{comp} \quad (3.1)$$

де υ_{edge} – кількість ребер υ_{node} – кількість вузлів; υ_{comp} – кількість компонентів зв'язності [39].

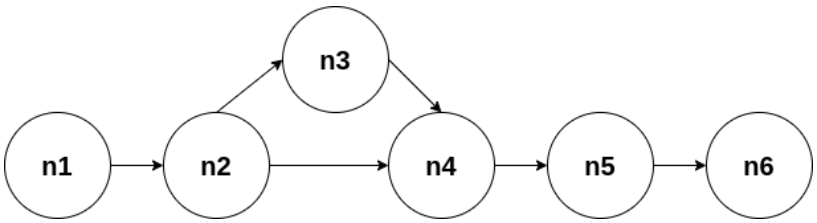

Рис 3.4 – Граф керуючої логіки для програмного коду

Обсяг програмного коду за Холстедом [98] є однією з основних метрик, яка використовується для оцінки складності програмного коду. Ця метрика ґрунтується на кількісному аналізі операторів та операндів у програмі, а також надає змогу оцінити не тільки складність програми, але й розуміння та підтримку коду. За допомогою неї розробники можуть визначити потенційну складність та зусилля, необхідні для написання, тестування та супроводу програмного забезпечення. Високий обсяг програмного коду за Холстедом може вказувати на високу складність програми та потенційну важкість її розуміння та підтримки. Обсяг програмного коду за Холстедом можна обчислити за формулою (3.2).

$$V = (\eta_0 + \eta_d) \cdot log_2(\eta_{u0} + \eta_{ud}) \quad (3.2)$$

де η_0 – загальна кількість операторів, η_d – загальна кількість операндів, η_{u0} – кількість унікальних операторів, η_{ud} – кількість унікальних операндів [8].

Індекс зручності підтримки програмного коду [99] використовується для оцінки легкості змін, підтримки та розуміння програмного коду. Цей індекс розраховується на основі декількох параметрів, включаючи цикломатичну складність, обсяг програмного коду за Холстедом, а також кількість рядків коду. Індекс зручності підтримки вираховується за формулою (3.3), що враховує ці параметри і відображає їх через певні коефіцієнти. Високі значення індексу звичайно вказують на те, що програмний код легше підтримувати і модифікувати.

$$\Psi_m = max\left(0, \frac{100 \cdot 171 - 5.2 \cdot lnV - 0.23 \cdot \Psi_c - 16.2 \cdot lnL}{171}\right) \quad (3.3)$$

де V – обсяг програмного коду, L – кількість рядків програмного коду.

Для визначення впливу синтаксису на метрики коду розглянемо реалізацію двох алгоритмічних завдань за допомогою обох версій мови ASAMPL.

У рамках першої задачі розглядається система, що складається з двох віддалених спарених камер, які використовуються для захоплення стереозображення. Кожна з цих камер обладнана вбудованими мікрофонами для запису стерео-аудіо. Крім того, у системі передбачена можливість корекції положення камер на основі даних з гіроскопу, розташованого на протилежному кінці каналу зв’язку. Така конфігурація дозволяє забезпечити високу точність та якість захоплення аудіо-відео даних.

Основною задачею є розроблення програмного забезпечення, яке надає можливість транслювати в реальному часі стерео відео- та аудіопотоки від обох камер і мікрофонів. Окрім того, програма повинна аналізувати дані з гіроскопу та відповідно корегувати положення камер для оптимального захоплення аудіо та відео. Особливою функціональною

вимогою є здатність програмного забезпечення знижувати роздільну здатність кадру залежно від поточної швидкості Інтернет-з'єднання. Це дозволяє підтримувати стабільність та неперервність відео трансляції навіть за несприятливих умов мережі. Такий підхід вимагає інтеграції різноманітних технологій та алгоритмів, включаючи обробку відео- та аудіосигналів, синхронізацію потоків даних, а також розширену обробку сигналів з гіроскопа для точного керування камерами.

Програмний код, розроблений на першій версії мови програмування ASAMPL, для цієї задачі наведено у Додатку В у лістингу В.1. Програмний код, розроблений на версії мови програмування ASAMPL 2.0, для цієї задачі наведено у Додатку В у лістингу В.2. Граф керуючої логіки, який потрібен для визначення значення цикломатичної складності, наведено на рис. 3.4.

Результати обчислень метрик для програмного коду двома версіями мови програмування ASAMPL для першої задачі наведено у табл. 3.1.

Таблиця 3.1 – Порівняння метрик програмного коду для двох версій мов програмування ASAMPL для першої задачі

Метрика	ASAMPL	ASAMPL 2.0	Зміна показника
Ψ_c (цикломатична складність)	2	2	1.00
V (обсяг програмного коду)	993.42	800.65	0.80
Ψ_m (індекс зручності підтримки)	39.054	44.694	1.14

У рамках другої задачі необхідно створити базову цифрову модель пацієнта медичного закладу на основі медичних записів, аналізів та знімків МРТ. Перелічені дані будуть використані для візуалізації поточного стану пацієнта. Проте задача ускладнюється тим, що у деяких випадках у хмарному сховищі можуть бути відсутні медичні записи з картки пацієнта,

тому в таких випадках потрібно автоматично використовувати дані, що зберігаються на локальному комп’ютері або в локальній мережі клініки.

Програмний код реалізації другої задачі за допомогою першої версії мови програмування ASAMPL наведено у Додатку Г у лістингу Г.1. Програмний код, розроблений на версії мови програмування ASAMPL 2.0, для цієї задачі наведено у Додатку Г у лістингу Г.2.

Результати обчислень метрик для програмного коду двома версіями мови програмування ASAMPL для першої задачі наведено у табл. 3.2. Граф керуючої логіки, який потрібен для визначення значення цикломатичної складності, наведено на рис. 3.5.

Таблиця 3.2 – Порівняння метрик програмного коду для двох версій мов програмування ASAMPL для другої задачі

Метрика	ASAMPL	ASAMPL 2.0	Зміна показника
Ψ_c (цикломатична складність)	3	3	1.00
V (обсяг програмного коду)	1162.81	816.61	0.70
Ψ_m (індекс зручності підтримки)	38.732	43.356	1.12

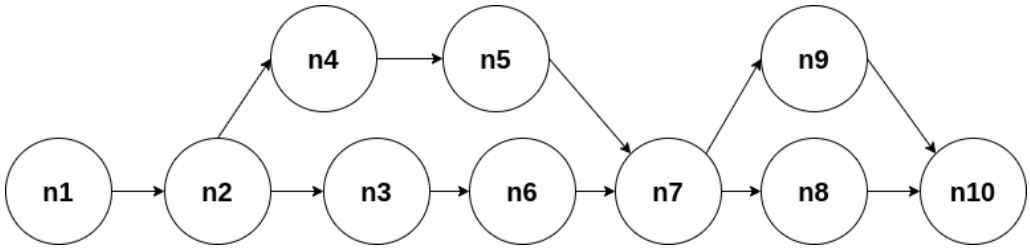

Рис 3.5 – Граф керуючої логіки для програмного коду другої задачі

Аналіз значень метрик, наведених у табл. 3.1 та табл. 3.2 дає змогу зробити висновок, що версія мови програмування ASAMPL 2.0 є більш ефективною у порівнянні з першою версією, оскільки програмний код,

розроблений на мові ASAMPL 2.0, характеризується меншим обсягом та вищім індексом зручності підтримки. Отже, ASAMPL 2.0 забезпечує розробникам більшу легкість та швидкість написання та модифікації коду.

3.3. Функції темпоральних відношень дискретних інтервалів у мові програмування ASAMPL 2.0

Як визначено у Розділі 2, дискретний інтервал – це кортеж значень, які визначають моменти часу, коли необхідно виміряти характеристики об'єкта спостереження. Для застосування дискретних інтервалів при розробленні мульсемедійного програмного забезпечення у мові програмування ASAMPL 2.0 уведено новий тип даних – *Interval*.

Значення типу *Interval* є кортежом із двох часових значень, що відповідають початковому та кінцевому моментам часу, які визначають дискретний інтервал. Для значень цього типу передбачено використання логічних функцій для визначення відношення між дискретними інтервалами.

Тип *Interval* визначається синтаксичним правилом [SR11].

інтервал = Interval , *ідентифікатор* , "=" , [SR11]
"[" , *кортеж_часових_значень* , "]" , ";"

На відміну від таких типів числових даних, як *real*, *float* та *double*, тип *Interval* спеціально призначений для визначення та обробки інтервалів часу. Це дозволяє розробникам працювати з часовими інтервалами, використовуючи вбудовані функції та операції ASAMPL 2.0.

У Розділі 2 запропоновано кількісні відношення між дискретними інтервалами. Для їх застосування у програмному коді пропонується уведення до синтаксичних правил мови ASAMPL 2.0 *функції*

темпоральних відношень дискретних інтервалів, яка визначена синтаксичним правилом [SR12].

виклик_ функції_відношення = ідентифікатор_логічної_змінної , [SR12]
"=" , *назва_функції_відношення* , "(" ,
ідентифікатор_першого_інтервалу ,
ідентифікатор_другого_інтервалу , ")" , ";"

У мові ASAMPL 2.0 передбачені такі назви функції відношення, які відповідають запопонованим кількісним відношенням, запропонованим у Розділі 2:

- *isQuantitivelyBefore* (відношення *кількісно передує*),
- *isQuantitivelyAfter* (відношення *кількісно настає після*),
- *QuantitivelyOverlaps* (відношення *кількісно перекриває*),
- *isQuantitivelyOverlappedBy* (відношення *кількісно перекривається*),
- *quantitivelyDuring* (відношення *кількісно відбувається під час*),
- *quantitivelyContains* (відношення *кількісно містить*),
- *quantitivelyStarts* (відношення *кількісно починає*),
- *isQuantitivelyStartedBy* (відношення *кількісно починається*),
- *quantitivelyFinishes* (відношення *кількісно закінчує*),
- *isQuantitivelyFinishedBy* (відношення *кількісно закінчується*),
- *coincidesWith* (відношення *збігається*),
- *meets* (відношення *стикається з початком*),
- *isMetBy* (відношення *стикається з кінцем*).

Функції темпоральних відношень використовують як вхідні дані тип даних *Interval*, який розроблений для полегшення роботи з темпоральними інтервалами, та повертають логічний тип даних, що вказує на дійсність відношення, яке перевіряється.

Приклад застосування типу даних *Interval* та функції темпорального відношення *кількісно передує* наведено на лістингу 3.4.

Лістинг 3.4 – Приклад використання типу даних *Interval* та функції темпорального відношення у мові програмування ASAMPL 2.0

```
…
Interval discreteInterval1 = [00:00:01, 00:09:00];
Interval discreteInterval2 = [00:09:30, 00:23:59];
bool result = – isQuantitivelyBefore(shiftInterval,
securityInterval);
…
```

Застосування функцій темпоральних відношень надає можливість:

- аналізувати перекриття часових інтервалів між різними подіями,
- визначати послідовності подій,
- аналізувати темпоральні характеристики об’єкта.

Уведення до мови програмування ASAMPL 2.0 типу даних *Interval* та функцій темпоральних відношень дискретних інтервалів розширило функціональність мови ASAMPL, оскільки розробники мають можливість легко визначати, контролювати та порівнювати часові інтервали, використовуючи природні та інтуїтивно зрозумілі конструкції мови.

3.4. Архітектурні шаблони проєктування для розроблення мульсемедійного програмного забезпечення

Архітектурні шаблони проєктування є фундаментальним елементом у сфері сучасної розробки програмного забезпечення. Вони є параметризованими та стандартизованими архітектурними рішеннями, що спрямовані на ефективне вирішення типових завдань розроблення. Ці шаблони слугують як рекомендації для створення структури програм, що надає можливість розробникам легко адаптувати та розширювати свої проєкти. Застосування цих шаблонів дає змогу досягти вищої гнучкості,

забезпечуючи при цьому масштабованість та підтримку великих та складних програмних систем.

Серед основних переваг використання шаблонів проєктування – зниження рівня складності розроблення та підвищення якості кінцевого продукту. Шаблони надають зрозумілу базу для кодування, дозволяючи розробникам ефективно організувати код та уникати повторень, що сприяє більшій читабельності та легкості підтримки коду. Крім того, вони сприяють забезпеченню високого рівня повторного використання коду, що є ключовим аспектом у створенні модульних та гнучких програмних систем. Шаблони також допомагають уникати помилок, які часто виникають при розробленні, оскільки вони забезпечують перевірені підходи та методики.

У контексті застосування мов програмування для створення мульсемедійного програмного забезпечення, особливо важливим є розроблення спеціалізованих архітектурних шаблонів проєктування. Ця потреба випливає з особливостей обробки мульсемедійних даних, які часто включають складні взаємодії між різними модальностями і потребують високого рівня абстракції для ефективної реалізації. Шаблони проєктування можуть надати стандартизований підхід до розроблення, що сприятиме підвищенню якості програмних продуктів та зменшенню часу, необхідного для їх створення.

Для розробки мульсемедійного програмного забезпечення запропоновані наступні архітектурні шаблони проєктування: «Мульсемедійний Конфігуратор», «Мульсемедійний Патерн», «Мульсемедійний Композитор» та «Мульсемедійний Адаптер».

Для опису розроблених шаблонів проєктування доцільно використати загальноприйняту схему опису шаблонів [100].

- *Intent (намір)*. Відповідь на питання: «Яку проблему вирішує шаблон?».
- *Also Known As (також відомий як)*. Інші відомі назви шаблону.

- *Motivation (мотивація)*. Опис сценарію, який описує архітектурну проблему, та того, як конкретні частини шаблону її вирішують.
- *Applicability (застосовність)*. Опис ситуацій, коли можливо застосувати шаблон, як їх можливо розпізнати.
- *Structure (структура)*. Графічне відображення структури шаблону у вигляді діаграми.
- *Participants (учасники)*. Опис структурних частин шаблону разом з описом того, за що відповідає кожна частина.
- *Collaborations (співпраці)*. Опис взаємодій між учасниками шаблону.
- *Consequences (наслідки)*. Наслідки використання шаблону.
- *Implementation (реалізація)*. Вказівки щодо реалізації шаблону на практиці.
- *Sample Code (приклад коду)*. Приклад застосування шаблону.
- *Known Uses (відомі використання)*. Відомі використання шаблону в реальних системах.
- *Related Patterns (пов'язані шаблони)*. Відомі шаблони, які найбільше зв'язані з шаблоном, що описується; відмінності між ними.

3.4.1. Шаблон «Мульсемедійний Конфігуратор»

Intent. Реалізувати мульсемедійне програмне забезпечення на основі концепції мультиобразу, фокусуючись лише на взаємодії з готовими агрегатами, їх поєднанні та модифікації.

Also Known As. Оскільки шаблон проєктування «Мульсемедійний Конфігуратор» запропонований уперше, він не має альтернативних найменувань.

Motivation. Розглянемо обов'язкові етапи, які необхідно виконати в процесі створення непустого агрегату, що є мультиобразом, у контексті розроблення мульсемедійного програмного забезпечення:

- отримання необхідних даних різних модальностей з різних джерел (сенсор, локальна база даних, хмарне сховище);
- декларування типів даних та параметрів, що будуть використані для формування результуючого агрегату;
- встановлення часових інтервалів або наборів часових міток для подальшої синхронізації даних різних модальностей;
- формування агрегату як непустого кортежу, що містить кортежі різних модальностей.

Таким чином, програмний код для формування агрегату є незмінним для різних сценаріїв і може відрізнятися лише кількістю та типом параметрів, що використовуються для його створення, а також наявністю певних додаткових умов чи операцій, що необхідні в конкретному сценарії. Якщо для формування агрегату за заданим сценарієм необхідно використати додаткові умови або операції, то в такому разі вони мають бути розглянуті як частина вхідних даних, разом з обов'язковими та додатковими параметрами. Такий підхід дозволить використовувати вже реалізовані мульсемедійні об'єкти без повторного написання коду та формування необхідних наборів параметрів.

При реалізації мульсемедійного програмного забезпечення, що виконує обробку даних великої кількості модальностей, які вже могли бути реалізовані в інших проєктах, або за необхідності повторного використання агрегату в різних модулях системи, може виникати проблема дублювання коду, який відповідає за реалізацію вищеописаних етапів формування агрегату. Наприклад, коли система поєднує декілька мульсемедійних об'єктів в один загальний об'єкт, використовуючи однакові кількості та типи вхідних даних для утворення різних вихідних агрегатів. Також проблема дублювання коду може виникнути внаслідок реалізації парадигми наслідування об'єктно-орієнтовного програмування, коли вже існуючий агрегат (об'єкт), потрібно доповнити новими характеристиками.

При традиційному підході ця проблема могла бути вирішена за допомогою одного із «фабричних» шаблонів, проте враховуючи темпоральну природу мульсемедійних об’єктів та необхідність їх синхронізації на єдиній часовій шкалі, виникає потреба у більш специфічному підході, що буде враховувати вказані особливості. Адже мульсемедійне програмне забезпечення вимагає точної синхронізації у часі логічно пов’язаних модальностей, що є важливим для подальшого відтворення.

Отже, головною проблемою при реалізації мульсемедійних систем, що призначені для обробки декількох агрегатів або повторно їх використовують у різних модулях системи, є дублювання коду, адже процес утворення агрегату є подібним для різних сценаріїв. Найбільш ефективним способом розв’язання цієї проблеми може бути використання зовнішніх конфігураційних файлів, які містять синхронізовану у часі структуру даних, що описує положення різних, але логічно пов’язаних модальностей на часовій шкалі (наприклад, TJSON-об’єкт [57-58] на основі JSON-файлу). Кожному із агрегатів може відповідати один конфігураційний файл, який містить у собі інформацію про типи даних, що мають бути використані, часові рамки та синхронізовані темпоральні дані. У результаті, розробник може зосередитися на реалізації основної логіки його мульсемедійного додатку, не витрачаючи час та ресурси на повторне формування однакових чи схожих агрегатів.

Таким чином, є необхідність у забезпеченні динамічної конфігурації мульсемедійного вмісту без необхідності зміни програмного коду, за допомогою зовнішніх конфігураційних файлів, що мають уніфіковану синхронізовану структуру даних і можуть бути створені сторонніми розробниками; також має залишатися можливість модифікації конфігурації.

Applicability. Запропонований шаблон проєктування «Мульсемедійний Конфігуратор» може бути застосований у наступних сценаріях:

- у системі наявна велика кількість мульсемедійних об’єктів або велика кількість повторного використання мульсемедійного об’єкту в різних модулях системи;
- необхідно уникнути дублювання коду для формування мульсемедійних об’єктів.

Structure. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 3.6. Сірим кольором виділені функції, що беруть участь у формуванні агрегату мульсемедійного об’єкту, які є типовими для різних сценаріїв.

Participants. Структурні частини запропонованого шаблону:

- InputSource1, InputSource2, ..., InputSourceN – джерела даних різних модальностей, що задаються розробником у конфігураційному файлі;
- OutputAggregate – агрегатор мультимодальних даних з можливістю їх подальшого збереження у вказаному розробником місці;
- TJSON – один із варіантів зовнішнього конфігураційного файлу на базі JSON-файлу, що може бути використаний мульсемедійним програмним забезпеченням для утворення агрегатів мульсемедійних об’єктів [58] та може бути збережений як на локальному пристрої, так і в хмарному сховищі;
- Creator – створювач агрегатів мульсемедійних об’єктів;
- GetConfigFileFunction – функція отримання зовнішнього конфігураційного файлу та його подальша трансформація в асоціативний тип даних, поля якого будуть використані як параметри для модальностей та агрегату;
- ProcessConfigFileFunction – функція обробки наявної структури, яка виконує всі підготовчі етапи формування агрегату, що були описані

раніше; також вона може виконувати додаткові операції чи умови, якщо це було передбачено розробником у вхідних даних;

- SynchroniseFunction1, SynchroniseFunction2, ..., SynchroniseFunctionN – функції синхронізації модальностей на єдиній часовій шкалі згідно з синхронізованою структурою даних, що була представлена у зовнішньому конфігураційному файлі.

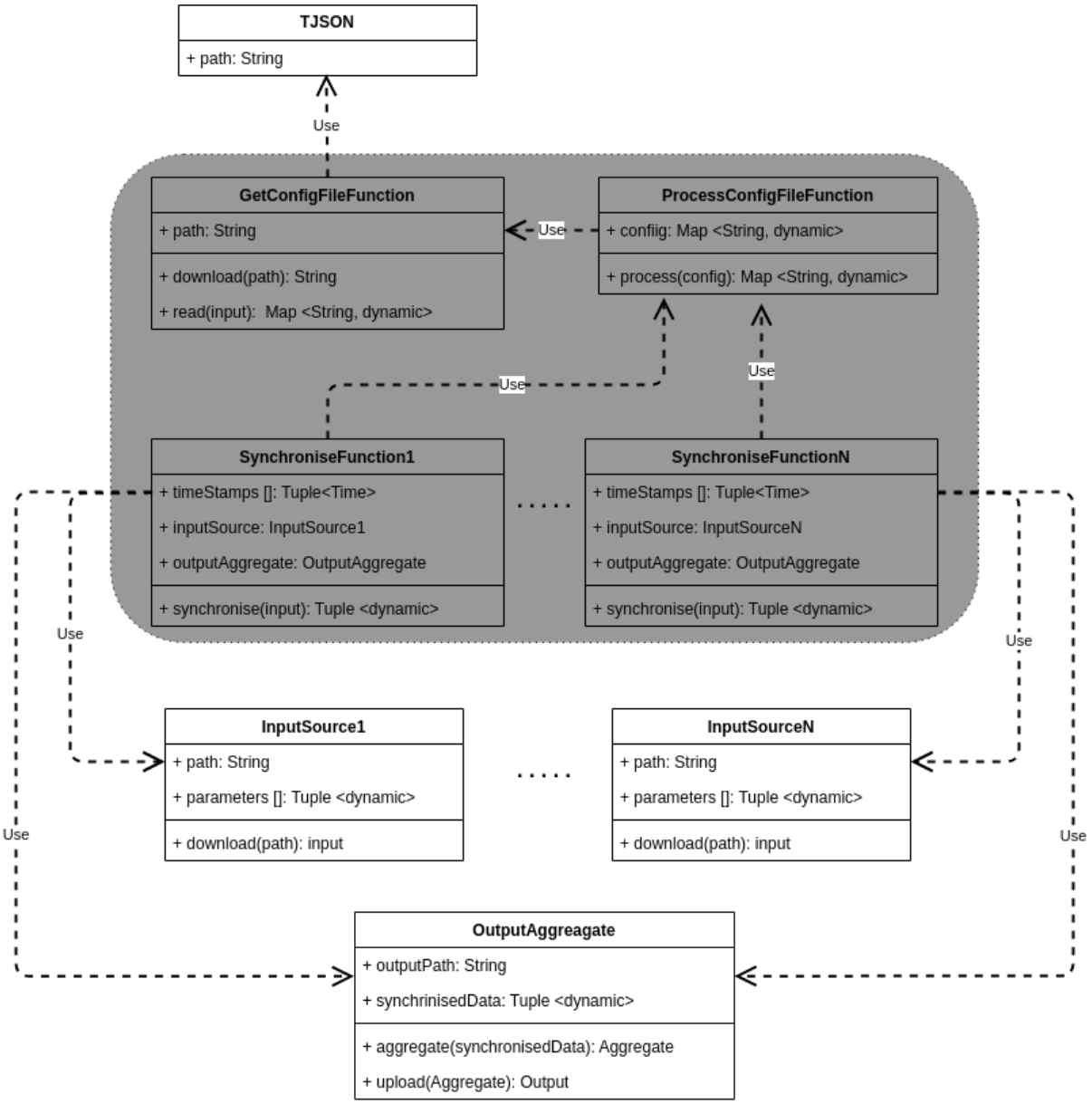

Рис. 3.6 – Структурна діаграма запропонованого архітектурного шаблону проєктування «Мульсемедійний Конфігуратор»

Collaborations. Учасники запропонованого шаблону проєктування «Мульсемедійний Конфігуратор» взаємодіють наступним чином:

- Creator використовує надані розробником вхідні дані та шлях до зовнішнього конфігураційного файлу, що може знаходитись на локальній машині або у хмарному сховищі;
- функція GetConfigFileFunction завантажує конфігураційний файл з вказаного джерела й перетворює його вміст з рядкового типу даних у формат, який дозволить мульсемедійний системі отримувати значення полів синхронізованої у часі структури даних, що є основним вмістом цього файлу;
- на основі полів конфігураційного файлу, отриманих за допомогою попередньої функції, функція ProcessConfigFileFunction створює аналогічні поля з аналогічними значеннями в мульсемедійній програмній системі, які будуть використані в подальшому для утворення агрегату мульсемедійного об’єкту;
- функції синхронізації (SynchroniseFunctionN) синхронізують окремо кожну із модальностей відповідно до єдиної часової шкали, що була визначена та описана у темпоральній структурі даних конфігураційного файлу, що в результаті повертає кортежі елементів визначених множин, які можна вважати елементами загального кортежу, що є агрегатом.

Consequences. Застосування запропонованого шаблону проєктування «Мульсемедійний Конфігуратор» має наступні наслідки:

- інкапсуляція деталей утворення агрегатів мульсемедійних об’єктів – в результаті розробник зосереджується лише на реалізації логіки мульсемедійного проєкту, що ним розробляється;
- наявність всіх необхідних даних у зовнішньому конфігураційному файлі зменшує кількість коду, що необхідно написати розробнику, тим

самим роблячи код більш гнучким та читабельним, а також спрощуючи і пришвидшуючи розроблення мульсемедійних програмних систем;

- зовнішні конфігураційні файли можна реалізувати один раз і далі використовувати їх необмежену кількість разів у різних проєктах або ділитися ними з іншими розробниками;
- за потреби дані, що знаходяться в зовнішньому конфігураційному файлі, можуть бути розширені в процесі форматування іншими даними та налаштуваннями, які необхідні для поточного сценарію застосування, або навпаки можуть бути використані не всі параметри конфігураційного файлу, тим самим спрощуючи конфігурацію фінального мульсемедійного об’єкту;
- при поєднанні агрегатів, що були утворені шляхом використання зовнішніх конфігураційних файлів, існує можливість створити новий конфігураційний файл, що буде відповідати утвореному мульсемедійному об’єкту, адже його дані вже є синхронізованими в часі та агреговані як єдина структура даних.

Implementation. Особливості реалізації запропонованого шаблону проєктування напряму залежать від особливостей конкретної мови програмування, а також від формату зовнішнього конфігураційного файлу, що буде використаний в процесі обробки даних. У цій дисертаційній роботі розроблений шаблон реалізовано з використанням мови програмування ASAMPL 2.0 та конфігураційного файлу, який містить TJSON-об’єкт [58].

Sample Code. Приклад фрагменту коду для реалізації запропонованого шаблону відображено в лістингу 3.5.

Лістинг 3.5. Реалізація запропонованого архітектурного шаблону проєктування «Мульсемедійний Конфігуратор»

```
String configJson;
Source ConfigFile = 'http://cloud-storage.com/configTJSON.json';
configJson = download(ConfigFile, default.JSONLib);
ConfigObject config = parseJSON(configJson);

Source VideoStream = config.get("videoStreamURL");
Source AudioStream = config.get("audioStreamURL");
Source GyroData = config.get("gyroDataURL");
Source VRdevice = config.get("VRdeviceURL");

Set Frame = int[ config.get("frameWidth"),
config.get("frameHeight")];
Set Audio = float();
Set Gyro = [];

Frame[] VisualDat;
Audio[] AudioDat;
Gyro[] GyroDat;

Agregate mlsdObject = [VisualDat, AudioDat, GyroDat];

Time time1 = config.get("data")[0];
Time time2 = config.get("data")[-1];
int step = config.get("step");

timeline(time1, time2, step) {
    VisualDat = download(VideoStream, default.VisualLib);
    AudioDat = download(VRdevice, MPEG2tuple);
    GyroDat = download(GyroData, default.GyroLib);
}
```

Known Uses. Оскільки шаблон проєктування «Мульсемедійний Конфігуратор» запропоновано уперше, на сьогодні він не має застосувань у відомих мульсемедійних програмних системах.

Related Patterns. Запропонований шаблон проєктування «Мульсемедійний Конфігуратор» ідейно пов’язаний з «фабричними» шаблонами, а також з шаблоном «External Configurator».

Метою застосування шаблону проєктування «Мульсемедійний Конфігуратор» є забезпечення динамічної конфігурації мульсемедійного вмісту без необхідності зміни програмного коду. Цей шаблон надає змогу розробнику створювати мульсемедійні об’єкти та визначати їх параметри за допомогою зовнішніх конфігураційних файлів або інтерфейсів. Ці параметри можуть включати налаштування відтворення, вибір джерел даних тощо. Головна перевага полягає в тому, що конфігурація може бути змінена вже під час виконання програмного коду без необхідності його зміни. Наперед задана конфігурація мульсемедійного об’єкту може бути реалізована, наприклад, за допомогою TJSON-об’єкту на основі JSON файлу, який вже є синхронізованим та визначеним на часовій шкалі [27]. За допомогою запропонованого шаблону проєктування розробники отримають можливість швидко адаптувати мульсемедійне програмне забезпечення до зміни умов або вимог користувача.

3.4.2. Шаблон «Мульсемедійний Композитор»

Intent. Забезпечити комбінування мульсемедійного програмного забезпечення на основі вже існуючих готових мульсемедійних об’єктів та їх модифікацій.

Also Known As. Оскільки шаблон проєктування «Мульсемедійний Композитор» запропонований уперше, він не має альтернативних найменувань.

Motivation. Розглянемо обов’язкові етапи, які необхідно виконати в процесі комбінування кількох готових мульсемедійних об’єктів в один загальний об’єкт:

- отримання необхідних мульсемедійних об’єктів, які необхідно комбінувати між собою (локальне або хмарне сховище);
- декларування агрегатів, що відповідають отриманим раніше мульсемедійним об’єктам і будуть використані в подальшому для модифікацій та поєднання, а також пошук спільних логічних характеристик;
- виконання модифікацій та операцій, що можуть включати операції для спрощення комбінування або операції, яких вимагає поточний сценарій застосування запропонованого шаблону (цей етап є необов’язковим);
- формування єдиного загального мульсемедійного об’єкту, який складається з логічно пов’язаних між собою за спільними характеристиками модальностей усіх використаних мульсемедійних об’єктів.

Запропонований підхід надає змогу використовувати вже наявні мульсемедійні об’єкти у нових проєктах, без написання коду для їх реалізації, а також з наявною можливістю модифікації їхніх поточних параметрів та властивостей. У разі модифікування мульсемедійних об’єктів, що використовується, необхідно надати додаткові умови чи операції як частину вхідних даних. Це дає можливість поєднувати між собою та модифікувати вже реалізовані мульсемедійні об’єкти без написання коду для створення цих об’єктів.

При реалізації складного мульсемедійного програмного забезпечення, що може містити в собі специфічне поєднання значної кількості різних мульсемедійних об’єктів, також виникає проблема дублювання коду, що була згадана при описі попереднього розробленого шаблону. Наприклад, коли необхідно реалізувати складну систему, що

складається з деякої кількості різних, але логічно пов’язаних мульсемедійних об’єктів.

У результаті, основною проблемою при реалізації складних високорівневих мульсемедійних додатків, що передбачають обробку певної кількості мульсемедійних об’єктів, також є дублювання коду, бо необхідно повторно використовувати код, вже готових мульсемедійних систем у новому проєкті. Запропонованим способом розв’язання цієї проблеми може бути використання наперед визначених додаткових параметрів, властивостей та методів при визначенні мульсемедійних об’єктів для надання можливості їх використання як модулей більш складних систем, а також можливості їх модифікації, якщо того вимагає поставлена перед розробником задача. Також, розроблений шаблон може бути використаний у поєднанні з попереднім запропонованим шаблоном у ситуаціях, коли готової мульсемедійної системи немає в наявності, проте існує конфігураційний файл, який може бути використаний для утворення мульсемедійного об’єкту.

Отже, запропонований шаблон доцільно використовувати, коли є потреба у повторному використанні готових мульсемедійних систем як модулів більш складних мульсемедійних систем для зменшення кількості необхідного коду й необхідних для розробки ресурсів, а також збільшенні швидкості розроблення, що робить розроблення нових проєктів продуктивнішим та дешевшим процесом.

Applicability. Запропонований шаблон проєктування «Мульсемедійний Композитор» може бути застосований у наступних сценаріях:

- заплановане розроблення нової мульсемедійної системи вимагає використання існуючих мульсемедійних систем як частини цієї нової системи чи самостійного модулю проєкту;

- потрібно уникнути дублювання коду при створенні мульсемедійних об’єктів, що були раніше розроблені для інших проєктів.

Structure. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 3.7.

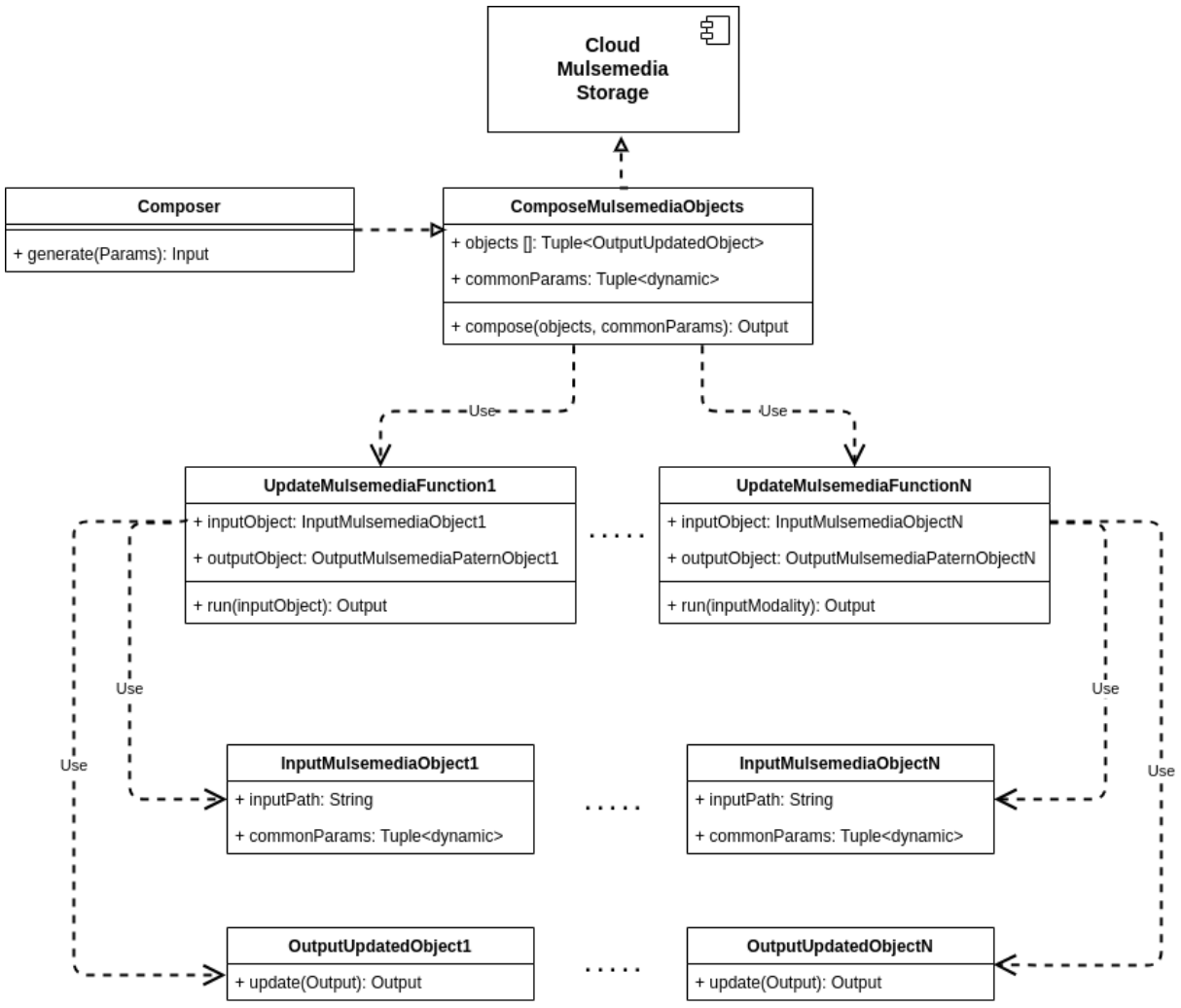

Рис. 3.7 – Структурна діаграма запропонованого архітектурного шаблону проєктування «Мульсемедійний Композитор»

Participants. Структурні частини запропонованого шаблону:

- InpuMulsemediaObject1, InpuMulsemediaObject 2, ..., InpuMulsemediaObject N – джерела готових мульсемедійних об’єктів;

- OutputUpdatedObject1, OutputUpdatedObject2, …, OutputUpdatedObjectN – модифікація та підготовка мульсемедійних об’єктів до об’єднання;
- Cloud Mulsemedia Storage – один із варіантів місця зберігання готових мульсемедійних об’єктів, що будуть використані як частини нового проєкту;
- Composer – композитор мульсемедійних об’єктів;
- ComposeMulsemediaObjects – функція поєднання мульсемедійних об’єктів;
- UpdateMulsemediaFunction1, UpdateMulsemediaFunction2, …, UpdateMulsemediaFunctionN – функції обробки наявних мульсемедійних об’єктів, які виконують всі підготовчі етапи поєднання, включно з пошуком логічних зв’язків, щоб результуючий мульсемедійний об’єкт був правильно синхронізований, а також можуть виконувати додаткові операції для кращого комбінування об’єктів між собою.

Collaborations. Учасники запропонованого шаблону проєктування «Мульсемедійний Конфігуратор» взаємодіють наступним чином:

- Composer використовує надані розробником вхідні дані про мульсемедійні об’єкти, а також їх властивості, що мають бути використані для утворення єдиної мульсемедійної системи;
- отримані об’єкти попередньо обробляються та модифікуються за допомогою функцій UpdateMulsemediaFunction, які приймають окремо кожен мульсемедійний об’єкт та необхідні параметри і в результаті віддають вже готовий до поєднання об’єкт;
- функція ComposeMulsemediaObjects отримує надані попередніми функціями мульсемедійні об’єкти і їхні властивості та формує єдину мульсемедійну систему, що містить всі надані розробником мульсемедійні об’єкти та їхні властивості.

Consequences. Застосування запропонованого шаблону проєктування «Мульсемедійний Композитор» має наступні наслідки:

- інкапсуляція коду для використання самостійних мульсемедійних систем у більш складному проєкті – в результаті розробник зосереджується лише на реалізації логіки взаємодії об’єктів, а також створенні нових об’єктів у проєкті, що ним розробляється;
- мульсемедійні об’єкти можна реалізувати один раз і далі використовувати їх необмежену кількість разів в проєктах систем різного рівня складності;
- за необхідності мульсемедійні об’єкти, що використовуються, можуть бути додатково опрацьовані та модифіковані для кращої сумісності між собою, а також для відповідності сценарію, який намагається реалізувати розробник;
- утворена складна мульсемедійна система, що призначена для обробки декількох мульсемедійних об’єктів, також може бути використана як один із елементів більш складної мульсемедійної системи, якщо це не суперечить сценарію, який потрібно реалізувати.

Implementation. Специфіка реалізації розробленого шаблону проєктування залежить від властивостей мови програмування, що буде використана для розроблення мульсемедійної системи, а також від форматів та властивостей готових мульсемедійних об’єктів, які мають бути використані. У цьому дослідженні пропонується використовувати мову програмування ASAMPL 2.0 для реалізації запропонованого шаблону.

Known Uses. Оскільки запропонований шаблон проєктування «Мульсемедійний Композитор» розроблено уперше, на сьогодні він не має відомих застосувань в мульсемедійних програмних системах.

Related Patterns. Запропонований шаблон проєктування «Мульсемедійний Композитор» є вузько-спеціалізованим, тому він не пов’язаний з іншими відомими шаблонами.

Шаблон «Мульсемедійний Композитор», який дозволяє поєднувати різні мульсемедійні додатки в один єдиний мульсемедійний додаток, надає

можливість для створення комплексних та багатофункціональних мульсемедійних систем. Він може значно спростити процес інтеграції різних медіа-компонентів та додатків, забезпечуючи при цьому високий рівень гнучкості та адаптивності.

Цей шаблон надасть розробникам можливість ефективно комбінувати вже наявні мульсемедійні рішення, наприклад, аудіовізуальні презентації, інтерактивні ігри та навчальні модулі, в єдину інтегровану платформу. Крім того, цей шаблон проєктування може сприяти створенню більш уніфікованих та зручних у використанні інтерфейсів, знижуючи необхідність для користувачів переходити між різними додатками для отримання повного спектру мульсемедійного досвіду.

3.4.3. Шаблон «Мульсемедійний Патерн»

Intent. Реалізувати мульсемедійне програмне забезпечення без витрати часу та ресурсів на реалізацію задач, які є типовими для обробки мульсемедійних даних.

Also Known As. Оскільки шаблон проєктування «Мульсемедійний Патерн» запропонований уперше, він не має альтернативних найменувань.

Motivation. Процесами, які досить часто зустрічаються при розробленні мульсемедійного програмного забезпечення, є наступні:

- завантаження даних необхідних модальностей;
- декларування типів даних та параметрів, що відповідають модальностям, які використовуються;
- виконання специфічних для кожної із модальностей операцій, що відрізняються лише вхідними параметрами.

Таким чином, у мульсемедійному програмному забезпеченні код для реалізації різних сценаріїв, що містять типові операції, є незмінним і може відрізнятися лише кількістю та типом параметрів, що використовуються

для їх реалізації. Запропонований підхід дозволить використовувати типові операції необмежену кількість разів, без повторного написання коду для них, використовуючи їх як окремі модулі, що можуть бути визначені як етапи для конкретного типу сценаріїв. Розроблений шаблон проєктування надає нові можливості у стандартизації та параметризації таких сценаріїв, що спрощує процес адаптації стандартних рішень до специфічних потреб проєкту.

Applicability. Запропонований шаблон проєктування «Мульсемедійний Патерн» може бути застосований у наступних випадках:

- для розроблення системи необхідно виконати набір типових операцій значну кількість разів;
- потрібно уникнути дублювання коду для типових операцій, що досить часто використовуються у контексті конкретної дії або набору дій у мульсемедійному програмному забезпеченні.

Structure. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 3.8. Сірим кольором виділені функції, які є частиною «патерну», який передбачений для однієї конкретної типової операції, що може бути одним із повторюваних етапів розроблення мульсемедійної системи.

Participants. Структурні частини запропонованого шаблону:

- InputSpecifiedModality1, InputSpecifiedModality 2, ..., InputSpecifiedModality N – джерела даних різних модальностей, що мають бути використані для поточного набору типових операцій;
- OutputMulsemediaPatternObject – об’єкт, що був утворений у процесі виконання повного переліку операцій, що передбачені поточним «патерном», який при певних обставинах може бути агрегатом мульсемедійного об’єкту;

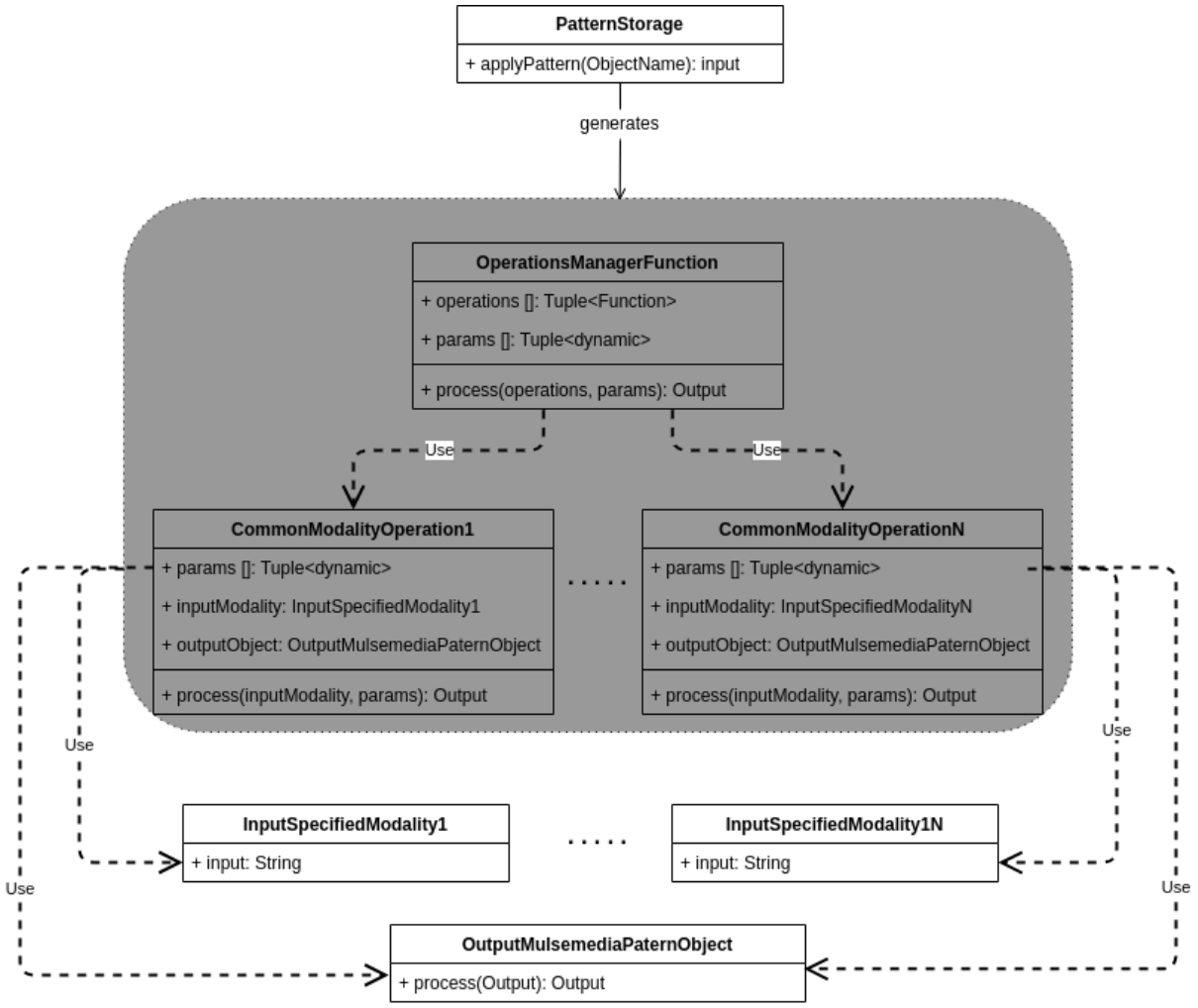

Рис. 3.8 – Структурна діаграма запропонованого архітектурного шаблону проєктування «Мульсемедійний Патерн»

- PatternStorage – сховище «патернів», яке може бути як набором модулів, що можуть бути імпортовані в код, так і вбудованими функціями інтегрованого програмного середовища;
- OperationsManagerFunction – функція, що керує станами, порядком та необхідними параметрами для виконання усіх передбачених поточним «патерном» операцій, а також відповідає за отримання очікуваного розробником результату;
- CommonModalityOperation1, CommonModalityOperation2, ..., CommonModalityOperationN – функції, які передбачені поточним

«патерном», які є типовими для модальностей, що використовуються «патерном», і є необхідними для розроблення запланованого розробником мульсемедійного проєкту.

Collaborations. Учасники запропонованого шаблону проєктування «Мульсемедійний Конфігуратор» взаємодіють наступним чином:

- функція OperationsManagerFunction отримує інструкції обраного розробником «патерну», що може зберігатися у спеціалізованому сховищі (PatternStorage), отримує необхідні дані й параметри та запускає послідовність операцій, які є типовими для вказаної модальності або сценарію;
- функції типових операцій (CommonModalityOperationN) – виконують дії, що передбачені конкретною модальністю або сценарієм та описані у «патерні», що використовується; вони є параметризованими та виконуються лише у певному порядку, що може бути змінений або доповнений новими функціями згідно певних умов, що задані розробником як вхідні параметри.

Consequences. Застосування запропонованого програмного шаблону проєктування «Мульсемедійний Патерн» має наступні наслідки:

- інкапсуляція типових операцій, що визначаються модальністю даних або сценарієм, які використовується в проєкті – в результаті розробник зосереджується лише на реалізації нетипових операцій, що відповідають за програмну логіку його мульсемедійного додатку;
- наявність опису всіх необхідних параметрів та операцій в «патерні» оптимізує кількість коду, який потрібно реалізувати розробнику, що робить програмний код гнучким і читабельним;
- «патерни» для типових операцій можна необмежено використовувати або комбінувати їх один з одним для зменшення розміру проєкту та пришвидшення розроблення;

– типові операції, що описані у «патерні» є параметризованими, що створює можливість їх повторного використання з різними параметрами, які також можуть бути додатковими функціями, що вказуються розробником і які можуть доповнювати сценарій поточного «патерну».

Implementation. Особливості реалізації запропонованого шаблону «Мульсемедійний Патерн» залежить від особливостей обраної мови програмування, а також від модальностей даних та сценаріїв, які мають бути реалізовані розробником.

Known Uses. Оскільки запропонований шаблон проєктування «Мульсемедійний Патерн» запропоновано уперше, наразі відсутні його відомі застосування для розроблення мульсемедійного програмного забезпечення.

Related Patterns. Розроблений шаблон проєктування «Мульсемедійний Патерн» віддалено пов’язаний з шаблоном «Abstract Factory».

При розробленні мульсемедійних застосунків, досить часто можуть виникати типові задачі, пов’язані зі створенням та відтворенням типових мульсемедійних об’єктів, що визначаються даними однакових модальностей і вимагають виконання однакового набору операцій. У подібних випадках розробник повинен повторно писати код для кожного із випадків, тим самим дублюючи свій код, збільшуючи розмір проєкту та зменшуючи його читабельність. Тому метою шаблону проєктування «Мульсемедійний Патерн» є реалізація сценаріїв, які можуть бути параметризовані та адаптовані для конкретного типу проєктів, і визначати типові «патерни», що значно пришвидшить розроблення, збільшить читабельність коду, стандартизує типові рішення і зменшить «поріг входу» у процеси створення мульсемедійного програмного забезпечення для нових розробників.

3.4.4. Шаблон «Мульсемедійний Адаптер»

Intent. Реалізувати гнучке управління та інтеграцію різних типів даних мульсемедійних модальностей шляхом їх конвертації до уніфікованих, наперед визначених форматів даних.

Also Known As. Оскільки шаблон проєктування «Мульсемедійний Адаптер» запропонований уперше, він не має альтернативних найменувань.

Motivation. У загальному випадку дані кожної із модальностей, що використовується мульсемедійною системою, можуть бути представлені в одному із форматів, в яких система їх зберігає, оброблює або передає. У зв’язку з відсутністю стандартизації форматів для багатьох видів мульсемедійних даних може виникнути проблема, що один або декілька форматів даних, у яких вони передані на обробку із зовнішнього джерела, не підтримуються певною системою, що унеможливить або ускладнює їхню обробку.

Таким чином, при розробленні мульсемедійного програмного забезпечення виникає потреба у застосуванні компонент, які призначені для приведення формату даних для кожної із модальностей, використання яких передбачається, до певного формату з наперед визначеного переліку форматів, який задається розробником цієї системи. Такий підхід дозволить уникнути непередбачуваних помилок та збоїв у роботі розроблюваної мульсемедійної системи, а також дозволить використовувати різні формати даних, що може бути корисним у ситуаціях, коли неможливо спрогнозувати формат даних, у якому вони будуть отримані із довільного зовнішнього джерела.

Отже, існує необхідність у забезпеченні динамічного форматування вхідних даних різних модальностей для подальшого використання у мульсемедійній системі.

Applicability. Запропонований програмний шаблон проєктування «Мульсемедійний Конфігуратор» може бути застосований у таких випадках:

- коли для коректної роботи системи передбачено використання лише конкретних форматів даних для певних модальностей, проте ці дані можуть бути збережені у довільному форматі з переліку можливих форматів, що значно відрізняються між собою;
- неможливо передбачити формати даних для модальностей, що будуть отримані: наприклад, при використанні мульсемедійної системи як загальнодоступного онлайн сервісу або при потоковій передачі даних з пристроїв різних типів.

Structure. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 3.9.

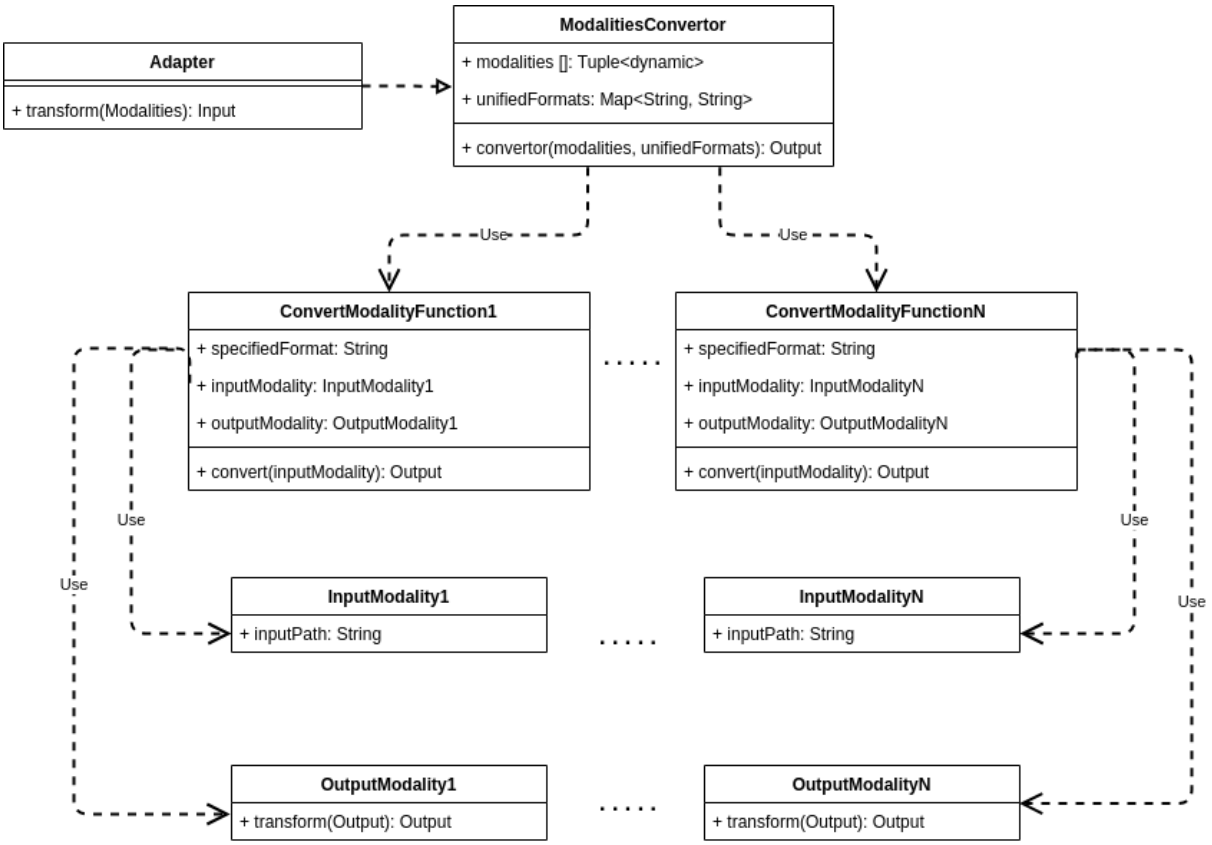

Рис. 3.9 – Структурна діаграма запропонованого архітектуного шаблону проєктування «Мульсемедійний Адаптер»

Participants. Структурні частини запропонованого шаблону:

- InputModality1, InputModality2, ..., InputModalityN – джерела даних різних модальностей, що задаються розробником;
- OutputModality1, OutputModality2, ..., OutputModalityN – модальності вхідних даних, що вже приведені до передбачених розробником форматів;
- Adapter – конвертор даних різних модальностей;
- ModalitiesConvertor – функція, яка визначає, які процеси конвертації виконувати для вказаних як параметри, що відповідають набору форматів, які задані розробником і також передані як параметри у цю функцію;

- ConvertModalityFunction1, ConvertModalityFunction2, ..., ConvertModalityFunctionN – функції конвертації, що передбачені окремо для кожної із модальностей, а в деяких випадках для окремого формату даних.

Collaborations. Учасники запропонованого шаблону проєктування «Мульсемедійний Конфігуратор» взаємодіють наступним чином:

- Adapter використовує заданий розробником перелік необхідних форматів та перелік модальностей;
- функція ModalitiesConvertor визначає необхідний перелік функцій конвертації, відповідно до заданого переліку форматів, а також співставляє задані модальності із сформованим переліком вказаних функцій форматування;
- функції конвертації модальностей (ConvertModalityFunctionN) – конвертують окремо кожну із модальностей відповідно до вказаного розробником формату або до формату, який передбачено для поточної функції «за замовчуванням».

Consequences. Застосування запропонованого шаблону проєктування «Мульсемедійний Конфігуратор» має наступні наслідки:

- інкапсуляція методів конвертування форматів даних, що дозволяє виділити цей етап в окремий модуль, який не буде впливати на логіку самого мульсемедійного проєкту;
- можливість підтримки широкого спектру форматів для кожної із модальностей, що використовується мульсемедійною системою;
- передбачуваність відповіді системи, що підвищує її стійкість та надійність;
- можливість використання сторонніх конверторів, що в свою чергу може забезпечити майбутню підтримку форматів, які будуть розроблені в подальшому.

Implementation. Особливості реалізації запропонованого шаблону проєктування залежить від особливостей конкретної мови програмування, а також від переліку форматів даних різних модальностей, що будуть використовуватися спроєктованою системою, і від наявних методів обробки й форматування цих форматів даних.

Known Uses. Оскільки запропонований програмний шаблон проєктування «Мульсемедійний Адаптер» запропоновано вперше, на сьогодні він не має відомих застосувань у мульсемедійних програмних системах.

Related Patterns. Запропонований програмний шаблон проєктування «Мульсемедійний Алаптер» пов’язаний з шаблоном «Adapter».

Розроблений шаблон «Мульсемедійний Адаптер» спрямований на вирішення задачі ефективної інтеграції та управління різними форматами даних. Запропонований шаблон надає змогу розробникам створювати адаптивні застосунки, здатні обробляти дані широкого спектру модальностей, що є важливим у контексті швидкого розвитку технологій і зростаючих вимог до мульсемедійного контенту. Використання такого шаблону полегшує процес розроблення і підтримки мульсемедійних систем й забезпечує більшу сумісність між різними пристроями та платформами.

3.4 Висновки до третього розділу

У цьому розділі розглянуто підходи до обробки мульсемедійних даних, зокрема, на основі стандарту MPEG-V та за допомогою мови програмування ASAMPL. Особлива увага приділена мові програмування ASAMPL та її вдосконаленню. Запропоновано нову версію цієї мови – ASAMPL 2.0, яка відрізняється від базової версії мови програмування

ASAMPL оновленим синтаксисом, що, як експериментально показано, призвело до поліпшення метрик коду.

Як подальший розвиток мови програмування ASAMPL у версії ASAMPL 2.0 запроваджено можливість виконання паралельних обчислень. Це дає змогу розробникам створювати багатопотокові додатки з метою оптимізації використання наявних обчислювальних ресурсів та скорочення часу обробки мульсемедійних даних.

Також запропоновано нові архітектурні шаблони проєктування: «Мульсемедійний Конфігуратор», «Мульсемедійний Композитор», «Мульсемедійний Патерн» та «Мульсемедійний Адаптер», які надають готові рішення для широкого спектру типових задач, що виникають при розробленні мульсемедійних додатків. Метою створення цих шаблонів є стандартизація, збільшення швидкості розроблення, підвищення рівня абстракції та зменшення «порогу входу» для нових розробників мульсемедійного програмного забезпечення. Також використання запропонованих шаблонів проєктування зробить процес розроблення мульсемедійного програмного забезпечення більш гнучким та надасть змогу швидко масштабувати вже реалізовані проєкти.

РОЗДІЛ 4. РОЗРОБЛЕННЯ АРХІТЕКТУР ПРОГРАМНИХ СИСТЕМ ДЛЯ ОБРОБКИ ДАНИХ ЦИФРОВИХ ДВІЙНИКІВ МУЛЬСЕМЕДІЙНИХ ОБ'ЄКТІВ

Програмна система – це група інтегрованих програмних засобів, які мають спільне цільове призначення [102]. *Архітектура програмної системи* є фундаментальною концепцією в галузі розроблення програмного забезпечення, яка визначає структурну організацію системи, її компоненти та взаємодію між ними. Вона забезпечує функціональність, ефективність та якість програмних продуктів. Архітектура програмної системи визначає високорівневу структуру і компоненти програмного продукту й набір архітектурних принципів і рекомендацій, які впливають на процес проєктування та розвитку продукту.

В основі архітектур програмних систем лежить низка принципів, серед яких – модульність, розділення відповідальності, інкапсуляція, повторне використання та конфігурування. Модульність передбачає поділ системи на відносно незалежні компоненти, кожен з яких виконує певну функцію. Це спрощує розроблення, тестування та обслуговування системи. Принцип розділення відповідальності вимагає розділення функціональності системи на відокремлені частини, що зменшує складність та збільшує гнучкість. Інкапсуляція дозволяє приховати внутрішні деталі реалізації компонентів, забезпечуючи захист даних та інтерфейсів. Принцип повторного використання полягає у використанні вже наявних програмних компонентів у нових системах, що зменшує час розроблення та забезпечує більшу надійність програмного продукту. Конфігурування надає можливість налаштовувати систему під конкретні вимоги та умови використання без необхідності зміни її вихідного коду, забезпечуючи адаптивність та гнучкість програмних рішень.

Архітектура програмних систем також відіграє важливу роль у визначенні нефункціональних вимог, таких як здатність до масштабування, продуктивність, безпека, надійність та зручність обслуговування. Ці вимоги мають критичне значення для загальної ефективності та стабільності системи. Для забезпечення цих характеристик, архітектори системи використовують шаблони проєктування та стратегії, які допомагають оптимізувати роботу системи і забезпечити її відповідність сучасним технологічним вимогам.

Отже, архітектура програмної системи має забезпечувати адаптивність до змін, що є невід'ємною частиною сучасного програмного забезпечення. Це означає, що система повинна бути готова для майбутніх розширень, інтеграцій з іншими системами, а також до впровадження нових технологій. Архітектура, яка враховує ці аспекти, сприяє створенню стійкого, гнучкого та легко підтримуваного програмного продукту, здатного задовольнити потреби користувачів.

4.1 Узагальнена архітектура програмної системи для обробки даних цифрових двійників мульсемедійних об'єктів

Розглянемо розроблення архітектури, яка забезпечує взаємодію з мульсемедійним об’єктом. Архітектура повинна враховувати особливості цифрових двійників, які представляють мульсемедійні об'єкти у цифровому форматі, та підтримувати широкий спектр сенсорів та актуаторів для реєстрації та відтворення різних видів відчуттів. Ключовим аспектом у розробленні цієї архітектури є реалізація взаємодії з джерелами даних різних модальностей та забезпечення синхронізації та обробки цих даних у реальному часі.

Враховуючи широкий спектр застосувань цифрових двійників мульсемедійних об’єктів, запропонована узагальнена програмна

архітектура повинна бути гнучкою та масштабованою. Це означає, що вона повинна легко адаптуватися до різних умов використання, від індивідуальних мульсемедійних застосунків до комплексних індустріальних систем. Такий підхід забезпечить здатність системи до еволюції та підтримки тривалого життєвого циклу [101].

Спроєктована узагальнена архітектура програмної системи для обробки даних цифрових двійників мульсемедійних об'єктів складається з компонентів, які відповідають за конкретні етапи роботи з темпоральними мультимодальними даними для реалізації технології цифрових двійників. Схематично цю архітектуру, її компоненти та зв'язки між ними, можна відобразити, як показано на рис 4.1.

Розроблена архітектура передбачає, що мульсемедійний об'єкт описується даними кількох модальностей. Наприклад, якщо мульсемедійний об'єкт є газорозподільчим вузлом, він може бути представлений даними чотирьох модальностей: (1) він може бути візуально контрольований за допомогою відеокамери; (2) він може бути керований через тактильний інтерфейс; (3) його поточний стан може бути зареєстрований з використанням детектора запахів та (4) за допомогою датчиків тиску газу. Застосовуючи сенсори та актуатори для цих модальностей, моніторинг та керування газорозподільчим вузлом можуть бути забезпечені в дистанційному режимі на якісно новому рівні шляхом розроблення та використання відповідного мульсемедійного програмного забезпечення. Якщо мульсемедійний об'єкт є технічним об'єктом, мульсемедійне програмне забезпечення може бути реалізоване як платформа цифрового двійника. Якщо передбачається, що мульсемедійний об'єкт буде реалізований як віртуальний об'єкт, мульсемедійне програмне забезпечення може бути частиною платформи метавсесвіту (Metaverse) [101].

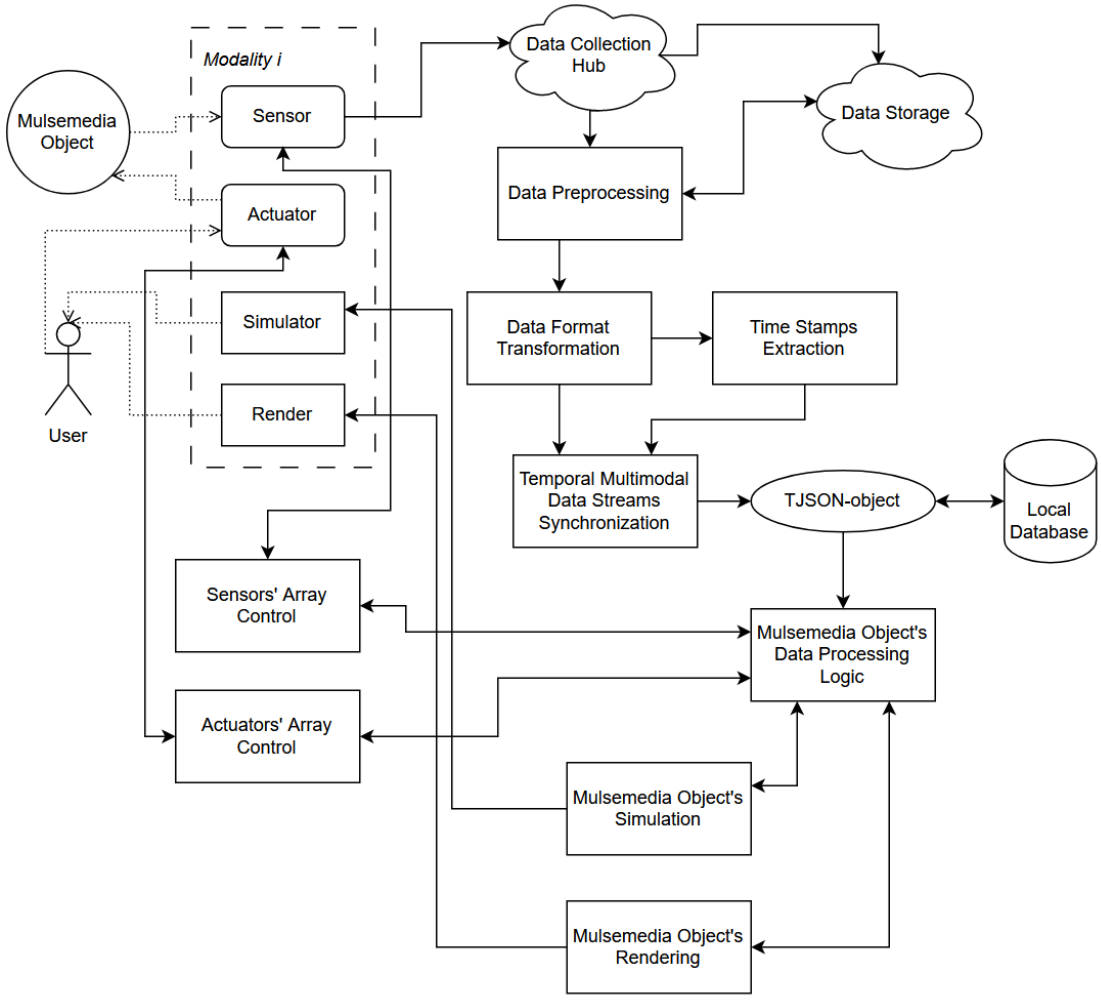

Рис. 4.1 – Узагальнена архітектура програмних систем для обробки даних цифрових двійників мульсемедійних об'єктів

Ядром запропонованої системи є модуль *логіки обробки даних мульсемедійного об'єкту* (Mulsemedia Object's Data Processing Logic). Він представляє важливу частину обробки темпоральних мультимодальних даних для заданої задачі, пов'язаної з роботою мульсемедійного об'єкту, та відповідає вимозі ФВ14, сформулоованій у Розділі 1. Передбачається, що цей модуль реалізує операції над TJSON-об'єктами.

У розробленій архітектурі TJSON-об'єкт створюється модулем *синхронізації потоків темпоральних мультимодальних даних* (Temporal Multimodal Data Streams Synchronization) та зберігається у *локальній базі*

даних (Local Database). Також цей об’єкт може бути використаний для реалізації шаблону проєктування «Мульсемедійний Конфігуратор», а саме для завантаження вже реалізованих мульсемедійних об’єктів із локальної бази даних або хмарного сховища. Запропонований підхід дає змогу досить швидко розгортати мульсемедійні системи, а також вдосконалювати і оновлювати їх, а за потреби перепрофілювати для інших задач та потреб.

Модуль *синхронізації потоків темпоральних мультимодальних даних* реалізує метод консолідації мульсемедійних даних, що дозволяє поєднувати та синхронізувати мульсемедійні дані, використовуючи принципи багатопотоковості. Модуль реалізує вимоги ФВ7 та ФВ8, що запропоновані у Розділі 1. Модуль отримує дані різних модальностей та їхні часові мітки як вхідні дані. У свою чергу, дані певної модальності та їхні часові мітки є вихідними даними двох інших модулів: модуля *перетворення формату даних* (Data Format Transformation) та модуля *видобування часових міток* (Time Stamps Extraction).

Призначення модуля *перетворення формату даних* – перетворити дані певної модальності з їх оригінального формату на формат, який використовується як основний у конкретній мульсемедійній програмній системі. Модуль реалізує вимогу ФВ5 запропоновану у першому розділі. Критерії вибору відповідного формату залежать від декількох аспектів, включаючи завдання для дослідження мульсемедійного об'єкта та обладнання, що використовується для сприйняття цього об'єкта. Проблема полягає в тому, що наразі обладнання для мульсемедіа не стандартизоване, особливо, коли мова йде про такі специфічні модальності, як тактильні, нюхові та смакові відчуття. Це означає, що не існує заздалегідь визначеного формату представлення даних, який слід використовувати в мульсемедійній програмній системі. Водночас це надає розробникам певну свободу у роботі з мультимодальними даними.

Модуль *видобування часових міток* призначений для видобування темпоральної інформації з мультимодальних даних, представлених у вигляді потоку даних або файлу у певному форматі. Він отримує дані певної модальності від модуля перетворення формату даних. Модуль реалізує вимогу ФВ6, описану у Розділі 1.

Кожна модальність інформації, яка визначає мульсемедійний об'єкт, підтримується набором пристроїв: сенсорами для реєстрації даних цієї модальності та актуаторами для надання зворотного зв'язку. Керування цими пристроями виконується через модуль *керування масивом сенсорів* (Sensors' Array Control) та модуль *керування масивом актуаторів* (Actuators' Array Control). Модулі реалізують вимоги ФВ3, ФВ4, ФВ9 та ФВ10, що були вказані у Розділі 1. Ці модулі обмінюються інструкціями з модулем *логіки обробки даних мульсемедійного об'єкта*.

Потоки даних усіх модальностей передаються з масиву сенсорів для реєстрації даних різних модальностей. Ці потоки збираються *хабом збору даних* (Data Collection Hub), який є хмарним компонентом мульсемедійної програмної системи. Модуль реалізує вимоги ФВ1 та ФВ2, запропоновані у першому розділі. Цей модуль спрямовує дані як до *хмарного сховища даних* (Cloud Data Storage), так і до модуля *передобробки даних* (Data Preprocessing). Призначення модуля *передобробки даних* – поліпшити якість даних перед тим, як потік даних буде оброблений модулем *перетворення формату даних*.

Взаємодія користувача з мульсемедійним об'єктом або його цифровим відображенням (цифровим двійником) забезпечується за допомогою набору симуляторів, рендерів та актуаторів, які працюють з певною модальністю. *Симулятор* (Simulator) може бути реалізованим як компонент мульсемедійної програмної системи або як стороннє програмне забезпечення, сумісне з цією системою. Використання симуляцій підтримується модулем *симуляції мульсемедійного об'єкта* (Mulsemedia

Object's Simulation), який обмінюється інструкціями з модулем *логіки обробки даних мульсемедійного об'єкту*. Модуль реалізує вимогу ФВ12 та ФВ13, що були запропоновані у Розділі 1.

Візуалізатор (Render) являє собою одночасно пристрій та його драйвер, які забезпечують відтворення інформації для певної модальності. Взаємодія між драйвером рендера та модулем *логіки обробки даних мульсемедійного об'єкту* здійснюється через модуль *візуалізації мульсемедійного об'єкта* (Mulsemedia Object's Rendering). Модуль реалізує вимогу ФВ11, описану у Розділі 1.

Запропонована архітектура, розроблена для обробки темпоральних мультимодальних даних, які визначають мульсемедійний об'єкт, передбачає використання комплексного підходу до збору, обробки та аналізу даних з різних джерел та модальностей. Це дозволяє досягти вищого рівня інтеграції та синергії між різними видами сенсорних даних, таких як візуальні, аудіо, тактильні, нюхові та інші, забезпечуючи більш точне розуміння мульсемедійного об'єкта. За допомогою цієї архітектури можна створювати складніші та більш інтерактивні додатки, що відкриває нові можливості для розвитку цифрових двійників та застосунків у метавсесвіті.

Реалізація зазначеної архітектури можлива через використання як універсальних, так і спеціалізованих мов програмування, наприклад, мови програмування ASAMPL 2.0, яка надає можливості для забезпечення ефективної обробки саме мульсемедійних даних.

Представлена високорівнева архітектура програмного забезпечення для мульсемедійних систем має на меті спрощення розроблення прикладних програмних рішень на основі мульсемедійних технологій для широкого спектру застосувань. Розроблена узагальнена архітектура може бути адаптована до різних наборів темпоральних мультимодальних даних, які описують мульсемедійний об'єкт. Основною особливістю цієї

архітектури є те, що вона передбачає: (1) видобування прихованих часових міток із даних, представлених у різних форматах; (2) композицію даних усіх модальностей у вигляді одного TJSON-об'єкта; (3) можливість різних сценаріїв роботи з інформацією про мульсемедійний об'єкт [43].

Подальші дослідження можуть бути спрямовані на розроблення специфічних програмних бібліотек, які реалізують процедури роботи з сенсорами, актуаторами, пристроями відображення та симуляторами для різних модальностей, що представляють мульсемедійний об'єкт. Ці бібліотеки нададуть змогу оптимізувати процес розроблення мульсемедійних застосунків й підвищити якість взаємодії користувача з мульсемедійним контентом, забезпечуючи більш глибоке залучення та інтерактивність. Важливим аспектом також є інтеграція різних типів даних, що дозволяє створювати більш цілісні та реалістичні віртуальні середовища, відкриваючи нові перспективи у використанні мульсемедіа та розширеної реальності для широкого кола застосувань. Іншим важливим аспектом є безпека даних та приватність користувачів. У контексті мульсемедійних систем, які часто включають чутливі дані, забезпечення конфіденційності та захисту інформації відіграє ключову роль у сприйнятті цих технологій кінцевими користувачами.

4.2 Основні компоненти архітектури та принципи функціонування медичної програмної системи для обробки даних цифрового двійника пацієнта

Застосуємо розроблену узагальнену архітектуру для проєктування медичної програмної системи на основі технології цифрових двійників. Ця система призначена для збору, аналізу та візуалізації медичних даних, що дозволяє створювати цифрові двійники пацієнтів. Основна мета такої системи – забезпечення більш точного та ефективного моніторингу стану

здоров'я пацієнтів. Досягнення цієї мети сприятиме підвищенню ефективності медичного обслуговування.

Важливим елементом проєктованої медичної програмної системи на основі технології цифрових двійників є модуль збору даних. Система використовує широкий спектр медичних сенсорів для отримання комплексної інформації про стан пацієнта, включаючи вимірювачі серцевого ритму, пульсу, кров’яного тиску, рівня кисню в крові, температури тіла тощо, що надає змогу збирати повний набір даних про фізичний стан пацієнта, які надалі мають бути проаналізовані та візуалізовані для підтримки лікарських рішень.

Проєктована архітектура передбачає можливість інтеграції нових видів сенсорів та адаптації до змін у медичних протоколах та стандартах. Запропоновану архітектуру, її компоненти та взаємодію між ними зображено на рис. 4.2.

Користувачем запропонованої медичної системи є *лікар* (Doctor), який взаємодіятиме з цією системою та отримуватиме у реальному часі візуалізований набір даних про пацієнта, а також матиме змогу на основі отриманих даних цифрового двійника пацієнта проводити симуляції, щоб промоделювати потенційний майбутній стан пацієнта.

Центральним елементом медичної системи є *пацієнт* (Patient), що відображає живий суб'єкт, чиї фізіологічні та біохімічні параметри постійно вимірюються. *Блок пристроїв*, які збирають дані (Sensors) – це *сенсори температури* (Temperature Sensors), *пульсометри* (Pulsometers), *вимірювачі кров'яного тиску* (Blood Preasure Sensors), *глюкометри* (Glucometers), сенсори *визначення рівня кисню* в крові (Blood O_2 Sensors) та *пристрої моніторингу дихання* (Breathe Sensors). Вони забезпечують точне та неперервне відстеження життєво важливих показників, на основі яких лікарі можуть ставити діагнози та назначати лікування або симулювати на

цифровому двійнику можливу реакцію організму людини на певні втручання або навпаки – їх відсутність.

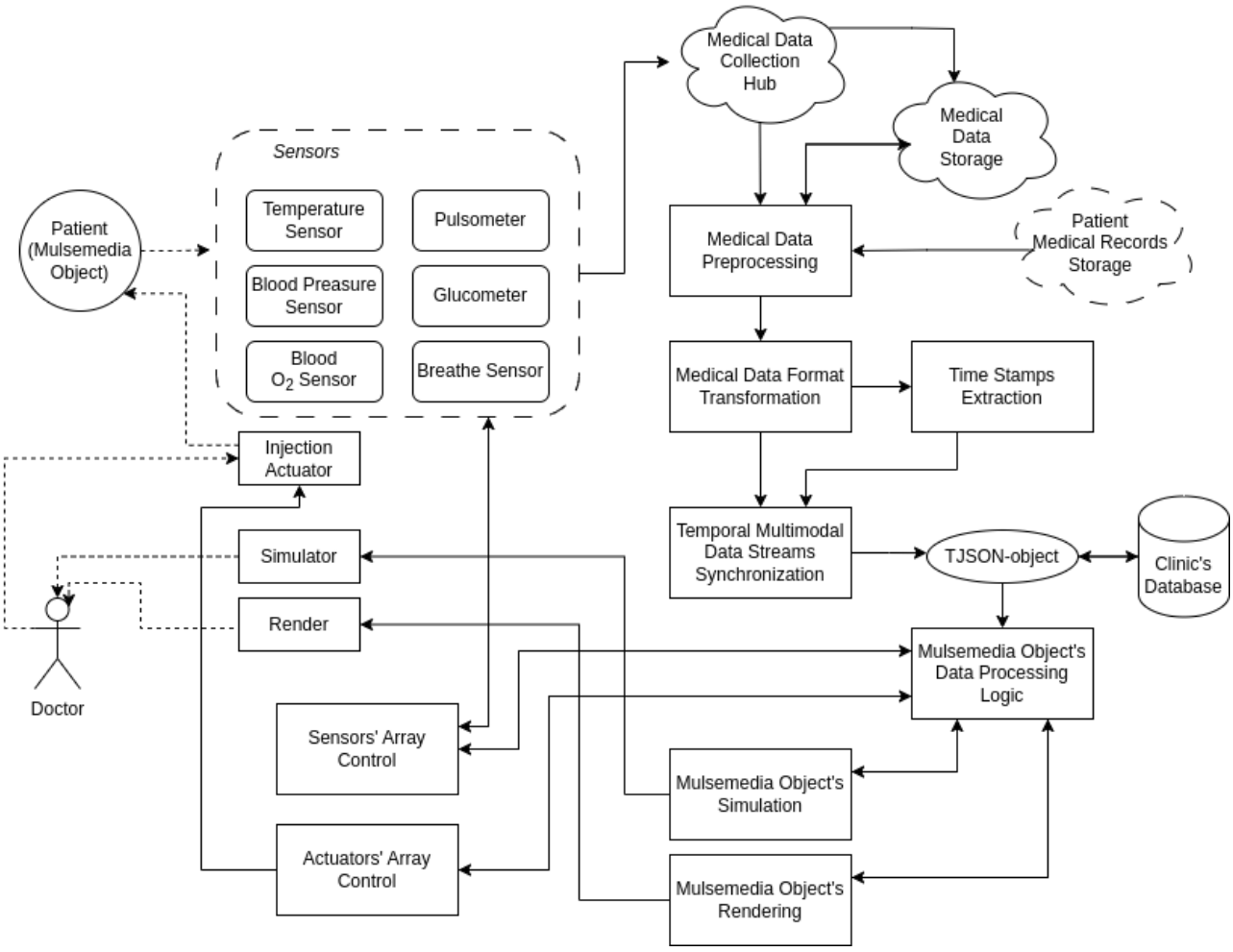

Рис. 4.2 – Архітектура медичної програмної системи для персоналізованого цифрового двійника пацієнта

Ін'єкційний Актуатор (Injection Actuator) призначений для автоматизованого або віддаленого ін'єкційного введення препаратів.

Симулятор (Simulator) надає можливість медичним працівникам моделювати різні клінічні сценарії для конкретного пацієнта. Передбачається, що використання симулятора сприятиме підготовці лікарів до ефективного реагування на можливі ситуації, що вимагатимуть невідкладної допомоги пацієнту. Симуляція допомагатиме наперед

визначати оптимальні лікувальні стратегії. Цей інструмент є обов’язковою частиною системи, яка забезпечує моделювання на основі цифрового двійника пацієнта.

Візуалізатор (Render) перетворює складні медичні дані та результати симуляції у зрозумілі візуальні образи, такі як графіки, діаграми, тривимірні моделі анатомії чи інтерактивні панелі. Ця функціональність критично важлива для забезпечення лікарів інструментами, які підвищують швидкість прийняття лікарського рішення.

Сховище медичних даних пацієнта (Patient Medical Records Storage) відповідає за агрегацію історичних медичних мультимодальних даних, такі як клінічні записи, результати лабораторних аналізів, медичні зображення та нотатки лікарів. Це дозволятиме створити багатовимірну картину стану здоров'я пацієнта, що є критично важливим для створення точного цифрового двійника. Передбачається, що отримані дані надалі використовутимуться у модулях візуалізації та симуляції.

Модулі *керування масивом сенсорів* (Sensors' Array Control) та *керування масивом актуаторів* (Actuators' Array Control) забезпечують координацію та управління сенсорами та актуаторами відповідно. Це дозволяє системі реагувати на зміни в стані здоров'я пацієнта в реальному часі, конфігуруючи налаштування сенсорів і актуаторів для поточного завдання.

Медичний хаб збору даних (Medical Data Collection Hub) є вузлом, куди надходять усі дані від сенсорів. Звідси вони потрібляють до модуля *передобробки медичних даних* (Medical Data Preprocessing), приводяться до єдиного формату у модулі *перетворення формату медичних даних* (Medical Data Format Transformation) і синхронізуються у модулі *синхронізації потоків темпоральних мультимодальних даних* (Temporal Multimodal Data Streams Synchronization).

Модуль *логіки обробки даних мульсемедійного об'єкту* (Mulsemedia Object's Data Processing Logic), модуль *симуляції мульсемедійного об'єкта* (Mulsemedia Object's Simulation) та модуль *візуалізації мульсемедійного об'єкта* (Mulsemedia Object's Rendering) реалізують подальшу обробку даних для визначення медичних станів і візуалізацію цих станів. Всі оброблені дані у подальшому зберігаються у *базі даних клініки* (Clinic's Database), забезпечуючи безперервність та надійність медичного документування.

Розглянемо приклад застосування запропонованої медичної системи для моніторингу стану пацієнта з хронічним захворюванням сердця.

На першому етапі відбувається використання пристроїв вимірювання серцевого ритму, кров'яного тиску та інших сенсорів, що інтегровані у систему, для отримання даних про функціонування серцево-судинної системи пацієнта в режимі реального часу.

Другим етапом використання запропонованої системи є обробка та аналіз зібраних даних з метою створення цифрового двійника серця пацієнта. При цьому зібрані медичні дані інтегруються та перетворюються у деталізовану цифрову модель серця, що дозволяє візуалізувати його структуру та параметри функціонування. За допомогою модуля симуляції може бути промодельована реакція серцевого м'яза на лікувальні втручання для виявлення потенційних проблем.

На третьому етапі система має використовуватися для планування конкретних лікувальних заходів.

Таким чином, запропонована медична програмна система для обробки даних цифрового двійника пацієнта може бути використана для довгострокового моніторингу стану здоров'я пацієнтів. Подальше розширення запропонованої системи можливе за рахунок застосування модуля аналізу медичних даних на основі алгоритмів штучного інтелекту.

4.3 Основні компоненти архітектури та принципи функціонування програмної системи для підтримки тренінгу фахівців, професійна діяльність яких пов’язана з підвищеним ризиком

Програмні системи для підтримки тренінгу фахівців, професійна діяльність яких пов’язана з підвищеним ризиком, призначені для проведення тренувань у реалістичних умовах із забезпеченням безпеки та ґрунтуються на застосуванні сенсорів, актуаторів та симуляторів, а також інтерактивних технологій для створення реалістичних тренувальних сценаріїв. Це дозволяє фахівцям зануритися у відтворене середовище, яке імітує реальні умови їхньої роботи.

Особливо актуальними такі системи стають у галузях, де фахівці постійно стикаються з високим рівнем ризику, наприклад у професії рятувальника ДСНС. Використання програмних систем для тренінгу рятувальників дозволяє відтворювати різноманітні сценарії, від стандартних процедур евакуації до рідкісних або непередбачуваних екстрених ситуацій.

Розглянемо архітектуру програмної системи для віртуальних тренінгів рятувальників, що містить підсистему зворотного зв’язку з аналізом та оцінкою пройденого тренінгу. У системі передбачено можливість вдосконалення сценаріїв створення симульованого середовища за рахунок накопичення даних реальних надзвичайних ситуацій, наприклад, пожеж в житлових будинках або господарських приміщеннях. Запропоновану архітектуру відображено на рис. 4.3.

Користувачем програмної системи є *рятувальник* (Firefighter), що проходить навчання або підвищує свою кваліфікацію.

Головним компонентом запропонованої системи є *цифровий двійник надзвичайної ситуації* (Emergency), що являє собою віртуальний еквівалент реальної події.

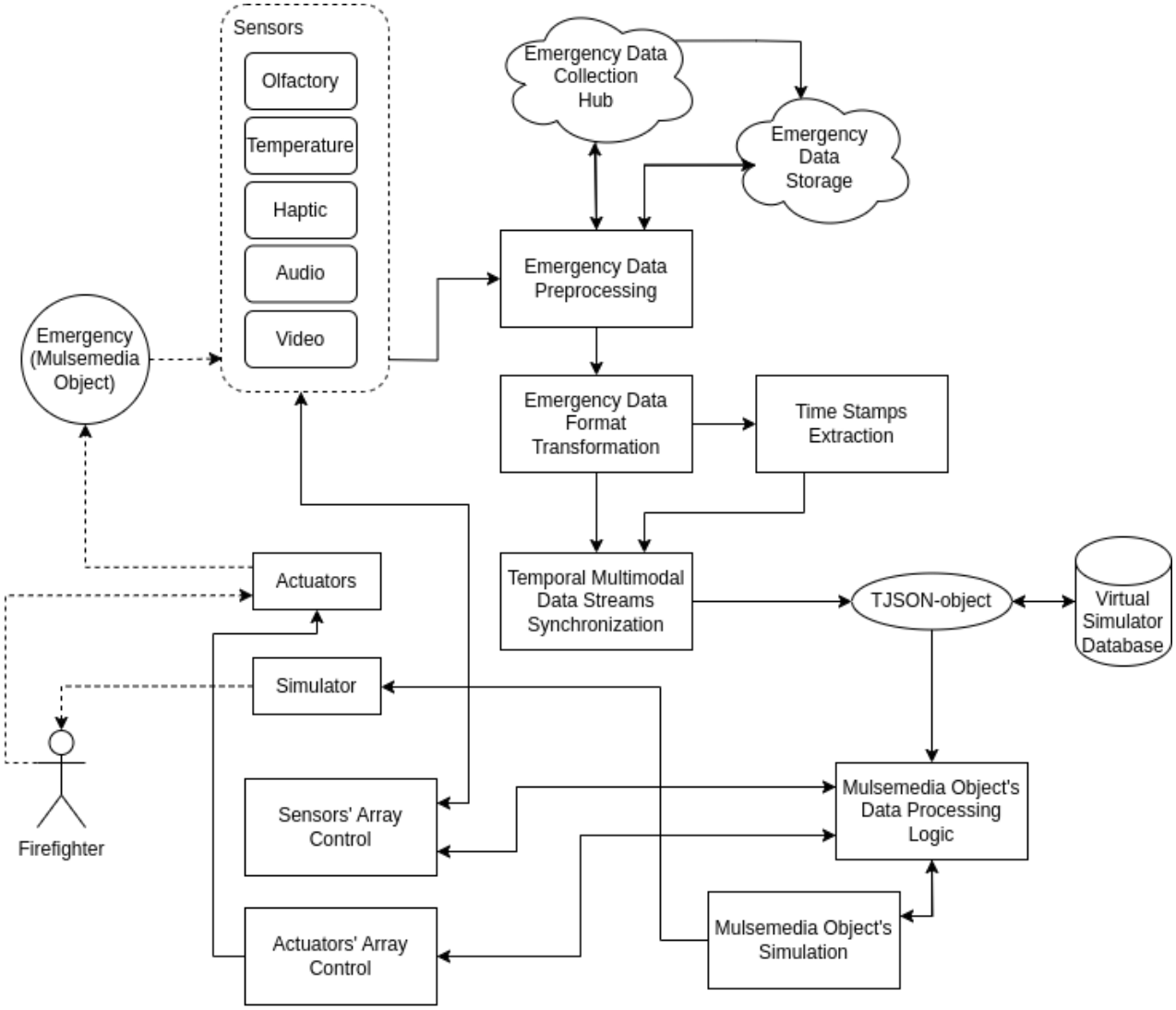

Рис. 4.3 – Архітектура програмної системи для віртуальних тренінгів рятувальників

Хаб збору даних надзвичайної ситуації (Emergency Data Collection Hub) призначений для збору даних із сенсорів, що знаходяться на одязі рятувальників, які беруть участь у ліквідації події. Ці дані передаються до модулю *передобробки даних надзвичайних ситуацій* (Emergency Data Preprocessing), конвертуються до уніфікованого формату за допомогою модулю *перетворення формату даних надзвичайних ситуацій* (Emergency Data Format Transformation). Далі різні набори мультимодальних даних

проходять синхронізацію за часом у модулі *синхронізації потоків темпоральних мультимодальних даних* (Temporal Multimodal Data Streams Synchronization).

Симулятор (Simulator) є центральним модулем, який створює інтерактивне тренувальне середовище.

Модуль *керування масивом актуаторів* (Actuators' Array Control) надає змогу впливати на різні аспекти процесу тренування для підвищення реалістичності імерсійного середовища.

Модуль *логіки обробки даних мульсемедійного об'єкту* (Mulsemedia Object's Data Processing Logic) та модуль *симуляції мульсемедійного об'єкта* (Mulsemedia Object's Simulation) реалізують симуляцію надзвичайної події та обробляють взаємодію фахівця з апаратним забезпеченням мульсемедіа.

Всі дані надалі зберігаються в *локальній базі даних віртуального симулятора* (Virtual Simulator Database) і можуть бути використані для покращення досвіду взаємодії з симулятором та для оцінки навичок користувачів.

Таким чином, запропонована система забезпечує динамічне та адаптивне тренувальне середовище, яке поєднує реалістичність надзвичайних ситуацій з безпекою віртуальної симуляції. Вона надає змогу аналізувати та вдосконалювати навички фахівців, забезпечуючи зворотний зв'язок для підвищення професійної майстерності та безпеки.

4.4 Висновки до четвертого розділу

У цьому розділі запропоновано узагальнену архітектуру програмних систем на основі технології цифрових двійників мульсемедійних об'єктів. Запропонована архітектура може бути реалізована за допомогою мови програмування ASAMPL 2.0, що забезпечує процеси обробки та відтворення мульсемедійних даних. Центральним елементом архітектури є

модуль синхронізації мультимодальних даних, що надає можливість для ефективної обробки темпоральних мультимодальних даних.

Запропоновано два приклади адаптації розробленої архітектури – для створення медичної програмної системи та імерсійної системи віртуальних тренінгів для рятувальників.

Медична програмна система передбачає обробку та аналіз медичних даних пацієнтів у реальному часі. Запропонована система може бути використана в різних медичних установах, від лікарень до амбулаторних клінік, а також для віддаленого медичного нагляду за пацієнтом.

Імерсійна система віртуальних тренінгів для рятувальників призначена для створення реалістичних симуляцій надзвичайних ситуацій, що дозволяє фахівцям набувати цінних практичних навичок у безпечному віртуальному середовищі. Можливі застосування запропонованої системи – тренування військових, спортсменів, космонавтів тощо.

Запропонована узагальнена архітектура може бути адаптована для різних сфер, де доцільно застосування концепції цифрового двійника мульсемедійного об’єкта.

ВИСНОВКИ

У дисертаційній роботі вирішено актуальну науково-технічну задачу підвищення ефективності обробки даних про стан мульсемедійних об’єктів у програмних системах на основі концепції цифрового двійника.

У результаті дисертаційного дослідження отримано такі основні наукові результати.

1. Продемонстровано, що є нагальна потреба у розробленні універсального програмного забезпечення на основі концепції цифрового двійника. Аналіз відомих програмних рішень для роботи з технологією цифрових двійників виявив, що наявні програмні системи надають можливість створювати цифрові копії фізичних об’єктів лише для певної чітко визначеної сфери. Це зумовлено відсутністю уніфікованої та стандартизованої методології розроблення цифрових двійників, що не дає змоги використати весь потенціал цієї технології, зокрема застосувати її у галузі освіти та медичній галузі.
2. Розроблено узагальнену архітектуру програмної системи для роботи з даними цифрових двійників мульсемедійних об’єктів, яка надає можливість адаптації технології цифрових двійників для застосування для вирішення задач, пов’язаних з моніторингом мульсемедійних об’єктів. Запропонована архітектура передбачає обробку наборів темпоральних мультимодальних даних, які представлені як комплексна структура даних – мультиобраз мульсемедійного об’єкта, та використання спеціалізованих програмно-апаратних засобів (сенсорів, актуаторів, симуляторів, рендерів), які призначені для забезпечення взаємодії комп’ютерної системи з мульсемедійним об’єктом.
3. Продемонстровано, що для уніфікації та стандартизації форматів агрегації та синхронізації мульсемедійних даних необхідно розробити метод консолідації мульсемедійних даних. Запропонований метод

ґрунтується на застосуванні концепції мультиобразу, кількісних відношень дискретних інтервалів, операцій алгебраїчної системи агрегатів, принципів стеганографії, дискретного вейвлет-перетворення, паралельних обчислень. Метод призначений для поєднання даних різних модальностей в єдиний цифровий об’єкт.

4. Показано, що застосування відношень дискретних інтервалів, що визначені в алгебраїчній системі агрегатів, для синхронізації наборів мульсемедійних даних не є доцільним через неможливість визначення кількісних параметрів синхронізації. Запропоновано ввести темпоральні характеристики цих даних – кількісні відношення дискретних інтервалів.
5. Запропоновано модифікацію мови програмування ASAMPL – мову ASAMPL 2.0, яка відрізняється оновленим синтаксисом, можливостю використання користувацьких функцій, паралельного програмування та логічних функцій кількісних відношень дискретних інтервалів.
6. Доведено, що оновлення синтаксису мови програмування ASAMPL дає змогу поліпшити метрики програмного коду: цикломатичну складність, обсяг програмного коду, індекс зручності підтримки програмного коду.
7. Запропоновано архітектурні шаблони проєктування «Мульсемедійний конфігуратор», «Мульсемедійний патерн», «Мульсемедійний композитор», «Мульсемедійний адаптер», які передбачають обробку даних, представлених як мультиобрази мульсемедійних об’єктів, що дає змогу спростити процеси розроблення мульсемедійного програмного забезпечення.
8. Продемонстровано, що для збереження синхронізованих наборів темпоральних мультимодальних даних доцільно використовувати єдину структуру таких даних – мультиобраз; для її збереження запропоновано розширення формату даних JSON – формат TJSON (Timeline JSON).

Застосування цього розширення передбачено у методі консолідації мульсемедійних даних та у запропонованих архітектурних шаблонах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gupta S. Airwave: Non-contact haptic feedback using air vortex rings / S. Gupta, D. Morris, S. N. Patel, D. Tan // Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing. – 2013. – P. 419–428.
2. Zou S. Investigating Individual Differences in Olfactory Adaptation to Pulse Ejection Odor Display by Scaling Olfaction Sensitivity of Intensity / S. Zou, Y. Ban, S. I. Warisawa // 2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW). – 2021. – P. 490-491.
3. Maggioni E. OWidgets: A toolkit to enable smell-based experience design / E. Maggioni, R. Cobden, M. Obrist // International Journal of Human-Computer Studies. – 2019. – Vol. 130. – P. 248-260.
4. Wang J. A conceptual design for smell based augmented reality: Case study in maintenance diagnosis / J. Wang, J. Erkoyuncu, R. Roy // Procedia CIRP. – 2018. – Vol. 78. – P. 109-114.
5. Askari F. Aroma profile of the essential oils from different parts of Pycnocycla aucherana Decne. ex Boiss / F. Askari, F. Sefidkon, Z. E. Bistgani, M. A. SOLTANİPOUR // International Journal of Secondary Metabolite. – 2023. – Vol. 10, No. 4. – P. 535-544.
6. Iwata H. Food simulator: A haptic interface for biting / H. Iwata, H. Yano, T. Uemura, T. Moriya // IEEE virtual reality 2004. – 2004. – P. 51-57.
7. Tan P. Self-Powered Gesture Recognition Wristband Enabled by Machine Learning for Full Keyboard and Multicommand Input / P. Tan, X. Han, Y. Zou, X. Qu, J. Xue, T. Li, Z. L. Wang // Advanced Materials. – 2022. – Vol. 34, No. 21. – 2200793.

8. Sarakoglou I. Hexotrac: A highly under-actuated hand exoskeleton for finger tracking and force feedback / I. Sarakoglou, A. Brygo, D. Mazzanti, N. G. Hernandez, D. G. Caldwell, N. G. Tsagarakis // 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). – 2016. – P. 1033-1040.
9. Yoshimura T. Development of a perfume emission system via internet / T. Yoshimura, Y. Sakashita // Journal of Computer Chemistry, Japan. – 2006. – Vol. 5, No. 4. – P. 227-230.
10. Murer M. LOLL io: exploring taste as playful modality / M. Murer, I. Aslan, M. Tscheligi // Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction. – 2013. – P. 299-302.
11. Nandal P. Smell-O-Vision Device // Computational Methods and Data Engineering: Proceedings of ICMDE 2020, Volume 2. – 2020. – P. 321-329. Singapore: Springer Singapore.
12. Miyashita H. Norimaki synthesizer: Taste display using ion electrophoresis in five gels // Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems. – 2020. – P. 1-6.
13. Yobas L. A novel bulk micromachined electrostatic microvalve with a curved-compliant structure applicable for a pneumatic tactile display / L. Yobas, M. A. Huff, F. J. Lisy, D. M. Durand // Journal of Microelectromechanical Systems. – 2001. – Vol. 10, No. 2. – P. 187-196.
14. Carpi F. EEG Investigation on the Tactile Perceptual Performance of a Pneumatic Wearable Display of Softness / F. Carpi, M. C. Valles, G. Frediani, T. Toci, A. Grippo // Actuators. – 2023. – Vol. 12, No. 12. – P. 431.
15. Kim Y. Design and psychophysical evaluation of pneumatic tactile display / Y. Kim, I. Oakley, J. Ryu // 2006 SICE-ICASE International Joint Conference. – 2006. – P. 1933-1938.

16. Smith C. Creating Project Sense // [Електронний ресурс]. Режим доступу: https://doi.org/10.4324/9781351153522-9. DOI: 10.4324/9781351153522-9. - 2017.
17. Chen C. A Quality of Experience Evaluation of an Interactive Multisensory 2.5 D Virtual Reality Art Exhibit / C. Chen, N. Murray, C. Keighrey // Proceedings of the 2023 ACM International Conference on Interactive Media Experiences. – 2023. – P. 170-173.
18. Sardo J. D. P. Portable device for touch, taste and smell sensations in augmented reality experiences / J. D. P. Sardo, J. Semião, J. M. Monteiro, J. A. Pereira, M. A. de Freitas, E. Esteves, J. M. Rodrigues // INCREaSE: Proceedings of the 1st International Congress on Engineering and Sustainability in the XXI Century-INCREaSE 2017. – 2018. – P. 305-320. Springer International Publishing.
19. Kunii Y. Development of 20 DOF glove type haptic interface device-Sensor Glove II / Y. Kunii, Y. Nishino, T. Kitada, H. Hashimoto // Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics. – 1997. – P. 132.
20. Matsukura H. Smelling screen: development and evaluation of an olfactory display system for presenting a virtual odor source / H. Matsukura, T. Yoneda, H. Ishida // IEEE transactions on visualization and computer graphics. – 2013. – Vol. 19, No. 4. – P. 606-615.
21. Matsukura H. Smelling screen: Technique to present a virtual odor source at an arbitrary position on a screen / H. Matsukura, T. Yoneda, H. Ishida // 2012 IEEE Virtual Reality Workshops (VRW). – 2012. – P. 127-128.
22. Shinogi R. Smelling Screen: Application to a Museum Exhibition and a Challenge for Scaling Up / R. Shinogi, H. Matsukura, H. Ishida // 2019 IEEE International Symposium on Olfaction and Electronic Nose (ISOEN). – 2019. – P. 1-3.

23. Matsukura H. Smelling screen: Presenting a virtual odor source on a LCD screen / H. Matsukura, T. Yoneda, H. Ishida // 2013 IEEE Virtual Reality (VR). – 2013. – P. 167-168.
24. Karunanayaka K. New thermal taste actuation technology for future multisensory virtual reality and internet / K. Karunanayaka, N. Johari, S. Hariri, H. Camelia, K. S. Bielawski, A. D. Cheok // IEEE transactions on visualization and computer graphics. – 2018. – Vol. 24, No. 4. – P. 1496-1505.
25. Obrist M. Touch, taste, & smell user interfaces: The future of multisensory HCI / M. Obrist, C. Velasco, C. T. Vi, N. Ranasinghe, A. Israr, A. D. Cheok, P. Gopalakrishnakone // Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems. – 2016. – P. 3285-3292.
26. Sinclair M. TouchMover 2.0-3D touchscreen with force feedback and haptic texture / M. Sinclair, M. Pahud, H. Benko // 2014 IEEE Haptics Symposium (HAPTICS). – 2014. – P. 1-6.
27. Sinclair M. TouchMover: actuated 3D touchscreen with haptic feedback / M. Sinclair, M. Pahud, H. Benko // Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces. – 2013. – P. 287-296.
28. Li W. Synthesizing personalized construction safety training scenarios for VR training / W. Li, H. Huang, T. Solomon, B. Esmaeili, L. F. Yu // IEEE Transactions on Visualization and Computer Graphics. – 2022. – Vol. 28, No. 5. – P. 1993-2002.
29. Kourtesis P. Validation of the virtual reality neuroscience questionnaire: maximum duration of immersive virtual reality sessions without the presence of pertinent adverse symptomatology / P. Kourtesis, S. Collina, L. A. Doumas, S. E. MacPherson // Frontiers in human neuroscience. – 2019. – Vol. 13. – P. 417.

30. Nakamoto T. Virtual environment with smell using wearable olfactory display and computational fluid dynamics simulation / T. Nakamoto, T. Hirasawa, Y. Hanyu // 2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR). – 2020. – P. 713-720.
31. Gelernter D. H. Mirror Worlds: or the Day Software Puts the Universe in a Shoebox – How It Will Happen and What It Will Mean // Oxford; New York: Oxford University Press. – 1991. – ISBN 978-0195079067. – OCLC 23868481.
32. Saleme E. B. Mulsemedia DIY: A survey of devices and a tutorial for building your own mulsemedia environment / E. B. Saleme, A. Covaci, G. Mesfin, C. A. Santos, G. Ghinea // ACM Computing Surveys (CSUR). – 2019. – Vol. 52, No. 3. – P. 1-29.
33. Ghinea G. Mulsemedia: State of the art, perspectives, and challenges. / Ghinea G., Timmerer C., Lin W., Gulliver S. R. // ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM). – 2014. – Vol. 11, No. 1s. – P. 1-23.
34. Grieves M. Completing the Cycle: Using PLM Information in the Sales and Service Functions [Slides]. // SME Management Forum. – 2002.
35. Glaessgen E. The digital twin paradigm for future NASA and US Air Force vehicles // In: 53rd AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference 20th AIAA/ASME/AHS adaptive structures conference 14th AIAA. – 2012. – P. 1818.
36. Grieves M. Digital twin: manufacturing excellence through virtual factory replication // White paper. – 2014. – Vol. 1(2014). – P. 1-7.
37. Sulema Ye., Rvach D. Models of computation for Digital Twins data processing // Наукові вісти КПІ. – 2020. – № 2. – С. 74–81.
38. Sulema Y. Mulsemedia vs. Multimedia: State of the art and future trends / Y. Sulema // In: 2016 International conference on systems, signals and image processing (IWSSIP). – IEEE, May 2016. – Pp. 1-5.

39. Nath S. V. Building industrial digital twins: Design, develop, and deploy digital twin solutions for real-world industries using Azure digital twins / S. V. Nath, P. Van Schalkwyk, D. Isaacs // Packt Publishing Ltd. – 2021.
40. Fei T. A. O. makeTwin: A reference architecture for digital twin software platform / Fei, T. A. O., Xuemin, S. U. N., CHENG, J., Yonghuai, Z. H. U., Weiran, L. I. U., Yong, W. A. N. G., Xiaohui, J. I. N. // Chinese Journal of Aeronautics. – 2023.
41. Hummel M. Leveraging nvidia omniverse for in situ visualization / M. Hummel, K. Van Kooten // In: High Performance Computing: ISC High Performance 2019 International Workshops, Frankfurt, Germany, June 16-20, 2019, Revised Selected Papers 34. – Springer International Publishing, 2019. – P. 634-642.
42. Ivanov D. A digital supply chain twin for managing the disruption risks and resilience in the era of Industry 4.0 / D. Ivanov, A. Dolgui // Production Planning & Control. – 2021. – Vol. 32, No. 9. – P. 775-788.
43. Wiegers K. Software Requirements / K. Wiegers, J. Beatty // Software Requirements. – 2014.
44. Chung L. Non-Functional Requirements in Software Engineering / Chung L., Nixon B. A., Yu E., Mylopoulos J. // The Kluwer International Series in Software Engineering. – Volume 5. – 1st edition.
45. Khan M. N. A. Review of requirements management issues in software development / Khan M. N. A., Khalid M., ul Haq S. // International Journal of Modern Education and Computer Science. – 2013. – Vol. 5, No. 1. – P. 21.
46. Дичка І. А. Модель подання мультимодальних даних для комплексного опису об’єктів спостереження / Дичка І. А., Сулема Є. С. // Вісник Вінницького політехнічного інституту. – 2020. – № 1. – С. 53-60.

47. Mattos D. P. An Approach for Authoring Mulsemedia Documents Based on Events/ Mattos D. P., Muchaluat-Saade D. C., Ghinea G. // Proceedings of the IEEE International Conference on Computing Networking and Communications (ICNC). – 2020. – P. 273–277.
48. Pester A. Temporal Multimodal Data-Processing Algorithms Based on Algebraic System of Aggregates / Pester A., Sulema Ye., Dychka I., Sulema O. // Algorithms. – 2023. – №16(4). – P. 186.
49. Handa M. Immersive Technology – Uses Challenges and Opportunities / Handa M., Aul G., Bajaj S. // International Journal of Computing & Business Research. – 2012.
50. Sulema Ye., Rvach D. Formal specification of mulsemedia object's digital twin based on discrete intervals temporal relations // Computer Systems and Information Technologies. – 2023. – № 4.
51. Dychka, I., Sulema, Ye., Rvach, D., Drozdenko, L. Programming Language ASAMPL 2.0 for Mulsemedia Applications Development // Engineering and Education Applications. – 2022. – №134. – P. 107-116.
52. Dychka, I. Logical operations in algebraic system of aggregates for multimodal data representation and processing / Dychka, I. A., Sulema Ye. // KPI Sciense News. – 2018. – №6. – P. 44-52.
53. Sulema Ye. Multimodal data processing based on algebraic system of aggregates relations // Radio Electronics, Computer Science. – 2020. – №1. – P. 169-180.
54. Milner R. Operational and Algebraic Semantics of Concurrent Processes. Formal Models and Semantics. – 1990. – P. 1203–1242.
55. Roşu G. Towards a Unified Theory of Operational and Axiomatic Semantics / Roşu G., Ştefănescu A. // Automata Languages and Programming. Springer. – 2012. –. P. 351-363.
56. Isah A. Digital Twins Temporal Dependencies-Based on Time Series Using Multivariate Long Short-Term Memory / Isah A., Shin H., Oh S.,

Oh S., Aliyu I., Um T. W., Kim J. // Electronics. – 2023. – №. 12(19). – P. 4187.

57. Pezoa F., et al. Foundations of JSON schema // Proceedings of the 25th international conference on World Wide Web. – 2016. – С. 263-273.

58. Dmytro R., Sulema Ye. Mulsemedia data consolidation method. // System technologies. – 2022. – Vol. 6, No. 143. – P. 69-79.

59. Sharing and Experiencing Hardware and Methods to Advance Smell, Taste, and Temperature Interfaces // In Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems (CHI EA '23), April 23-28, 2023, Hamburg, Germany. – ACM, New York, NY, USA. – 9 Pages.

60. Taha M. S. Combination of Steganography and Cryptography: A short Survey / Taha M. S., Rahim M. S., Lafta S. A., Hashim M. M., Alzuabidi H. M. // IOP Conference Series: Materials Science and Engineering. – 2019. – № 518. – 052003.

61. Dutta S. Securing data: A study on different transform domain techniques. / Dutta S., Saini K. // WSEAS Transactions on Systems And Control. – 2021. – Vol. 16, No. 37394.

62. Debnath, D. MultiImage Hiding Blind Robust RGB Steganography in Transform Domain / Debnath, D., Ghosh, E., Banik, B. G. // International Journal of WebBased Learning and Teaching Technologies (IJWLTT). – 2020. – №15(1). – P. 24-52.

63. Hemachandran K. Study of Image Steganography using LSB, DFT and DWT / K. Hemachandran // International Journal of Computers & Technology. – 2016. – Vol. 11. – P. 2618-2627.

64. Kavitha K. K. Steganography using least significant bit algorithm / K. K. Kavitha, A. Koshti, P. Dunghav // International Journal of Engineering Research and Applications (IJERA). – 2012. – P. 2248-9622.

65. ALabaichi A. Image steganography using least significant bit and secret map techniques / A. ALabaichi, M. A. A. A. K. Al-Dabbas, A. Salih // International journal of electrical & computer engineering (2088-8708). – 2020. – Vol. 10, No. 1.
66. Rabie T. High-capacity steganography: a global-adaptive-region discrete cosine transform approach / T. Rabie, I. Kamel // Multimedia Tools and Applications. – 2017. – Vol. 76. – P. 6473-6493.
67. Hashad A. I. A robust steganography technique using discrete cosine transform insertion / A. I. Hashad, A. S. Madani, A. E. M. A. Wahdan // 2005 International Conference on Information and Communication Technology. – 2005, December. – P. 255-264.
68. Soni A. Image steganography using discrete fractional Fourier transform / A. Soni, J. Jain, R. Roshan // 2013 International Conference on Intelligent Systems and Signal Processing (ISSP). – 2013, March. – P. 97-100.
69. Mandal J. K. Discrete Fourier transform-based steganography // Reversible Steganography and Authentication via Transform Encoding. – 2020. – P. 63-98.
70. Reddy H. M. High capacity and security steganography using discrete wavelet transform / H. M. Reddy, K. B. Raja // International Journal of Computer Science and Security (IJCSS). – 2009. – Vol. 3, No. 6. – P. 462-472.
71. Narasimmalou T. Optimized discrete wavelet transform based steganography / T. Narasimmalou, J. R. Allen // 2012 IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT). – 2012, August. – P. 88-91.
72. Al-Korbi H. A. High-capacity image steganography based on Haar DWT for hiding miscellaneous data. / Al-Korbi H. A., Al-Ataby A., Al-Taee M. A., Al-Nuaimy W. // In 2015 IEEE Jordan Conference on Applied

Electrical Engineering and Computing Technologies (AEECT). – 2015, November. – P. 1-6.

73. Houssein E. H. An image steganography algorithm using haar discrete wavelet transform with advanced encryption system / E. H. Houssein, M. A. Ali, A. E. Hassanien // 2016 Federated Conference on Computer Science and Information Systems (FedCSIS). – 2016, September. – P. 641-644.
74. Islam M. R. Wavelets, its Application and Technique in signal and image processing // Global Journal of Computer Science and Technology. – 2011. – Vol. 11, No. 4.
75. Antonini M. Image coding using wavelet transform / M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies // IEEE Trans. Image Processing. - 1992. – Vol. 1. – P. 20-5.
76. Hore A. Image quality metrics: PSNR vs. SSIM. / Hore A., Ziou D. // In 2010 20th International Conference on Pattern Recognition. – 2010, August. – P. 2366-2369.
77. Tanchenko A. Visual-PSNR measure of image quality / A. Tanchenko // Journal of Visual Communication and Image Representation. – 2014. – Vol. 25, No. 5. – P. 874-878.
78. Ndajah P. SSIM image quality metric for denoised images / P. Ndajah, H. Kikuchi, M. Yukawa, H. Watanabe, S. Muramatsu // Proc. 3rd WSEAS Int. Conf. on Visualization, Imaging and Simulation. – 2010, November. – P. 53-58.
79. Gelissen, J. H. Introduction to mpeg-v // Journal For Virtual Worlds Research. – 2009. – № 3 (2).
80. Yoon K. MPEG-V: Bridging the virtual and real world / K. Yoon, S. K. Kim, J. J. Han, S. Han, M. Preda // Academic Press. – 2015.
81. Oh H. W. Annotation creation and representation system of sensory effects based on MPEG-V / H. W. Oh, J. K. Yun, J. D. Huh // The 18th

IEEE International Symposium on Consumer Electronics (ISCE 2014). – 2014, June. – P. 1-2.

82. Timmerer, C. Interfacing with virtual worlds / Timmerer, C., Gelissen, J., Waltl, M., Hellwagner, H. // Network and Electronic Media Summit. – 2009.

83. Waltl M. A test-bed for quality of multimedia experience evaluation of sensory effects / M. Waltl, C. Timmerer, H. Hellwagner // 2009 International Workshop on Quality of Multimedia Experience. – IEEE, 2009. – P. 145-150.

84. Cho M.-S., Kim J.-S. Method and apparatus for providing metadata for sensory effect, computer readable record medium on which metadata for sensory effect is recorded, method and apparatus for representating sensory effect : пат. 8,675,010 США. – № 8,675,010; заявл. 18.03.2014; опубл. 18.03.2014.

85. Waltl M. An end-to-end tool chain for Sensory Experience based on MPEG-V / M. Waltl, B. Rainer, C. Timmerer, H. Hellwagner // Signal Processing: Image Communication. – 2013. – Vol. 28, No. 2. – P. 136-150.

86. Shin S. H. Realistic media authoring tool based on MPEG-V international standard / S. H. Shin, K. S. Ha, H. O. Yun, Y. S. Nam // 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN). – 2016, July. – P. 730-732.

87. Josué M. Modeling sensory effects as first-class entities in multimedia applications / M. Josué, R. Abreu, F. Barreto, D. Mattos, G. Amorim, J. dos Santos, D. Muchaluat-Saade // Proceedings of the 9th ACM Multimedia Systems Conference. – 2018, June. – P. 225-236.

88. Yoon K. 4-d broadcasting with mpeg-v / K. Yoon, B. Choi, E. S. Lee, T. B. Lim // 2010 IEEE International Workshop on Multimedia Signal Processing. – 2010, October. – P. 257-262.

89. Bray T. Extensible markup language (XML) / T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau // World Wide Web Journal. – 1997. – Vol. 2, No. 4. – P. 27-66.
90. Sulema Ye. ASAMPL: Programming language for mulsemedia data processing based on algebraic system of aggregates // Interactive Mobile Communication Technologies and Learning: Proceedings of the 11th IMCL Conference. – Springer International Publishing, 2018. – P. 431-442.
91. Sulema Ye. Semantics and pragmatics of programming language ASAMPL / Sulema Ye., Glinskii V. // Проблеми програмування. – 2020. – P. 74-83. ISO/IEC 14977:1996
92. Information technology – Syntactic metalanguage – Extended BNF. https://www.iso.org/standard/26153.html
93. Nutaro J. Race conditions and data partitioning: risks posed by common errors to reproducible parallel simulations / J. Nutaro, O. Ozmen // Simulation. – 2023. – Vol. 99, No. 4. – P. 417-427.
94. Netzer R. H. What are race conditions? Some issues and formalizations / R. H. Netzer, B. P. Miller // ACM Letters on Programming Languages and Systems (LOPLAS). – 1992. – Vol. 1, No. 1. – P. 74-88.
95. Erciyes K. Parallel and distributed computing / K. Erciyes, K. Erciyes // Distributed and Sequential Algorithms for Bioinformatics. – 2015. – P. 51-77.
96. Cicirelli F. An approach to concurrent/parallel programming in Java / F. Cicirelli, C. Nigro, L. Nigro // 2015 IEEE 13th International Scientific Conference on Informatics. - 2015, November. - P. 67-72.
97. Code metrics values. Microsoft, 2018. https://docs.microsoft.com/en-us/visualstudio/code-quality/code-metrics-values?view=vs-2019

98. Halstead M. H. Software Complexity Measure. / Halstead M. H., McCabe T. A. // IEEE Transactions on Software Engineering. – 1976. – Vol. 2, No. 12. – P. 308–320.
99. Maintainability Index Rangeand Meaning. Microsoft, 2007. https://docs.microsoft.com/en-us/archive/blogs/codeanalysis/maintainability-index-range-and-meaning
100. Gamma E. Design patterns: Elements of reusable object-oriented software / Gamma E., Helm R., Johnson R., Vlissides J. // Addison-Wesley. – 1994.
101. Sulema Ye., Rvach D. High-level architecture of mulsemedia software system // Вісник Хмельницького національного університету Серія: «Технічні науки». – 2023. – № 5 Т2 (143). – P. 121-125.
102. Горбачов Ю. В., Пастернак І. І. Програмна система для пошуку медикаментів у режимі онлайн // Комп'ютерні системи та мережі. – 2021. – Вип. 3, № 1. – P. 29-37. – DOI : 10.23939/csn2021.01.029.

ДОДАТКИ

Додаток А. Апаратні засоби мульсемедіа

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
AirWave (Microsoft) [1]	Використовує повітря для створення відчуття дотику. Цей пристрій здатен симулювати різні тактильні відчуття, такі як тиск, вібрацію та інші форми фізичного взаємодії. Ця технологія може знайти застосування у різноманітних сферах, включаючи навчання, медицину та інженерію, надаючи нові можливості для інтерактивної та мультимодальної взаємодії з цифровим контентом.	Тактильне
AromaPlayer (AromaJoin) [2]	Інструмент, який синхронізує запахи з відеоконтентом за допомогою пристрою AromaShooter. Він дозволяє користувачам створювати, редагувати та відтворювати смуги запахів у часовій лінійці ароматів, а також зберігати та ділитися своїми роботами в хмарі. Пристрій підключається до комп'ютерів на базі Windows або Mac	Запах

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	OS через Bluetooth або USB-з'єднання. AromaPlayer використовується для створення більш імерсивного відеодосвіду, дозволяючи користу-вачам насолоджуватися відео з дода-ванням реалістичних запахів, які син-хронізовані з візуальним контентом.	
AromaShooter (AromaJoin) [3]	Пристрій для дифузії ароматів, який здатний випромінювати запахи на відстані до 2 метрів. Цей пристрій використовується для створення інтенсивних і реалістичних ароматичних вражень у віртуальній реальності, цифровому маркетингу та інших застосуваннях. AromaShooter працює, засмоктуючи навколишнє повітря, яке далі проходить через картридж, додаючи запах до повітря. Форсунки пристрою спрямовані на користувача, доставляючи запах. Проєкт Aroma Shooter орієнтований на створення індивідуальних ароматичних досвідів і може підключатися до смартфонів,	Запах

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	комп'ютерів або гарнітур віртуальної реальності через Bluetooth, що відкриває можливості для створення інтерактивних та імерсійних застосунків з використанням запахів.	
Feelreal Mask [4, 5]	Є прототипом багатофункціональної маски, яка розроблена для створення максимально імерсійного віртуального досвіду шляхом симуляції різних сенсорних відчуттів. Маска обладнана механізмами для імітації температурних умов, вологих та теплих середовищ, використовуючи складну комбінацію розпилювачів, нагрівачів, охолоджувачів та генератора запахів. Основна мета Feelreal Mask полягає в підвищенні реалістичності віртуальних середовищ шляхом додавання фізичних відчуттів. Ця технологія може знайти застосування у різноманітних сферах, включаючи ігрову індустрію, симуляції тренувань та терапевтичні програми.	Тактильне, Запах

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
Food Simulator [6]	Пристрій призначений для імітації відчуттів, пов'язаних з їжею, включаючи текстуру та смак. Він використовує сенсори для вимірювання характеристик жування реальних продуктів, таких як сила, необхідна для прикушування шматка їжі. Для симуляції їжі пристрій вводиться до рота користувача. Він обладнаний сенсорами, які реєструють силу жування, а двигун забезпечує відповідний опір. Для доповнення враження тонка трубка впорскує на язик суміш ароматизаторів, які стимулюють основні смакові відчуття: солодке, кисле, гірке, солоне та умамі. Також використовується мініатюрний динамік для відтворення звуку від процесу взаємодії з їжею (жування).	Смак, Тактильне
H2L Technologies Wristband [7]	Пристрій розроблений японським стартапом H2L Technologies за підтримки Sony. Це зап'ястний браслет, який використовує	Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	електричну стимуляцію для імітації ваги, тиску та навіть болю, щоб збільшити реалістичність віртуальних відчуттів. Застосування цієї технології може допомогти створити більш правдоподібні віртуальні реальності, зокрема у тренувальних симуляціях. Можливість імітації болю, хоча і є суперечливою, може бути корисна у тренінгах військових та рятувальників.	
HEXOTRAC [8]	Пристрій у формі екзоскелету для руки, що симулює тактильні відчуття. Він призначений для імітації відчуття дотику, тиску та інших фізичних взаємодій. Пристрій має спеціальні кріплення до кінчиків пальців і складається з кінематичного ланцюга з шестьма ступенями свободи. Цей пристрій може бути використаний у різноманітних застосуваннях, включаючи тренування, реабілітацію, віртуальну реальність та доповнену реальність, де він надає користувачам	Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	можливість відчувати віртуальні об'єкти з більшою реалістичністю.	
iSmell [9]	Прототип персонального синтезатора ароматів, створений для використання з комп'ютерами через USB або COM-порт. Цей пристрій розроблений для відтворення цифрових запахів, тим самим збагачуючи користувацький досвід при перегляді веб-сайтів або взаємодії з цифровим контентом. Пристрій створює аромати, використовуючи набір хімічних компонентів, які змішуються та розподіляються залежно від потреби.	Запах
LOLLio [10]	Пристрій є інтерактивним цифровим «льодяником», який динамічно змінює свій смак та використовується як вхідний пристрій для ігрових застосунків. Основна ідея застосування пристрою полягає у створенні нових, захопливих ігрових користувацьких досвідів шляхом	Смак

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	інтеграції відчутя смаку як частини інтерактивності. Цей пристрій є прикладом використання смаку як модальності ігрової взаємодії та відкриває нові можливості в області ігрового дизайну та цифрових розваг. Пристрій розроблений з метою дослідження потенціалу смаку як нової форми взаємодії у відеоіграх.	
Nose-Zapping Wearable [11]	Пристрій реалізує електричну стимуляцію для імітації запахів. Він не використовує реальних хімічних речовин для створення запахів. Замість цього пристрій стимулює нервові закінчення в носі, що відповідають за відчуття запаху, створюючи ілюзію різних ароматів. Носимий пристрій працює за допомогою Bluetooth та впливає на так звану тригемінальну нервову систему, що надає змогу мозку користувача реєструвати конкретні нюхові відчуття.	Запах

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
Norimaki Synthesizer [12]	Пристрій, який імітує різні смаки за допомогою електрофорезу, використовуючи гелі з електролітами. Цей синтезатор смаку надає змогу користувачам контролювати інтенсивність різних смаків: солоного, солодкого, гіркого, кислого та умамі. Пристрій відкриває нові можливості для симуляції смаків у віртуальних середовищах, надаючи змогу користувачам зазнавати різних смакових відчуттів, наприклад, під час віртуальних вечерь або візитів у віртуальні кафе.	Смак
Pneumatic Tactile Displays [13, 14]	Тип пристроїв, які використовують повітряні камери між особливими шарами прозорого акрилу та латексу для створення простих форм та кнопок. Ця технологія надає змогу імітувати різні тактильні відчуття, такі як тиск або текстура. Пневматичні тактильні дисплеї використовуються в дослідницьких та академічних цілях	Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	для розроблення інтерактивних інтерфейсів та різноманітних додатків, зокрема в галузі віртуальної реальності та доповненої реальності, що вимагають більшої реалістичності тактильних відчуттів.	
Project Sense [15, 16]	Пристрій призначений для забезпечення користувачів відчуттям дотику та запаху у віртуальному середовищі. Проєкт передбачає використання тактильного рукава, який нагрівається, та пристрою для відтворення смаку та запаху. Система стимулює нервові рецептори в руці, відтворюючи такі властивості об'єкта, як тиск, температуру, шорсткість, м'якість або твердість. Крім того, пристрій може використовуватися для допомоги незрячим особам читати текст на екрані, перетворюючи його на шрифт Брайля. Проєкт також передбачає використання шести картриджів для імітації основних типів смаків, а також сімох воскових	Запах, Смак, Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	картриджів з мікроеміттерами для відтворення різних типів запахів.	
Sensiks Sensory Reality Pods [17, 18]	Кабіни, що надають повністю імерсійний досвід віртуальної реальності. Вони обладнані системами, що імітують вітер, світло і тепло, доповнюючи візуальні відчуття від гарнітури віртуальної реальності. Ці кабіни використовуються в медичних установах для відновлення після травм, лікування пацієнтів з розладами психіки та для допомоги літнім людям.	Запах, Смак, Тактильне
Sensor Glove [19]	Багатофункціональний тактильний пристрій, який має 20 ступенів свободи і призначений для симуляції відчуття дотику. Цей пристрій використовується для досліджень і прототипування в області віртуальної реальності та телеробототехніки. Sensor Glove надає змогу користувачам відчувати і	Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	маніпулювати віртуальними об'єктами, відтворюючи тактильні відчуття, такі як текстура, температура та вага об'єктів, у віртуальному середовищі. Ця технологія відіграє важливу роль у розробленні більш інтуїтивних інтерфейсів для взаємодії з віртуальними середовищами і може застосовуватися у різноманітних сферах, включаючи освіту, медицину та інженерію.	
Smelling Screen [20-23]	Пристрій, який поєднує візуалізацію на екрані з можливістю відтворення запахів. Використовуючи чотири вентилятори, цей екран здатен доносити аромати до користувача, створюючи більш імерсійне мультисенсорне відчуття. Ця технологія особливо ефективна у сферах, де необхідно доповнити візуальну інформацію запахами, наприклад, у рекламі, навчанні або розвагах. Таке поєднання запахів і	Запах

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	візуального контенту відкриває нові перспективи для створення унікальних і запам'ятовуваних відчуттів.	
Taste+ [24]	Пристрій посилює смакові відчуття за допомогою електричних імпульсів. Він використовує слабкі та контрольовані електричні імпульси на язиці для посилення смакових відчуттів від їжі та напоїв без додавання додаткових смакових інгредієнтів. Taste+ складається з двох прототипів посуду – пляшки та ложки; обидва оснащені електронними модулями управління, які застосовують контрольовані електричні імпульси на язик через срібні електроди. Цей пристрій надає змогу підсилювати такі смаки, як кислий, солоний та гіркий, що розширює можливості цифрового відтворення смакових відчуттів.	Смак

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
Thermal Taste Technology [25]	Технологія спрямована на створення смакових відчуттів у віртуальній реальності за допомогою термальної стимуляції (тепла). Ця технологія передбачає поєднання з іншими цифровими технологіями відтворення смаку та запаху для створення складних віртуальних смакових відчуттів. Ідея полягає в тому, щоб користувачі віртуальної реальності могли насолоджуватися улюбленими стравами без зайвих калорій, забезпечуючи більш реалістичне віртуальне споживання їжі.	Смак
TouchMover (Microsoft Research) [26]	Пристрій використовується для симуляції тактильних відчуттів, зокрема текстур. TouchMover призначений для створення відчуття руху та тиску на шкірі користувача, імітуючи реальні дотики та текстури. Цей пристрій може відігравати значну роль у розробленні інтерактивних інтерфейсів та застосунків на основі	Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	віртуальної реальності для освіти та у медичних симуляціях, надаючи користувачам можливість відчувати реалістичні фізичні взаємодії з віртуальними об'єктами.	
Vaqso VR [27-29]	Компактний аксесуар для віртуальної реальності, який здатний випромінювати кілька запахів одночасно, значно підвищуючи рівень занурення до віртуальної реальності. Цей пристрій є сумісним з такими системами віртуальної реальності, як PSVR, Oculus Rift та HTC Vive. Vaqso VR забезпечує більш реалістичне відчуття віртуальної реальності, надаючи змогу користувачам відчувати запахи, які відповідають візуальним відчуттям.	Запах
Wearable Olfactory Display [30]	Носимий пристрій призначений для відтворення різних запахів. Цей пристрій використовує мікроелектричні насоси, які	Запах

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	створюють потік повітря через фільтр з ароматом, вивільняючи його поблизу обличчя користувача. Основна ідея полягає в забезпеченні користувачів віртуальної реальності можливістю відчувати відповідні запахи, що підсилює занурення у віртуальне середовище і створює більш реалістичний досвід. Ця технологія може знайти застосування у різноманітних сферах, включаючи ігри, навчання, маркетинг та медичні застосування, де відтворення запахів може відігравати ключову роль у створенні багатовимірного користувацького досвіду.	
WindCube [31]	Пристрій, який використовує потік повітря для симуляції відчуття дотику. Цей пристрій може використовуватись для відтворення вітру, створюючи відчуття руху повітря, що робить його корисним у віртуальних середовищах та застосуваннях доповненої реальності. Ця технологія забезпечує	Тактильне

Назва пристрою (технології)	Короткий опис	Сенсорне відчуття
	користувачів унікальними відчуттями, які допомагають створити більш реалістичне імерсійне середовище, особливо у застосуваннях, що вимагають репродукції натуралістичних погодних умов.	

Додаток Б. Синтаксичні правила для оновленого синтаксису мови програмування ASAMPL 2.0, згідно з розширеною формою Бекуса-Наура

Оператор виходу з циклічних функцій [SR13]:

оператор_обриву = break , ";" [SR13]

Імпорт об’єктів [SR14]:

імпорт = import , *тип_об’єкту* , *"."* , *назва_об’єкту* , [as , *ідентифікатор*] , ";" [SR14]

Декларування джерел [SR15].

джерело = Source , *ідентифікатор* , "=" , *посилання* , ";" [SR15]

Декларування множин даних [SR16]:

множина_даних = Set , *ідентифікатор* , "=" , *тип* , ";" [SR16]

Декларування кортежів даних [SR17]:

кортеж = *тип_множини* , "[]" , *ідентифікатор* , ";" [SR17]

Декларування агрегатів [SR18]:

агрегат = Aggregate , *ідентифікатор* , "=" , "[" , *кортеж_значень* , "]" , ";" [SR18]

Завантаження ресурсів [SR19]:

оператор_завантаження = *ідентифікатор* , "=" , download , [SR19]
"(" , *ідентифікатор* , "," , *ідентифікатор* , ["," , *ідентифікатор* ,] , ";"

Вивантаження ресурсів [SR20]:

оператор_вивантаження = upload , "(" , *ідентифікатор* , "," [SR20]
ідентифікатор , ["," , *ідентифікатор* ,] , ";"

Заміна ресурсів [SR21]:

оператор_заміни = substitute , "(" , *ідентифікатор* , [SR21]
ідентифікатор , ")" , ";"

Відтворення мульсемедійного об’єкту [SR22]:

оператор_відтворення = render , "(" , *ідентифікатор* , [SR22]
"[" , *ідентифікатор* , "]" , ")" , ";"

Оператор розгалуження [SR23]:

оператор_розгалуження = if , "(" , *логічний_вираз* , ")" , [SR23]
"{" , *оператор* , "}" , [else , "(" , *логічний_вираз* , ")" , "{" , *оператор* , "}"

Приведення типів [SR24]:

приведення_типів = *ідентифікатор* , "=" , [SR24]
тип_даних , "(" , *значення* , ")" , ";"

Додаток В. Код програми для реалізації першої задачі.

Лістинг В.1. Реалізація першої задачі за допомогою ASAMPL

```
Program VRsportTranslation {
      Libraries {
            VisualLib IS ‘D:\MultiImage\Library\vsl.lib’;
            AudioLib IS ‘D:\MultiImage\Library\aud.lib’;
            GyroLib IS ‘D:\MultiImage\Library\gyr.lib’;
      }
      Handlers {
            MPEG2tuple IS ‘D:\MultiImage\Handler\mpg2tup.exe’;
            Tuple2MPEG IS ‘D:\MultiImage\Handler\tup2mpg.exe’;
      }
      Renders {
            VisualRen IS ‘C:\Renderer\vslren.exe’;
            AudioRen IS ‘C:\ Renderer\audren.exe’;
            GyroRen IS ‘C:\Renderer\gyrren.exe’;
      }
      Sources {
            VideoStream1 IS ‘http://vr-stream.com/003402/left_side’;
            VideoStream2 IS ‘http://vr-stream.com/003402/left_side’;
            GyroData IS ‘COM3’;
            VRdevice IS ‘COM4’;
            NetworkAdapter IS ‘lan0’;
      }
      Sets {
            Frame IS Integer(1920,1080);
            FrameLow IS Integer(1280,720);
            Audio IS Real;
            Gyro IS Integer;
      }
      Elements {
            duration IS Time;
```

```
            Time time1 = 00:00:01;
            Time time2 = 00:30:00;
            Integer step = 1;
            Real networkSpeed = NetworkAdapter.speed;
      }
      Tuples {
            VisualDat1 Is Frame;
            VisualDat2 Is Frame;
            AudioDat1 Is Audio;
            AudioDat2 Is Audio;
            GyroDat Is Gyro;
      }
      Aggregates {
            VRstream = [VisualDat1, VisualDat2, AudioDat1, AudioDat2,
GyroDat];
      }
      Actions {
            IF (networkSpeed < 5000) {
                  Frame = FrameLow;
            }
            Timeline time1 : step : time2 {
                  Download AudioDat1 From VideoStream1.audio
                  With MPEG2tuple;
                  Download VisualDat1 From VideoStream1.visual
                        With MPEG2tuple;
                  Download AudioDat2 From VideoStream2.audio
                        With MPEG2tuple;
                  Download VisualDat2 From VideoStream2.visual
                        With MPEG2tuple;
                  Download GyroDat From GyroData
                        With default.GyroLib;
            }
            Upload VRstream To VRdevice With default.all;
```

```
            Render VRstream
                  With [VisualRen, AudioRen, GyroRen];
      }
}
```

Лістинг В.2. Реалізація першої задачі за допомогою ASAMPL 2.0.

```
import libraries.video.VisualLib;
import libraries.audio.AudioLib;
import libraries.gyro.GyroLib;
import handlers.mpeg2tup as MPEG2tuple;
import handlers.tup2mpeg as Tuple2MPEG;
import renders.video.VisualRen;
import renders.audio.AudioRen;
import renders.gyro.GyroRen;
Source VideoStream1 = 'http://vr-stream.com/003402/left_side';
Source VideoStream1 = 'http://vr-stream.com/003402/right_side';
Source GyroData = ‘/dev/gyroscope01’;
Source VRdevice = ‘/dev/vr1/’;
Source NetworkAdapter = ‘lan0’;
Set Frame = int[1920, 1080];
Set FrameLow = int[1280, 720];
Set Audio = float();
Set Gyro = [];
Time time1 = 00:00:01;
Time time2 = 00:30:00;
int step = 1;
float networkSpeed = NetworkAdapter.speed;
if (networkSpeed < 5000) {
      Frame = FrameLow;
}
Frame[] VisualDat1;
Frame[] VisualDat2;
Audio[] AudioDat1;
```

```
Audio[] AudioDat2;
Gyro[] GyroDat;
Agregate VRstream = [VisualDat1, VisualDat2, AudioDat1, AudioDat2,
GyroDat];
timeline(time1, time2, step) {
      VisualDat1 = download(VisualDataStream1, default.VisualLib);
      VisualDat2 = download(VisualDataStream2, default.VisualLib);
      AudioDat1 = download(VisualDataStream1, MPEG2tuple);
      AudioDat2 = download(VisualDataStream2, MPEG2tuple);
      GyroDat = download(GyroData, default.GyroLib);
}
upload(VRstream, VRdevice, default.all);
render(VRstream, [VisualRen, AudioRen, GyroRen]);
```

Додаток Г. Код програми для реалізації другої задачі.

Лістинг Г.1. Реалізація другої задачі за допомогою ASAMPL

```
Program MedicalAnalysis {
    Libraries {
        MedicalLib Is 'path\to\medical_library.lib';
        LabResultsHandler Is 'path\to\lab_results_handler.exe';
        VisualRender IS 'path\to\visual_render.exe';
    }
    Handlers {
        MedicalRecordsHandler IS
'path\to\medical_records_handler.exe';
        LabDataHandler IS 'path\to\lab_data_handler.exe';
        MriImageHandler IS 'path\to\mri_image_handler.exe';
    }
    Renderers {
        VisualRen IS 'C:\Renderer\vslren.exe';
        AudioRen IS 'C:\Renderer\audren.exe';
        TempRen IS 'C:\Renderer\temprender.exe';
    }
    Sources {
        MedicalRecordsSource IS 'path\to\medical_records.db';
        MedicalChildRecordsSource IS
  'path\to\medical_child_records.db';
        LabResultsSource IS 'path\to\lab_results.csv';
   LocalLabResultsSource IS 'path\to\local_lab_results.csv';
        MriImagesSource IS 'path\to\mri_images_folder\';
        PatientTypeSource IS 'path\to\patient\type';
    }
    Sets {
        PatientData IS Text;
        LabData IS Real;
```

```
        ImageData IS Image;
    }
    Elements {
        duration IS Time;
        time1 = 00:00:01;
        time2 = 00:15:00;
        step = 00:00:01;
        patientType = '';
    }
    Tuples {
        MedicalRecordsTuple IS PatientData;
        LabResultsTuple IS LabData;
        MriImagesTuple IS ImageData;
    }
    Aggregates {
        DigitalTwin = [MedicalRecordsTuple, LabResultsTuple,
MriImagesTuple];
    }
    Actions {
        TIMELINE time1 : step : time2 {
           Download patientType From PatientTypeSource With
default.all;
           IF (patientType IS “adult”) {
                Download MedicalRecordsTuple From
                MedicalRecordsSource With MedicalRecordsHandler;
           } ELSE {
                Download MedicalCRecordsTuple From
                MedicalChildRecordsSource With
                MedicalRecordsHandler;

           }

           IF (LabResultSource != Null) {
```

```
                Download LabResultsTuple From LabResultsSource With
                LabDataHandler;
            } ELSE {
                Download LocalLabResultsTuple From LabResultsSource
                With LabDataHandler;
            }
          Download MriImagesTuple From MriImagesSource With
MriImageHandler;
        }
        UPLOAD DigitalTwin To 'path\to\digital_twin.dtw' With
default.all;
        Render DigitalTwin With [VisualRen, AudioRen, TempRen];
    }
}
```

Лістинг Г.2. Реалізація другої задачі за допомогою ASAMPL 2.0

```
import libraries.medical.MedicalLib;
import libraries.labresults.LabResultsHandler;
import libraries.visual.VisualRender;

import handlers.medicalrecords as MedicalRecordsHandler;
import handlers.labdata as LabDataHandler;
import handlers.mriimages as MriImageHandler;
import renders.visual.VisualRen;
import renders.audio.AudioRen;
import renders.temperature.TempRen;

Source PatientTypeSource = 'path/to/patient/type';
Source MedicalRecordsSource = 'path/to/medical_records.db';
Source MedicalChildRecordsSource =
'path/to/medical_child_records.db';
Source LabResultsSource = 'path/to/lab_results.csv';
Source LocalLabResultsSource = 'path/to/local_lab_results.csv';
```

```
Source MriImagesSource = 'path/to/mri_images_folder/';

Set PatientData = str();
Set LabData = float();
Set ImageData = image();

Time time1 = 00:00:01;
Time time2 = 00:15:00;
int step = 1;
String patientType = “”;

PatientData[] MedicalData;
LabData[] LaboratoryData;
ImageData[] MriImages;

Aggregate DigitalTwin = [MedicalData, LaboratoryData, MriImages];

timeline(time1, time2, step) {
      patientType = download(PatientTypeSource, default.all);

    if (patientType == "adult") {
             MedicalData = download(MedicalRecordsSource,
      MedicalRecordsHandler);
    } else if (patientType == "child") {
             MedicalData = download(MedicalChildRecordsSource,
      MedicalRecordsHandler);
         }

    if (LabResultsSource != null) {
            LaboratoryData = download(LabResultsSource,
      LabDataHandler);
    } else {
```

```
            LaboratoryData = download(LocalLabResultsSource,
     LabDataHandler);
    }
     MedicalData = download(MedicalRecordsSource,
MedicalRecordsHandler);
    LaboratoryData = download(LabResultsSource, LabDataHandler);
    MriImages = download(MriImagesSource, default.MriImageHandler);
}

upload(DigitalTwin, 'path/to/digital_twin.dtw', default.all);
render(DigitalTwin, [VisualRen, AudioRen, TempRen]);
```