

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

ПЕСЧАНСЬКИЙ ВЛАДИСЛАВ ЮРІЙОВИЧ

УДК 004.62 : 004.043

ДИСЕРТАЦІЯ

АЛГОРИТМІЧНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ СТВОРЕННЯ ЦИФРОВИХ ДВІЙНИКІВ МЕДИКО-БІОЛОГІЧНИХ 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

_____ В. Ю. Песчанський
(підпис)

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

Київ – 2024

АНОТАЦІЯ

Песчанський В.Ю. Алгоритмічне та програмне забезпечення технології цифрових двійників медико-біологічних 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 набувають дедалі більшого поширення у сфері медицини, оскільки вони дозволяють створити динамічну цифрову модель реального медико-біологічного об'єкта, відображаючи як анатомічні, так і функціональні аспекти його роботи. Ця технологія передбачає інтеграцію даних із різних джерел (медичні зображення, відеоендоскопія, аудіодані, сенсорні показники) у єдину інформаційну структуру, що описує стан і динаміку об'єкта. Цифровий двійник при цьому здатен оновлюватися у режимі реального часу, відображаючи зміни фізіологічних характеристик та надаючи можливість досліджувати, прогнозувати та моделювати розвиток патологічних процесів.

Незважаючи на значний прогрес у цій сфері, існують проблеми уніфікації форматів даних, відсутність узагальнених підходів до представлення мультимодальних темпоральних сигналів та складнощі з інтеграцією технологій цифрових двійників у медичні інформаційні системи. Також потребують уваги питання анонімізації та безпеки медичних даних, стандартизації програмної архітектури, а також адаптації базових 3D-моделей під індивідуальні особливості конкретного пацієнта.

Метою дисертаційної роботи є підвищення ефективності процесів проєктування медичних інформаційних систем на основі технології цифрових двійників за рахунок розроблення алгоритмічного та програмного

забезпечення для оброблення мультимодальних темпоральних даних про медико-біологічний об'єкт. Для досягнення цієї мети було розроблено метод синхронізації та семантичного аналізу темпоральних мультимодальних медико-біологічних даних, метод адаптації базової 3D-моделі медико-біологічного об'єкта, розроблено узагальнену архітектуру програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів а також архітектурні шаблони проєктування для розроблення програмних систем на основі цифрових двійників медико-біологічних об'єктів.

У першому розділі дисертації проаналізовано відомі методи обробки даних про медико-біологічні об'єкти, підходи до створення цифрових двійників та вимоги до програмного забезпечення. Виявлено проблеми у представленні мультимодальних даних і визначено ключові критерії для забезпечення високої точності та функціональності цифрових двійників.

У другому розділі розроблено методи синхронізації темпоральних мультимодальних даних, а також семантичного аналізу та інтеграції інформації. Запропоновано використання семантичних графів та онтологій, а також графових нейронних мереж для ефективного виявлення закономірностей та прогнозування поведінки. Обґрунтовано застосування нормалізації, синхронізації сигналів, крос-кореляційного аналізу й інтеграції даних у графовій моделі.

У третьому розділі розроблено метод адаптації базової 3D-моделі медико-біологічного об'єкта. Застосовано 3D-згорткові нейронні мережі для аналізу просторово-часових залежностей у відеопотоці та потоці аудіоданих. Інтегровано мультимодальний підхід, що дає змогу поєднувати інформацію з різних джерел для персоналізованого моделювання стану медико-біологічного об'єкта та його візуалізації.

У четвертому розділі розроблено архітектуру медичної програмної системи на основі технології цифрових двійників. Запропоновано

мікросервісний підхід, використання контейнеризації, засобів безпеки, анонімізації та автоматизації тестування (CI/CD) для забезпечення масштабованості, надійності та високої якості програмного забезпечення. Описано можливості інтеграції з існуючими медичними системами, принципи проєктування інтерфейсів візуалізації та ролей користувачів у системі.

У дисертаційній роботі отримано низку **нових наукових результатів**, зокрема, **уперше** розроблено узагальнену архітектуру програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів, характерними рисами якої є поєднання мультимодальних темпоральних даних у форматі, який підтримує двосторонню інтеграцію з фізичним об'єктом через давачі, актуатори та інші програмно-апаратні засоби, що надає можливість перейти від фрагментарного оброблення окремих типів даних до цілісної моделі, яка може оновлюватися в реальному часі.

Уперше розроблено метод синхронізації темпоральних мультимодальних даних, характерною рисою якого є поєднання відео та аудіо у єдиний потік даних, що забезпечує узгодження даних різної модальності та, у такий спосіб, спрощує процес створення цифрового двійника медико-біологічного об'єкта на основі даних, які надходять з давачів різних типів.

Уперше розроблено метод семантичного аналізу для виявлення кореляцій між наборами даних та прогнозування поведінки програмно-апаратних компонентів цифрового двійника, характерними рисами якого є застосування графових баз даних та алгоритмів машинного навчання, що дає змогу об'єднувати дані з різних джерел (пацієнти, пристрої, записи) в єдину онтологічну модель, що забезпечує автоматизоване виявлення залежностей у даних про медико-біологічний об'єкт, а також надає

інструменти для прогнозування стану програмно-апаратного забезпечення цифрового двійника.

Удосконалено теоретичні засади оброблення просторово-часових параметрів медико-біологічного об'єкта для побудови його цифрового двійника, що полягає у застосуванні тривимірних згорткових нейронних мереж (3D-CNN) та рекурентних архітектур для оброблення відео- та аудіоданих та, на відміну від відомих підходів, дає змогу забезпечити комплексний аналіз динамічних змін структури та функціонування медико-біологічного об'єкта з врахуванням його індивідуальних анатомічних особливостей та динаміки.

Уперше розроблено архітектурні шаблони проєктування для розроблення програмних систем на основі цифрових двійників медико-біологічних об'єктів, які, на відміну від відомих, орієнтовані на оперування складними наборами мультимодальних темпоральних даних, інтегрованих у єдину семантичну модель, що дає змогу спростити процес розроблення, обслуговування та масштабування медичних програмних систем.

Основні наукові результати дисертаційної роботи **опубліковано** у наукових працях, зокрема у 4 наукових статтях, опублікованих у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б», та у 2 матеріалах наукової конференції.

Ключові слова: інженерія програмного забезпечення, програмне забезпечення технології цифрових двійників, програмування, алгоритмічне забезпечення, мультимодальні дані, медико-біологічний об'єкт, нейронні мережі, семантичні графи, онтологія, моделювання, архітектура програмної системи, хмарні обчислення, мікросервіси, анонімізація, CI/CD, давачі.

Peschanskyi V.Y. Algorithmic and Software Support for the Digital Twin Technology of Medical and Biological Objects. – A Qualifying Scientific Work in the form of a Manuscript.

technologies in speciality 121 Software Engineering. – National Technical

Digital twins are becoming increasingly widespread in the field of medicine as they make it possible to create a dynamic digital model of a real medical-biological object, reflecting both the anatomical and functional aspects of its operation. This technology involves integrating data from various sources (medical images, video endoscopy, audio data, sensor readings) into a single information structure describing the state and dynamics of the object. The digital twin can be updated in real time, reflecting changes in physiological characteristics and enabling the study, prediction, and modeling of the development of pathological processes.

Despite significant progress in this area, there are issues related to the unification of data formats, the lack of generalized approaches to representing multimodal temporal signals, and the complexities of integrating digital twin technologies into medical information systems. Questions of anonymization and security of medical data, standardization of software architecture, and adaptation of basic 3D models to individual patient characteristics also require attention.

The purpose of this dissertation is to improve efficiency of the processes of medical information systems design based on digital twin technology by developing algorithmic and software tools for processing multimodal temporal data about a medical-biological object. To achieve this goal, a method for synchronizing and performing semantic analysis of temporal multimodal medical-biological data was developed, along with a method for adapting a basic 3D model of a medical-biological object, a generalized architecture of a software

system for creating and using digital twins of medical-biological objects, and architectural design patterns for developing software systems based on digital twins of medical-biological objects.

In the first chapter of the dissertation, known methods of data processing for medical-biological objects, approaches to creating digital twins, and software requirements are analyzed. Problems in representing multimodal data are identified, and key criteria for ensuring high accuracy and functionality of digital twins are determined.

In the second chapter, methods for synchronizing temporal multimodal data, as well as for performing semantic analysis and information integration, are developed. The use of semantic graphs and ontologies, as well as graph neural networks, is proposed for the effective detection of patterns and prediction of behavior. The application of normalization, signal synchronization, cross-correlation analysis, and data integration in a graph-based model is substantiated.

In the third chapter, a method for adapting the basic 3D model of a medical-biological object is developed. Three-dimensional convolutional neural networks (3D CNNs) are used to analyze spatio-temporal dependencies in the video stream and audio data stream. A multimodal approach is integrated, enabling the combination of information from various sources for personalized modeling of the state of the medical-biological object and its visualization.

In the fourth chapter, the architecture of a medical software system based on digital twin technology is developed. A microservices approach is proposed, utilizing containerization, security tools, anonymization, and automated testing (CI/CD) to ensure scalability, reliability, and high-quality software. Possibilities for integration with existing medical systems are described, along with principles for designing visualization interfaces and user roles within the system.

A number of **new scientific results** have been obtained in this dissertation, specifically, **for the first time**, a generalized architecture of a software system for creating and using digital twins of medical-biological objects has been

developed. Its distinctive features include the integration of multimodal temporal data in a format that supports bidirectional interaction with a physical object via sensors, actuators, and other hardware-software means, enabling a transition from fragmented processing of individual data types to a holistic model capable of real-time updates and providing a simplified process for designing medical information systems based on digital twin technology.

For the first time, a method for synchronizing temporal multimodal data has been developed. Its distinctive feature is the combination of video and audio into a single data stream, ensuring alignment of different data modalities. This approach simplifies the creation of a digital twin of a medical-biological object based on data from sensors of various types.

For the first time, a method for semantic analysis has been developed for detecting correlations between datasets and predicting the behavior of digital twin hardware-software components. Its distinctive features include the use of graph databases and machine learning algorithms, allowing data from various sources (patients, devices, records) to be combined into a single ontological model. This provides automated detection of dependencies in medical-biological object data and offers tools for predicting the state of the digital twin's hardware and software components.

The theoretical foundations of processing spatio-temporal features for constructing a digital twin **have been improved** by applying three-dimensional convolutional neural networks (3D CNNs) and recurrent architectures for video and audio data processing. Unlike known approaches, this makes it possible to provide a comprehensive analysis of the dynamic changes in the structure and functioning of a medical-biological object, taking into account its individual anatomical features and functional dynamics.

For the first time, architectural design patterns for developing software systems based on digital twins of medical-biological objects have been created. Unlike known patterns, they are oriented towards managing complex sets of

multimodal temporal data integrated into a single semantic model, thereby simplifying the development, maintenance, and scaling of medical software systems.

The main scientific results of the dissertation are published in 6 scientific works, including 4 articles in specialized journals recognized by the list of scientific specialized editions of Ukraine (category “B”) and 2 conference proceedings.

Keywords: software engineering, digital twin technology software, programming, algorithms, multimodal data, medical-biological object, neural networks, semantic graphs, ontology, modelling, software system architecture, cloud computing, microservices, anonymization, CI/CD, sensors.

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. Песчанський В.Ю., Сулема Є.С. Проектування архітектури програмної системи для створення цифрових двійників медико-біологічних об'єктів. «Системні технології». 2023. № 5 (148). С. 62-70.
D
2. Песчанський В.Ю., Сулема Є.С. Пошук ключових точок на зображеннях для створення цифрових двійників медико-біологічних об'єктів.
«Системні технології». 2023. № 6 (149). С. 3-10.
D
3. Песчанський В.Ю., Сулема Є.С. Метод синхронізації темпоральних мультимодальних даних для створення цифрового двійника гортані.
«Системні технології». 2024. № 5 (154). С. 137-145.
D
4. Песчанський В.Ю., Сулема Є.С. Архітектурні принципи забезпечення верифікації та якості системи створення цифрових двійників медико-біологічних об'єктів. «Вісник Хмельницького національного університету. Технічні науки». 2024. Том 345 № 6(2). С. 158-164.
D
O
I

Публікації у матеріалах наукових конференцій / the publications in

1. Песчанський В.Ю., Сулема Є.С. Відтворення тривимірної моделі об'єкту на основі набору зображень. Тринадцята наукова конференція магістрантів та аспірантів «Прикладна математика та комп'ютинг (ПМК'2020)». Київ. 18 - 20 листопада 2020 р. Збірник тез доповідей Нац. техн. ун-т України «Київ. політехн. ін-т ім. Ігоря Сікорського». Київ: Просвіта. – 2020. – С. 233-236.
2. Песчанський В.Ю., Сулема Є.С. Алгоритм зчитування та аналізу даних медико-біологічних об'єктів у форматі PDF на основі оптичного розпізнавання символів. П'ятнадцята наукова конференція магістрантів та аспірантів «Прикладна математика та комп'ютинг (ПМК'2022)», Київ, 16 - 18 листопада 2022 р. Збірник тез доповідей Нац. техн. ун-т України «Київ. політехн. ін-т ім. Ігоря Сікорського», Київ: Просвіта. – 2022. – С. 416-420.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	14
ВСТУП	16
РОЗДІЛ 1. АНАЛІЗ МАТЕМАТИЧНОГО, АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ОБРОБЛЕННЯ ДАНИХ ПРО МЕДИКО-БІОЛОГІЧНІ ОБ'ЄКТИ	22
1.1. Аналіз математичного та алгоритмічного забезпечення оброблення даних	22
1.2. Аналіз концепції цифрового двійника медико-біологічного об'єкта	26
1.3. Аналіз досліджень та відомих рішень	32
1.4. Вимоги до системи створення цифрових двійників медико- біологічних об'єктів	39
1.5. Висновки до розділу 1	50
РОЗДІЛ 2. РОЗРОБЛЕННЯ МЕТОДІВ СИНХРОНІЗАЦІЇ ТА АНАЛІЗУ ТЕМПОРАЛЬНИХ МУЛЬТИМОДАЛЬНИХ ДАНИХ	52
2.1. Організація темпоральних даних медико-біологічних об'єктів	55
2.2. Використання метаданих та часових міток у темпоральних даних ..	61
2.3. Основні принципи семантичного аналізу та синхронізації темпоральних мультимодальних даних	63
2.4. Семантичні графи	65
2.5. Метод синхронізації темпоральних мультимодальних даних	69
2.6. Метод семантичного аналізу для виявлення кореляцій між наборами даних та прогнозування поведінки програмно-апаратних компонентів цифрового двійника	78
2.7. Висновки до розділу 2	91
РОЗДІЛ 3. РОЗРОБЛЕННЯ МЕТОДУ АДАПТАЦІЇ БАЗОВОЇ МОДЕЛІ МЕДИКО-БІОЛОГІЧНОГО ОБ'ЄКТА	93
3.1. Теоретичні засади оброблення даних для створення цифрових двійників	95

3.2. Попереднє оброблення даних	97
3.3. Побудова цифрового двійника	105
3.4. Висновки до розділу 3	119
РОЗДІЛ 4. РОЗРОБЛЕННЯ АРХІТЕКТУРИ МЕДИЧНОЇ ПРОГРАМНОЇ СИСТЕМИ НА ОСНОВІ ТЕХНОЛОГІЇ ЦИФРОВИХ ДВІЙНИКІВ	120
4.1. Архітектурні шаблони проєктування для розроблення мультимедійного програмного забезпечення	121
4.2. Узагальнена архітектура програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів	139
4.3. Інтерфейси та взаємодія з користувачами	148
4.4. Особливості впровадження.....	150
4.5. Висновки до розділу 4	152
ВИСНОВКИ	154
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	156
ДОДАТКИ	159

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

3D-CNN – Three-Dimensional Convolutional Neural Network (тривимірна згорткова нейронна мережа)

AR – Augmented Reality (доповнена реальність)

AWS – Amazon Web Services (веб-сервіси Амазон)

CI/CD – Continuous Integration / Continuous Deployment (безперервна інтеграція / безперервне розгортання)

CNN – Convolutional Neural Network (згорткова нейронна мережа)

CSV – Comma Separated Values (значення, розділені комами)

DICOM – Digital Imaging and Communications in Medicine (цифрова система зберігання та обміну медичними зображеннями)

DWT – Discrete Wavelet Transform (дискретне вейвлет-перетворення)

DTDL – Digital Twins Definition Language (мова опису цифрових двійників)

FHIR – Fast Healthcare Interoperability Resources (швидкі ресурси взаємодії у системах охорони здоров'я)

GAT – Graph Attention Network (графова мережа з механізмом уваги)

GCN – Graph Convolutional Network (графова згорткова нейронна мережа)

GNN – Graph Neural Network (графова нейронна мережа)

GraphSAGE – Graph Sample and Aggregate (метод для генерації представлень вузлів графа через вибірку та агрегацію)

HDF5 – Hierarchical Data Format version 5 (ієрархічний формат даних версії

HL7 – Health Level Seven International (стандарт обміну електронними медичними даними)

IoT – Internet of Things (Інтернет речей)

JSON – JavaScript Object Notation (формат обміну даними в текстовому вигляді)

JSON-LD – JSON for Linked Data (JSON для зв'язаних даних)

LIS – Laboratory Information System (лабораторна інформаційна система)

LSTM – Long Short-Term Memory (архітектура рекурентної нейронної мережі для довгострокових залежностей)

MRI – Magnetic Resonance Imaging (магнітно-резонансна томографія)

NASA – National Aeronautics and Space Administration (Національне управління з авіації і космічного простору, США)

PACS – Picture Archiving and Communication System (система архівування та передачі медичних зображень)

RIS – Radiology Information System (радіологічна інформаційна система)

UTC – Coordinated Universal Time (універсальний скоординований час)

VR – Virtual Reality (віртуальна реальність)

ДІ – дискретний інтервал

МБО – медико-біологічний об'єкт

MRT – магнітно-резонансна томографія (MRI)

ОС – операційна система

ВСТУП

Актуальність теми. Стрімкий розвиток інформаційних технологій, штучного інтелекту та інструментів глибокого навчання створює сприятливі умови для вдосконалення технології цифрових двійників та впровадження її у різні сфери діяльності, зокрема медицину та освіту. Цифрові двійники являють собою динамічні комплексні цифрові моделі реальних фізичних об'єктів, процесів або систем, які можуть оновлюватися в реальному часі на основі фактичних даних. Такий підхід дозволяє здійснювати глибокий аналіз, симуляції, прогнозування та оптимізацію, що сприяє підвищенню ефективності, якості та точності прийняття рішень.

Технологія цифрових двійників уже знайшла широке застосування у промисловості, виробництві, логістиці та енергетиці. Зокрема, у виробничій сфері цифрові двійники використовуються для моделювання та оптимізації технологічних процесів, підвищення надійності устаткування та прогнозування можливих збоїв. У логістиці створення цифрових двійників допомагає вдосконалити системи управління ланцюгами постачання, планувати маршрути, зменшувати витрати та покращувати точність прогнозування запасів. Крім того, технологія цифрових двійників сприяє сталому розвитку шляхом оцінювання впливу різних сценаріїв експлуатації технічних об'єктів на довкілля.

Водночас існує значний потенціал для застосування технології цифрових двійників у таких людиноцентричних галузях, як медицина та освіта. У медицині застосування цифрових двійників може сприяти дистанційній діагностиці, плануванню складних хірургічних втручань, точному аналізу результатів досліджень та прогнозуванню стану пацієнта. В освітній сфері цифрові двійники дозволяють створювати імерсивні навчальні середовища, що дають змогу студентам набувати практичних навичок у безпечних та контрольованих умовах.

Однак аналіз наявних програмних систем для створення цифрових двійників виявляє їх переважну вузькофункціональну спрямованість та недостатню універсальність. Більшість рішень орієнтовані на конкретну галузь або тип об'єкта та не надають узагальнені підходи та інструменти для створення цифрових двійників довільних фізичних об'єктів. Це ускладнює розвиток та впровадження універсальних систем, здатних адаптуватися під специфічні вимоги різних предметних галузей.

Актуальність проблеми посилюється появою та розвитком концепції мультимедіа (multimedia) – інтеграції мультимодальних даних (відео, аудіо, тактильних, смакових, нюхових сигналів) у єдину інформаційну структуру. Використання мультимедійних даних у поєднанні з технологіями віртуальної (VR) та доповненої (AR) реальності розширює можливості цифрових двійників, дозволяючи створювати більш реалістичні, інтерактивні та адаптивні моделі фізичних об'єктів – від технічних пристроїв до медико-біологічних систем і навчальних симуляторів.

Таким чином, актуальною науково-технічною задачею є вдосконалення теоретичних засад розроблення алгоритмічного та програмного забезпечення технології цифрових двійників для оброблення темпоральних мультимодальних даних про об'єкт дослідження. Розв'язання цієї задачі відкриває можливості для створення уніфікованих, гнучких і стандартизованих програмних систем, придатних до використання у різних галузях, включаючи медицину та освіту.

Зв'язок роботи з науковими програмами, планами, темами. Дослідження виконані у рамках держбюджетної НДР «Математичні та програмні методи оброблення мультимодальних даних моніторингу медико-біологічних об'єктів для діагностики стану здоров'я пацієнтів» (№ держреєстрації 0120U102134), яка здійснювалася у Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського». Запропоновані у дисертації методи та підходи до

створення цифрових двійників мультимедійних об'єктів узгоджуються з програмами розвитку інформаційних технологій у сфері охорони здоров'я та освіти.

Мета і задачі дослідження. Метою дисертаційної роботи є підвищення ефективності процесів проєктування медичних інформаційних систем на основі технології цифрових двійників за рахунок розроблення алгоритмічного та програмного забезпечення для оброблення мультимодальних темпоральних даних про медико-біологічний об'єкт.

Для досягнення поставленої мети необхідно вирішити такі задачі:

- провести аналіз програмного забезпечення на основі технології цифрових двійників та сформулювати вимоги до програмних систем, призначених для створення цифрових двійників медико-біологічних об'єктів;
- розробити алгоритмічне та програмне забезпечення процесів консолідації даних про медико-біологічний об'єкт з урахуванням їхніх темпоральних та мультимодальних властивостей;
- розробити узагальнену архітектуру програмної системи для оброблення даних цифрових двійників медико-біологічних об'єктів;
- розробити архітектурні шаблони проєктування, орієнтовані на оперування даними медико-біологічних об'єктів.

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

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

Методи дослідження: теорія програмування, теорія програмних систем, теорія інформаційних систем, теорія алгоритмів, теорія машинного навчання, теорія паралельних і розподілених обчислень, методи проєктування архітектур програмних систем.

Наукова новизна одержаних результатів полягає у наступному.

Уперше розроблено узагальнену архітектуру програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів, характерними рисами якої є поєднання мультимодальних темпоральних даних у форматі, який підтримує двосторонню інтеграцію з фізичним об'єктом через давачі, актуатори та інші програмно-апаратні засоби, що надає можливість перейти від фрагментарного оброблення окремих типів даних до цілісної моделі, яка може оновлюватися в реальному часі.

Уперше розроблено метод синхронізації темпоральних мультимодальних даних, характерною рисою якого є поєднання відео та аудіо у єдиний потік даних, що забезпечує узгодження даних різної модальності та, у такий спосіб, спрощує процес створення цифрового двійника медико-біологічного об'єкта на основі даних, які надходять з давачів різних типів.

Уперше розроблено метод семантичного аналізу для виявлення кореляцій між наборами даних та прогнозування поведінки програмно-апаратних компонентів цифрового двійника, характерними рисами якого є застосування графових баз даних та алгоритмів машинного навчання, що дає змогу об'єднувати дані з різних джерел (пацієнти, пристрої, записи) в єдину онтологічну модель, що забезпечує автоматизоване виявлення залежностей у даних про медико-біологічний об'єкт, а також надає інструменти для прогнозування стану програмно-апаратного забезпечення цифрового двійника.

Удосконалено теоретичні засади оброблення просторово-часових параметрів медико-біологічного об'єкта для побудови його цифрового двійника, що полягає у застосуванні тривимірних згорткових нейронних мереж (3D-CNN) та рекурентних архітектур для оброблення відео- та аудіоданих та, на відміну від відомих підходів, дає змогу забезпечити

комплексний аналіз динамічних змін структури та функціонування медико-біологічного об'єкта з врахуванням його індивідуальних анатомічних особливостей та динаміки.

Уперше розроблено архітектурні шаблони проєктування для розроблення програмних систем на основі цифрових двійників медико-біологічних об'єктів, які, на відміну від відомих, орієнтовані на оперування складними наборами мультимодальних темпоральних даних, інтегрованих у єдину семантичну модель, що дає змогу спростити процес розроблення, обслуговування та масштабування медичних програмних систем.

Практичне значення одержаних результатів полягає у створенні алгоритмічно-програмного забезпечення та архітектурних рішень, які спрощують процеси розроблення медичних програмних систем на основі цифрових двійників медико-біологічних об'єктів. Зокрема, результатами дослідження можуть користуватися розробники медичних інформаційних систем, освітніх симуляторів, систем телемедицини, а також підприємства та організації, зацікавлені у впровадженні технології цифрових двійників.

Особистий внесок здобувача. Усі основні результати дисертаційного дослідження, які представлені до захисту, одержані автором особисто. У публікаціях, написаних у співавторстві, здобувач є основним автором результатів, пов'язаних із методами.

У роботі [1] здобувачем запропоновано методи проєктування архітектури системи для створення цифрових двійників медико-біологічних об'єктів. У роботі [2] здобувачем запропоновано метод створення цифрових двійників медико-біологічних об'єктів на прикладі отоларингології. У роботі [3] здобувачем запропоновано метод синхронізації темпоральних мультимодальних даних для створення цифрового двійника гортані. У роботі [4] архітектурні принципи забезпечення верифікації та якості системи створення цифрових двійників медико-біологічних об'єктів.

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

1. Песчанський В.Ю., Сулема Є.С. Відтворення тривимірної моделі об'єкту на основі набору зображень. Тринадцята наукова конференція магістрантів та аспірантів «Прикладна математика та комп'ютинг (ПМК'2020)». Київ. 18 - 20 листопада 2020 р. Збірник тез доповідей Нац. техн. ун-т України «Київ. політехн. ін-т ім. Ігоря Сікорського». Київ: Просвіта. – 2020.
2. Песчанський В.Ю., Сулема Є.С. Алгоритм зчитування та аналізу даних медико-біологічних об'єктів у форматі pdf на основі оптичного розпізнавання символів. П'ятнадцята наукова конференція магістрантів та аспірантів «Прикладна математика та комп'ютинг (ПМК'2022)», Київ, 16 - 18 листопада 2022 р. Збірник тез доповідей Нац. техн. ун-т України «Київ. політехн. ін-т ім. Ігоря Сікорського», Київ: Просвіта. – 2022.

Публікації. Основні наукові результати дисертаційної роботи опубліковано у 6 наукових працях, зокрема у 4 наукових статтях, опублікованих у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б», та у 2 матеріалах наукової конференції.

Структура роботи. Дисертація складається із вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг становить 172 сторінок, включаючи 140 сторінок основного тексту, 16 рисунків та 2 таблиці.

РОЗДІЛ 1. АНАЛІЗ МАТЕМАТИЧНОГО, АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ОБРОБЛЕННЯ ДАНИХ ПРО МЕДИКО-БІОЛОГІЧНІ ОБ'ЄКТИ

Дослідження медико-біологічних об'єктів передбачає виконання кількох важливих завдань: отримання даних про конкретний об'єкт, їхню попередню обробку, структурування та подальший аналіз, що залежить від мети дослідження. До споріднених задач належать моделювання стану об'єкта, аналіз його характеристик і поведінки, а також вивчення зовнішніх аспектів та складових частин об'єкта. Для ефективного вирішення цих задач доцільно застосовувати концепцію цифрового двійника — комплексної моделі, яка максимально точно відтворює реальний об'єкт дослідження.

Для визначення основних проблем, пов'язаних з обробкою та представленням даних про медико-біологічні об'єкти, розглянемо технологію створення цифрових двійників. Також проаналізуємо наявне математичне, алгоритмічне та програмне забезпечення, яке може бути використане для вирішення поставленої науково-технічної задачі, пов'язаної зі створенням цифрового двійника медико-біологічного об'єкта.

1.1. Аналіз математичного та алгоритмічного забезпечення оброблення даних

Математичне моделювання внутрішніх органів відіграє критичну роль у розумінні їхньої функції, взаємодії з іншими системами організму та відповіді на різноманітні медичні втручання. Проаналізуємо основні математичні методи, які застосовуються для моделювання внутрішніх органів людини, дозволяючи детально аналізувати їхню структуру.

Для проєктування програмного забезпечення медичних інформаційних систем можна виділити такі основні види моделювання [1].

Анатомічне моделювання призначене для створення точних тривимірних репрезентацій структури внутрішніх органів. У цьому типі моделювання дані, отримані з МРТ або КТ, використовуються для детальної візуалізації анатомічних особливостей органів з урахуванням їхніх розмірів, форми та будови. Анатомічне моделювання може використовуватися для планування хірургічних втручань, медичної освіти та досліджень.

Фізіологічне моделювання спрямоване на імітацію функцій органів та систем організму шляхом використання математичних та комп'ютерних моделей для відтворення біологічних процесів. Це може включати моделювання кровообігу, дихання, метаболічних процесів, а також взаємодії між різними органами. Фізіологічне моделювання дозволяє дослідникам розуміти, як зміни в одному органі можуть вплинути на функціонування інших частин організму, а також прогнозувати відповідь організму на лікувальні втручання.

Тепер розглянемо три основні підходи, які можуть бути застосовані для моделювання медико-біологічних систем, а саме: детерміноване моделювання, стохастичне моделювання та мульти-масштабне моделювання.

Детерміноване моделювання [2-3] ґрунтується на застосуванні математично визначеного опису процесів, де результат однозначно визначається початковими умовами. Прикладом такого моделювання може слугувати модель глюкозно-інсулінової взаємодії в організмі людини, також відома як модель Бергмана [4], яка застосовується для аналізу динаміки рівнів глюкози та інсуліну в крові та може бути використана для оптимізації стратегій лікування діабету.

Стохастичне моделювання [5] ґрунтується на припущенні наявності випадковості у процесах, що моделюються, і використовує вірогіднісні розподіли для опису цих процесів. Цей підхід дозволяє враховувати невизначеність і варіабельність у процесах, що досліджуються. Прикладом

може слугувати моделювання розвитку раку шкіри, при якому необхідно врахувати випадкові мутації та екологічні фактори, які впливають на ризик розвитку захворювання. Така модель дозволяє оцінити ризик розвитку раку шкіри в залежності від рівня УФ-випромінювання, захисних заходів (наприклад, використання сонцезахисного крему), генетичної схильності та інших факторів. Цей підхід може бути використаний для подальшого розроблення персоналізованих рекомендацій та програм для профілактики раку шкіри та його раннього виявлення.

Мульти-масштабне моделювання [6] ґрунтується на комбінуванні різних рівнів деталізації, від молекулярного до органного рівня, для створення комплексного уявлення про біологічну систему. Прикладом може слугувати моделювання впливу лікарського препарату, де на молекулярному рівні моделюється взаємодія препарату з молекулами-мішенями, а на органному рівні – моделюються процеси зменшення концентрації препарату в організмі з плином часу. Ці методи дозволяють здійснювати комплексний аналіз властивостей та поведінки медико-біологічних систем, враховуючи їх складність та ієрархічність.

Методи оптимізації [7] дають змогу визначити оптимальні параметри моделей для точного відображення реальних біологічних процесів та підтримки ефективного лікування. Ці методи використовуються для вирішення різноманітних задач, включаючи калібрування моделей, проєктування експериментів, а також для розроблення та оптимізації стратегій лікування. Одним з основних застосувань оптимізації в медико-біологічному моделюванні є визначення оптимальних значень параметрів моделі, які найкраще відповідають доступним експериментальним даним. Для цього часто використовують наступні методи:

- **метод найменших квадратів** [8], який призначений для мінімізації значення суми квадратів відхилень прогнозованих значень від спостережуваних даних;

- **метод максимальної правдоподібності** [9], який є статистичним підходом для оцінки параметрів моделі для вибору значень параметрів, що максимізують функцію правдоподібності, тобто, ймовірність отримати спостережувані дані з даної моделі.

Для поставленої задачі доцільно буде використовувати **метод максимальної правдоподібності**, так як однією з цілей дослідження є створення такої моделі медико-біологічного об'єкту, що дозволить працівникам медичної галузі моделювати вплив ліків або інших видів терапії на медико-біологічний об'єкт без взаємодії з реальним екземпляром.

Також одним з перспективних напрямів дослідження є алгоритми машинного навчання. Наприклад, алгоритми глибокого навчання застосовуються для аналізу медичних зображень, таких як рентгенівські знімки, МРТ та КТ [10], дозволяючи ідентифікувати патології, що раніше могли залишитися непоміченими людським оком. Це значно підвищує точність діагностики захворювань, таких як рак, дозволяючи розпочати лікування на ранніх стадіях. Крім того, алгоритми машинного навчання знаходять застосування у прогнозуванні перебігу захворювань, дозволяючи медичним працівникам визначати найбільш ймовірні сценарії розвитку стану пацієнта на основі історичних даних про подібні випадки. Це допомагає приймати обґрунтовані клінічні рішення та планувати лікування, зосереджуючись на найкращих можливих результатах для пацієнта.

Гіпотеза про доцільність застосування алгоритмів машинного навчання для оброблення медичних графічних зображень перевірена на прикладі оброблення анонімних мультимодальних даних, наданих фахівцями Інституту отоларингології ім. проф. О.С. Коломійченка Національної академії медичних наук України. На рис. 1.1 зображено знаходження ключових точок гортані в рамках вищезазначеного дослідження. Отримані результати [11] підтвердили доцільність використання машинного навчання при обробці графічних зображень.

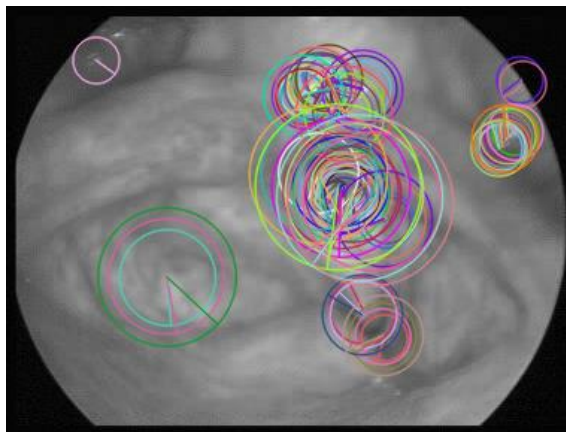


Рис. 1.1. Знайдені ключові точки на зображенні гортані

Таким чином, використання алгоритмів машинного навчання у моделюванні медико-біологічних об'єктів стає ключовим елементом сучасної медицини, відкриваючи нові можливості для підвищення якості та ефективності медичного догляду [11].

1.2. Аналіз концепції цифрового двійника медико-біологічного об'єкта

Технологія цифрового двійника представляє собою відносно новий метод представлення фізичних об'єктів або процесів у цифровому форматі, включаючи їх минулий, поточний та майбутній стани, а також поведінку, атрибути та зовнішній вигляд. Згідно з визначенням, наданим розробниками концепції цифрового двійника [12], цифровий двійник становить сукупність віртуальних інформаційних структур, що детально описують фізичний об'єкт на всіх рівнях, від мікроскопічного до макроскопічного, включаючи загальний вигляд та основні властивості.

Цифровий двійник слугує динамічною моделлю фізичного об'єкта або процесу, точно відтворюючи його характеристики протягом часу. Ключовою особливістю цифрового двійника є здатність представляти, обробляти та маніпулювати всіма даними, які характеризують фізичний

об'єкт, та ефективно використовувати ці дані для аналізу, прогнозування та оптимізації управління.

Поняття «цифровий двійник» було вперше введено у звіті Національного управління з аеронавтики і дослідження космічного простору США [13], де було визначено, що цифровий двійник є інтегрованою мультифізичною, багаторівневою, ймовірнісною симуляцією об'єкта дослідження, такого як літальний апарат, яка використовує передові моделі та безперервно оновлюється за даними з сенсорів реального об'єкта, відображаючи весь життєвий цикл фізичного об'єкта.

Цифровий двійник забезпечує детальне і реалістичне відображення фізичного об'єкта, а аналіз даних цифрового двійника допомагає ідентифікувати потенційні проблеми до виникнення критичних ситуацій, тим самим дозволяючи попередити небезпечні ситуації. Наприклад, застосування цифрового двійника до медичної галузі інтегрує дані з медичних приладів, історію лікування та іншу релевантну інформацію, створюючи індивідуалізовану та реалістичну віртуальну модель об'єкта дослідження.

Цифровий двійник являє собою комплексну цифрову репрезентацію фізичного об'єкта або процесу, включаючи його візуальні атрибути та поведінкові характеристики, які засновані на відповідних математичних моделях та методах подання даних. Це дозволяє досягти безперервної синхронізації між віртуальною моделлю та реальним об'єктом завдяки даним, отриманим від сенсорів, які здійснюють постійний моніторинг стану об'єкта.

З технічної точки зору, цифровий двійник являє собою складну програмну систему, яка інтегрує бази даних і спеціалізовані програмні модулі для збору, обробки, аналізу та візуалізації інформації про реальний об'єкт.

Цифрові двійники поділяються на два основні типи: *прототипи* та *екземпляри* [14].

Цифровий двійник-прототип розробляється для об'єктів, які ще не існують у фізичному світі, і слугує основою для їх майбутнього створення. Він містить комплексні набори даних, необхідні для опису всіх аспектів фізичного об'єкта, включаючи його вимоги, тривимірні моделі, специфікації матеріалів та інструкції з виробництва.

Цифровий двійник-екземпляр створюється для вже існуючого фізичного об'єкта і супроводжує його на всіх етапах життєвого циклу, від виробництва до експлуатації та утилізації, забезпечуючи детальний моніторинг його стану та ефективності.

Крім вищенаведених, існує також концепція *цифрового двійника-агрегата* [15], який утворюється шляхом об'єднання даних з множини цифрових двійників-екземплярів. Цей підхід використовується для аналізу цілої групи об'єктів або системи об'єктів, дозволяючи отримати узагальнену інформацію про їх поведінку, характеристики та можливі проблеми. Цифровий двійник-агрегат дозволяє проводити більш широкомасштабні дослідження та аналізи, що є особливо корисним у моделюванні медико-біологічних об'єктів, де в одній з задач необхідно враховувати взаємодію між різними компонентами надзвичайно складної системи.

Середовище цифрового двійника визначається як інтегрована система, що включає як програмне, так і апаратне забезпечення, призначене для детального дослідження одного або декількох фізичних об'єктів. Дана система виконує дві наступні ключові ролі.

Прогнозування, тобто цифровий двійник застосовується для аналізу майбутньої поведінки фізичного об'єкта, дозволяючи оцінити можливі сценарії його розвитку ще до реалізації в реальному світі. У контексті прототипу, це дає змогу розглянути різні варіанти проектування та впливу на об'єкт перед його виробництвом. Для уже існуючих екземплярів,

прогнозування спрямоване на ідентифікацію потенційних ризиків та моделювання реакції об'єкта на різноманітні впливи.

Інформаційна функція, тобто використання цифрового двійника надає важливу інформацію про стан та характеристики досліджуваного об'єкта, що є критичним для проведення глибоких дослідницьких аналізів.

На основі ступеня інтеграції між цифровими та фізичними компонентами, розрізняють три наступні категорії [17], наочна ілюстрація відмінностей категорій наведена на рис 1.2.

Цифрова модель, тобто цифрове представлення, яке не передбачає автоматичного обміну даними між цифровою та фізичною версіями об'єкта. Цифрові моделі використовують для імітації та аналізу об'єкта, коли зворотний зв'язок між реальним станом об'єкта та моделлю відсутній.

Цифрова тінь, тобто цифрове відображення, що забезпечує односторонній потік даних від фізичного об'єкта до цифрового. Це дозволяє цифровій версії автоматично оновлюватися відповідно до змін у фізичному об'єкті, але без можливості впливати на нього.

Цифровий двійник, тобто цифрове подання, яке підтримує двосторонній обмін даними, дозволяючи не лише відстежувати зміни в фізичному об'єкті, але й впливати на нього через цифрові втручання. Цифровий двійник діє як повністю інтегрована система для управління та оптимізації стану фізичного об'єкта.

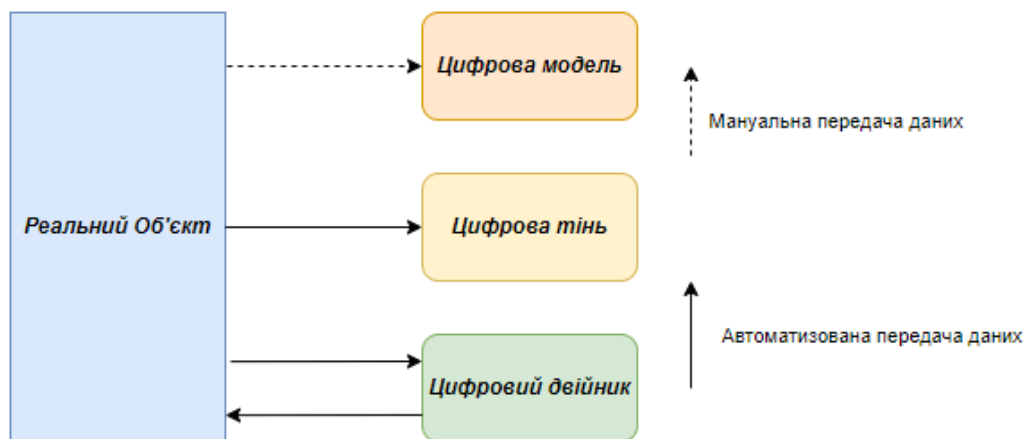


Рис. 1.2. Категорії цифрових двійників за ступенем інтеграції

Цифрова тінь є найбільш доцільною формою цифрового двійника для дослідження та моделювання медико-біологічних об'єктів, оскільки вона дозволяє точно відображати їх поточний стан і динаміку змін, надаючи цінні дані для аналізу, прогнозування та ухвалення рішень.

Відповідно до складності моделі, цифрові двійники класифікують відповідно до наступних чотирьох основних рівнів [18].

Рівень «*Pre-digital twin*» — первинний віртуальний прототип, створений на ранній стадії інженерних розробок. Він використовується для підтримки прийняття рішень на етапах концептуального та попереднього проектування. Застосовує прості, малодеталізовані моделі для мінімізації технічних ризиків і виявлення потенційних проблем на ранніх етапах дизайну. Автоматизований обмін даними відсутній; використовуються віртуальні прототипи, які не впливають безпосередньо на кінцеву систему.

Рівень «*Digital twin*» — цифровий двійник, який інтегрує дані про ефективність, технічний стан та обслуговування від фізичного аналога. Він підтримує прийняття рішень на різних етапах проектування та експлуатації. Використовує більш деталізовані моделі, що отримують пакетні оновлення

від фізичної системи, відображаючи її поточний стан і графік обслуговування. Це дозволяє двосторонню взаємодію та дослідження різних сценаріїв, аналізуючи поведінку фізичної системи і використовуючи виявлені недоліки для вдосконалення моделі.

Рівень «*Adaptive digital twin*» передбачає наявність адаптивного інтерфейсу користувача. Використовує алгоритми машинного навчання для оновлення моделей на основі даних у режимі реального часу. Інтеграція даних у реальному часі та навчання з урахуванням користувацьких переваг покращують адаптивність моделі і її відповідність оперативним потребам. Це забезпечує ефективне оперативне управління, обслуговування та прийняття рішень, адаптуючись до потреб користувача та умов експлуатації.

Рівень «*Intelligent digital twin*» розширює можливості адаптивного цифрового двійника шляхом включення некерованого машинного навчання для виявлення закономірностей та обробки даних з високим ступенем автономності. Використовує передові алгоритми для аналізу детальних даних, виявлення патернів та автономного прийняття рішень на основі глибокого розуміння системи і її середовища. Неперервне автономне навчання з операційного середовища дозволяє точно аналізувати дані для прогнозування та оптимізації системи. Це дає змогу проактивно оптимізувати систему, планувати обслуговування та автономно адаптуватися до експлуатаційних умов.

Тепер розглянемо складові цифрового двійника, які охоплюють широкий спектр компонентів і разом формують повноцінну віртуальну репрезентацію фізичного об'єкта або процесу.

Моделі та симуляції охоплюють як фізичні, так і поведінкові аспекти об'єкта [19]. Фізичні моделі описують його геометрію, масу, матеріали та інші властивості, а поведінкові моделі визначають, як об'єкт функціонує, реагує на зовнішні впливи, взаємодіє з оточенням та як працюють його системи контролю. Застосування симуляцій дозволяє відтворювати

поведінку об'єкта в різних сценаріях, аналізувати реакції та оптимізувати проєктування, експлуатацію чи обслуговування.

Дані, які є основою динаміки цифрового двійника, надходять з різних джерел. Це можуть бути дані з сенсорів, що постійно оновлюють моделі, оперативні відомості про використання, продуктивність та стан обслуговування, а також історична інформація, яка допомагає аналізувати тенденції, патерни та прогнозувати майбутній розвиток [20].

Інтеграція відбувається через спеціальні технології, які забезпечують інтерфейси для обміну даними між цифровим двійником та фізичним об'єктом. Спеціалізовані платформи для цифрових двійників об'єднують різноманітні моделі, дані та аналітичні інструменти в єдину систему.

Аналітика в межах цифрового двійника охоплює моніторинг поточного стану [22], детальний аналіз на основі різних метрик та алгоритмів, а також прогнозування, що ґрунтується на історичних і поточних даних. У результаті користувач отримує змогу контролювати та управляти системою через інтерфейс, який забезпечує наочну візуалізацію даних та інтуїтивно зрозумілий механізм взаємодії з цифровим двійником, задаючи параметри симуляцій, коригуючи аналітичні процеси та впливаючи на поведінку об'єкта.

1.3. Аналіз досліджень та відомих рішень

Більшість наукових робіт, які стосуються технології цифрових двійників, присвячено основним аспектам цієї технології, її застосуванням у різних галузях, перевагами, які вона пропонує, та питаннями організації її впровадження. Цей фокус на ширші теми можна пояснити відносною новизною дослідницького напрямку цифрових двійників, що спонукає вчених концентруватися на більш загальних питаннях, пов'язаних з розробленням та розумінням основної концепції цифрових двійників.

Конкретні дослідження щодо застосування цифрових двійників у медичній сфері також наявні [23], однак, незважаючи на значний потенціал, кількість досліджень, які зосереджені на їх використанні у цій сфері, залишається обмеженою. Це можна пояснити кількома ключовими факторами які будуть наведені нижче.

Комплексність біологічних систем. Технологія цифрових двійників має забезпечувати адекватність подання інформації про біологічні системи та їхні компоненти, а також розроблення високоточних моделей окремих

Дані великої розмірності. Медичні дані часто характеризуються великою розмірністю та об'ємом, включаючи генетичну інформацію, медичні записи, зображення та результати лабораторних тестів [25]. Збір, оброблення та інтеграція цих даних у цифрові двійники представляє собою значний виклик.

Питання конфіденційності та етики. Використання медичних даних для створення цифрових двійників порушує питання конфіденційності пацієнтів і етичних стандартів; це вимагає розроблення чітких протоколів збору даних та їх використання [27].

Високі вимоги до точності та надійності. У медичній галузі помилки можуть мати серйозні наслідки. Це висуває високі вимоги до точності, надійності та валідації цифрових двійників перед їх застосуванням у клінічній практиці [28].

Регуляторні обмеження. Медична галузь є однією з найбільш регульованих, і нові технології, такі як цифрові двійники, повинні пройти строгу процедуру сертифікації та схвалення перед використанням [29].

Розглянемо дослідження та технології, що застосовують цифрові двійники поза медичною галуззю.

У дослідженні [30] представлено комплексну архітектуру цифрового двійника, розділену на шість основних рівнів.

Базовий рівень являє собою фізичний об'єкт (фізичний двійник), оснащений датчиками для збору основних характеристик та актуаторами для взаємодії з ним. Локальні контролери відповідають за певні функції цифрового двійника, а локальні бази даних зберігають дані, зібрані на попередніх рівнях. Комунікаційні засоби забезпечують обмін даними між локальними та хмарними сховищами, тоді як хмарне сховище даних слугує централізованим сховищем для подальшого аналізу. Завдяки емулюванню та симуляції моделюється поведінка фізичного двійника як у поточний момент, так і в майбутньому.

Такий підхід дозволяє завчасно виявляти потенційні несправності, перевіряти вплив змін у параметрах експлуатації та вчасно коригувати стратегії управління. Усе це, зрештою, допомагає мінімізувати ризики, скоротити витрати на тестування і підвищити надійність роботи системи в цілому.

Дослідження [31] описує абстрактну модель цифрового двійника, яка надає концептуальний опис фізичного об'єкта на всіх етапах його життєвого циклу. Проте ця модель не передбачає детального представлення всіх характеристик об'єкта.

Тепер перейдемо до проблеми збереження даних цифрового двійника, для цього можуть використовуватися різні формати, наприклад такі як на XML [35-36], дозволяє архівувати інформацію про моделі промислових об'єктів, представляючи об'єкт у вигляді ієрархічної структури його компонентів. Така модель може слугувати для обміну даними в межах програмної платформи. Для зберігання інформації про моделі промислових об'єктів також можна використовувати формат B2MML [37], який також базується на XML і стандарті IEC/ISO 62264 [38]. B2MML використовує набір схем XML, створених з використанням мови XSD [39], для представлення даних.

Аналіз наявної літератури [40-44] дає змогу зробити висновок про відсутність сформульованих теоретичних принципів у сфері розробки цифрових двійників, що в основному зумовлено ранньою стадією розвитку даного дослідницького напрямку. Також проблемою може бути повсякчасне застосування формату XML, який є дещо застарілим форматом даних. Основні проблеми його використання включають наступне.

Великий обсяг даних. XML файли часто містять багато додаткової метаінформації та тегів, що збільшує обсяг даних [45]. Для цифрових двійників, які залежать від швидкого обміну та обробки даних, це може призвести до зайвих затримок і підвищення вимог до зберігання.

Складність оброблення. Читання та аналіз XML файлів вимагає відповідних парсерів та може бути ресурсомістким процесом, особливо для великих документів. Це може уповільнити виконання програм, що працюють з цифровими двійниками, та підвищити витрати на обчислювальні ресурси [46].

Обмеження у представленні даних. XML має обмеження у представленні складних типів даних або дуже великих об'ємів даних, що може бути недостатньо для деяких випадків використання цифрових двійників, де потрібна висока точність та глибина аналізу.

Як бачимо, методи, які зараз існують, не можна вважати всезагальними; вони зосереджені на специфічних прикладах використання та спрямовані на розроблення цифрового двійника для окремих випадків застосування [47].

Також однією з потенційних проблем можна вважати складність інтеграції технології цифрових двійників з існуючими медичними системами. Однією з основних проблем є несумісність даних між різними медичними системами та пристроями. Це вимагає додаткових зусиль для конвертації та адаптації даних, що може призвести до затримок у роботі та потенційної втрати важливої інформації.

Крім того, інтероперабельність медичних інформаційних систем становить ще один важливий бар'єр [48]. Відсутність єдиних стандартів для обміну даними між системами ускладнює взаємодію цифрових двійників з іншими компонентами медичної інфраструктури, обмежуючи їхню здатність до швидкої та ефективної інтеграції.

Питання конфіденційності та безпеки даних також відіграють критичну роль, оскільки інтеграція різних систем вимагає обміну чутливою інформацією. Забезпечення захисту цих даних під час передачі та зберігання є ключовим викликом, що вимагає впровадження комплексних механізмів безпеки [49].

На додаток до цього, технічні обмеження та відсутність необхідної інфраструктури можуть перешкоджати впровадженню цифрових двійників у медичні процеси. Наприклад, високі вимоги до обчислювальних ресурсів для моделювання складних біологічних процесів потребують значних інвестицій у відповідне апаратне забезпечення [50].

Таким чином, хоча цифрові двійники відкривають нові перспективи для медицини, сучасний стан розвитку медичних технологій стикається з низкою складнощів, що вимагають комплексного підходу до вирішення.

Тепер розглянемо програмні рішення, які можуть бути використані для втілення технології цифрових двійників.

Watson™ IoT Platform [51] слугує прикладом програмної платформи, яка частково підтримує функціонал цифрових двійників. Ця платформа дозволяє збирати дані з різноманітних датчиків, підтримуючи кілька форматів даних, включно з JSON, текстовими форматами та форматами для геопросторових даних. Вона пропонує три способи зберігання даних: в реальному часі в NoSQL базах даних, для аналізу в хмарному сховищі типу Data Lake та для довготривалого зберігання в хмарному сховищі об'єктів. Однак, ця платформа не має компонентів для створення та візуалізації цифрового двійника.

] розроблено для підтримки впровадження цифрових двійників у виробничому секторі. Це програмне забезпечення спрямоване на моделювання технологічних ліній, дозволяючи створювати графічні прототипи промислових продуктів та моделювати їх виробництво для ідентифікації потенційних проблем. Однак, його можливості щодо взаємодії з фізичними об'єктами обмежені, і воно призначене виключно для виробничої сфери. Спеціалізоване програмне забезпечення компанії Simio дозволяє створювати візуальні симуляційні моделі цифрових двійників для промислового використання. Воно включає стандартну бібліотеку об'єктів і редактор для налаштування симуляції, пропонуючи розширені можливості для опису фізичних властивостей об'єктів з використанням різноманітних типів даних.

] представляє собою платформу з найбільш розвинутими функціями для створення цифрових двійників, використовуючи мову DTDL для детального опису моделей. Ця платформа дозволяє визначати поведінку цифрових двійників через п'ять класів метамоделі, включаючи телеметрію, властивості, команди, взаємозв'язки та компоненти, забезпечуючи багатогранне відображення фізичного об'єкта та його взаємодії з іншими об'єктами. Також вона надає уніфікований формат для праці з мультимодальними даними цифрових двійників – DTDL [47]. Приклад представлення даних у форматі DTDL наведено нижче у лістингу 1.1.

При розробленні цифрового двійника використання семантичних анотацій допомагає упростити обробку даних. Мова DTDL включає семантичні типи, такі як Acceleration (прискорення), Density (густина), Force (сила), для позначення даних телеметрії та характеристик.

Основні дії, які можна виконувати з цифровими двійниками, охоплюють їх створення, оновлення та видалення, використовуючи API DigitalTwins на рівні управління та даних. API управлінського рівня

служать для керування загальним екземпляром Azure Digital Twins, включаючи його ініціалізацію та видалення, тоді як API рівня даних дозволяють керувати конкретними елементами в рамках Azure Digital Twins, підтримуючи різноманітні дії, такі як керування моделями.

Лістинг 1.1. – Приклад представлення даних у форматі DTDL

```
{
  "@id": "dtmi:com:adt:dtsample:v2sensor;1",
  "@type": "Interface",
  "@context": "dtmi:dtdl:context;2",
  "displayName": "Sensor (v2 model)",
  "contents": [
    {
      "@type": ["Property", "Temperature"],
      "name": "Temperature",
      "schema": "double",
      "unit": "degreeFahrenheit"
    },
    {
      "@type": ["Property", "Humidity"],
      "name": "Humidity",
      "schema": "double",
      "unit": "gramPerCubicMetre"
    }
  ]
}
```

Для інтеграції цифрових двійників окремих компонентів складного об'єкта, наприклад виробничої лінії, може застосовуватися графова модель. Azure Digital Twins Explorer — це інструмент [55], призначений для роботи з графовим представленням, що дозволяє завантажувати моделі, налаштовувати параметри та створювати зв'язки між компонентами.

Дані, зібрані з цифрових двійників, зберігаються в Azure Data Lake, а аналіз цих даних здійснюється за допомогою Azure Synapse Analytics [57]. Azure Digital Twins спрямовано на моделювання з метою оптимізації робочих процесів великих та складних систем, таких як заводи, ферми, стадіони.

Серед інших програмних платформ, що підтримують або сприяють використанню технології цифрових двійників, можна виділити Digital Twin

зосереджені на виробничій сфері та управлінні бізнес-процесами, не охоплюючи застосування в інших областях, таких як медицина чи освіта.

Отже, розгляд методів та програмних інструментів для відтворення та аналізу мультимодальних даних цифрових двійників виявляє декілька ключових недоліків у сучасних підходах та системах обробки цих даних.

1. Відомі програмні рішення переважно орієнтовані на інженерні та адміністративні задачі у секторі виробництва і не адаптовані для використання в інших сферах, таких як медицина.
2. Більшість програмних систем створюють візуальні моделі цифрових двійників з обмеженим рівнем реалізму, зосереджуючись в основному на двомірному графічному представленні фізичних об'єктів. Інші види мультимодальної інформації зазвичай не враховуються.
3. Не існує універсального, системного підходу до представлення і обробки часових мультимодальних даних для цифрових двійників, що досліджуються, а також до розроблення програмного забезпечення на основі концепції цифрового двійника для широкого спектру застосувань. Можливості концепції цифрових двійників не використовуються в повному обсязі для аналізу та дослідження різноманітних об'єктів або процесів.

Визначимо вимоги до програмного забезпечення технології цифрових двійників, які потрібно рахувати при розробленні узагальненої архітектури програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів.

1.4. Вимоги до системи створення цифрових двійників медико-біологічних об'єктів

Для успішного створення функціонального та високоякісного програмного продукту, призначеного для обробки даних про медико-біологічні об'єкти з використанням технології цифрових двійників, необхідно спочатку розробити чіткі вимоги до програмного забезпечення. Цей процес є окремою технологічною процедурою, яка включає власні етапи, методи та особливості реалізації. Кінцевим результатом є підготовлена специфікація вимог до програмного забезпечення системи, яку планується розробити.

Вимоги до програмного забезпечення – це сукупність потреб потенційних користувачів та зацікавлених сторін щодо функціональності, якості та характеристик майбутнього програмного продукту. У контексті медико-біологічних досліджень та цифрових двійників, такі вимоги можуть охоплювати специфічні функції, як-от моделювання фізіологічних процесів, інтеграція з медичними приладами та забезпечення високої точності симуляцій. Розроблення вимог зазвичай складається з кількох етапів: ідентифікація вимог (збір та аналіз потреб медичних фахівців та дослідників), аналіз вимог, документування (створення специфікації) та перевірка вимог [62].

Виділяють три рівні вимог до програмного забезпечення.

Бізнес-вимоги визначають основну мету програмного забезпечення, ключові ідеї, що лежать в основі розробки, та стратегічні цілі, наприклад, покращення діагностики чи лікування завдяки використанню цифрових двійників.

Вимоги користувача описують перелік завдань, які користувачі мають виконувати за допомогою програмного забезпечення, та сценарії їх реалізації. Наприклад, можливість моделювати реакцію організму на певний лікарський препарат або проводити віртуальні хірургічні операції.

Функціональні вимоги деталізують конкретні функції, які повинен виконувати програмний продукт. Вони описуються у документі під назвою

"Специфікація вимог до програмного забезпечення" і включають такі аспекти, як оброблення медичних зображень чи симуляція фізіологічних процесів.

Аналіз вимог є критичним етапом у розробці програмного забезпечення. Він полягає у виявленні та узгодженні потреб користувачів і замовників, а також умов, що висувуються до розроблюваного продукту. Цей аналіз має бути ретельним і об'єктивним, щоб виявити та виправити можливі недоліки у формулюванні вимог, такі як неточності, неповнота, неоднозначності чи суперечності. Вимоги повинні бути достатньо деталізованими для розроблення системи, а також вимірюваними, тестованими та відповідати реальним потребам медичної практики.

Специфікація вимог до програмного забезпечення—це спеціалізований документ, що містить повний опис поведінки системи, яку розробляють. Специфікація включає набір функціональних вимог, які описують можливі взаємодії користувачів із програмним забезпеченням, а також нефункціональні (додаткові) вимоги, такі як безпека даних, продуктивність та сумісність з існуючими медичними системами.

Структура специфікації зазвичай містить три основні розділи, кожен із яких виконує власну функцію. У вступі надається загальна інформація про продукт, викладається його мета та описуються взаємозв'язки з іншими системами. Розділ «Загальний» охоплює опис продукту, включно з перспективами розвитку, функціями програмного забезпечення, характеристиками потенційних користувачів та загальними обмеженнями. Нарешті, у розділі «Деталізація вимог» визначаються вимоги до зовнішніх інтерфейсів (зокрема, користувацького інтерфейсу, інтеграції з медичними приладами та протоколів передачі даних), формулюються властивості програмного продукту, а також окреслюються вимоги до бази даних і обробки великих обсягів інформації.

Чітко сформульована специфікація вимог допомагає знизити загальні витрати та час розроблення, оскільки дозволяє уникнути переробок через зміни вимог і забезпечує відповідність продукту потребам медичної галузі. Для однозначного визначення вимог недостатньо лише текстового опису; їх доцільно доповнювати набором атрибутів. Прикладами таких атрибутів можуть бути дата створення, автор, статус вимоги, можливість перевірки, метод перевірки, пріоритет реалізації тощо. У контексті медичного програмного забезпечення особливу увагу слід приділити атрибутам, пов'язаним із безпекою та конфіденційністю даних пацієнтів.

Якісно сформульовані вимоги та правильно визначені набори атрибутів для кожної категорії дозволяють підвищити ефективність роботи команди проєкту на всіх етапах його реалізації: від проєктування до тестування та впровадження. Зокрема, це сприяє зменшенню помилок, оптимізації витрат, скороченню термінів розробки та підвищенню загальної якості продукту.

Використання чітких, зрозумілих і стандартизованих атрибутів допомагає уникнути неоднозначностей у вимогах, полегшує інтеграцію компонентів системи та забезпечує можливість масштабування продукту в майбутньому. Це особливо важливо в контексті медичних систем, де точність, надійність та відповідність галузевим стандартам є критично важливими для забезпечення безпеки пацієнтів і підтримки високого рівня довіри до системи. Нижче, у таблиці 1.1. наведено формальне представлення атрибутів вимог, які доцільно використовувати під час розроблення медичних систем.

Таблиця 1.1. – Категорії атрибутів, які доцільно використовувати під час розроблення вимог до ПЗ

Категорія атрибутів	Приклади значень атрибутів
Ідентифікація вимоги	
Ідентифікатор	Унікальний номер вимоги (наприклад, "REQ-001")
- Назва	Коротка унікальна назва, що точно характеризує вимогу (наприклад, "Моделювання серцевого ритму")
Внутрішні характеристики	
- Основний тип	Функціональність, продуктивність системи, якість моделювання, середовище використання, інтерфейс, обмеження
- Підтип якості	Доступність, гнучкість, цілісність даних, можливість масштабування, легкість підтримки, зручність використання, відповідність стандартам
- Тип продукту/процесу	Програмний модуль, процес обробки даних, сервіс моделювання, етап життєвого циклу проєкту

Продовження таблиці 1.1.

Пріоритет вимоги	
- Пріоритет	Критична, необхідна, бажана, необов'язкова; або обов'язкова (Must), рекомендована (Should), можлива (Could), бажана (Wish)
Джерело та автор вимоги	
- Джерело	Назва документа (наприклад, "Клінічні протоколи"), ім'я зацікавленої особи або організації
- Автор	Ім'я особи, яка сформулювала вимогу (наприклад, "д-р Іваненко")
Інші показники	
- Зрілість (стабільність)	Кількість змін, тривалість стабільності вимоги
- Рівень ризику	Високий, середній, низький
- Оцінювана вартість	Вартість реалізації вимоги, оцінена експертами
- Фактична вартість	Реальна вартість реалізації вимоги
- Реліз продукту	Версія програмного продукту, в якій вимога буде реалізована (наприклад, "Версія 1.0")

Вимоги до програмного забезпечення, супроводжені наборами атрибутів, повинні надавати команді розробників такі можливості.

Ідентифікація кожної вимоги – забезпечення унікального визначення для кожного положення, щоб уникнути плутанини та забезпечити прозорість процесу розроблення.

Класифікація вимог – можливість розподіляти вимоги за важливістю реалізації (пріоритетом), терміновістю виконання (дати) та іншими критеріями, що впливають на планування проєкту.

Відстеження статусу вимог – моніторинг статусів аналізу, виконання, задоволення та перевірки кожної вимоги для ефективного управління проєктом.

Оцінка стратегії тестування – можливість оцінювати кожну вимогу з точки зору її тестування та забезпечення якості.

Для розроблення програмного забезпечення, призначеного для моделювання та аналізу медико-біологічних об'єктів із використанням технології цифрових двійників, необхідно визначити **функціональні та нефункціональні** вимоги до системи.

Функціональні вимоги визначають перелік функцій та сервісів, які має забезпечувати програмне забезпечення для задоволення потреб користувачів – медичних працівників, дослідників та інших зацікавлених сторін – у контексті моделювання та аналізу медико-біологічних об'єктів.

Це можуть бути функції, такі як:

- створення та управління цифровими двійниками органів чи систем організму;
- моделювання фізіологічних процесів та симуляція реакцій на медичні втручання;
- інтеграція з медичними пристроями та базами даних для отримання актуальної інформації;

- візуалізація результатів моделювання у зрозумілому для користувача форматі.

Нефункціональні вимоги описують додаткові характеристики програмної системи, які не пов'язані безпосередньо з її функціональністю, але є критичними для успіху проєкту:

- надійність – система повинна працювати без збоїв та забезпечувати коректність результатів моделювання;
- безпека – захист конфіденційних медичних даних від несанкціонованого доступу;
- продуктивність – ефективне використання ресурсів та швидка оброблення великих обсягів даних;
- масштабованість – можливість розширення функціоналу та обробки більшої кількості даних без суттєвого переписування системи;
- сумісність – інтеграція з існуючими медичними інформаційними системами та обладнанням.

Функціональні вимоги до програмної системи для створення цифрових двійників медико-біологічних об'єктів наведені в таблиці 1.3. У цьому розділі ми зосередимося саме на них, залишаючи детальний аналіз нефункціональних вимог для наступних етапів проєкту.

У розроблюваному програмному забезпеченні важливість кожної функціональної вимоги будемо визначати за допомогою атрибуту пріоритету. Це необхідно для ефективного планування та розподілу ресурсів, оскільки обсяг людських, технічних та фінансових ресурсів проєкту може бути обмеженим. Пріоритезація вимог допоможе оптимізувати час і вартість розроблення, а також максимізувати користь від впровадження системи.

Для визначення пріоритетності вимог пропонується використати ртості та цінності (**Cost-Value Analysis**) [64].

Цей метод передбачає оцінювання кожної вимоги за двома ключовими параметрами, вартістю реалізації (Cost) та цінністю для користувача або бізнесу (Value).

Вартість реалізації охоплює ресурси, необхідні для виконання вимоги, зокрема фінансові витрати, часові рамки та технічну складність, а також інші фактори, що впливають на обсяг роботи. Цінність вимірюється ступенем, до якої реалізація вимоги сприятиме досягненню стратегічних цілей і задовольнить потреби користувачів, наприклад, шляхом покращення якості медичної допомоги, підвищення ефективності роботи медичного персоналу чи підтримки наукових досліджень.

Процес оцінки полягає у визначенні вартості та цінності для кожної вимоги. На етапі аналізу вартості експертна команда оцінює, які ресурси будуть необхідні для її реалізації, враховуючи складність завдання, потребу у залученні додаткових фахівців і можливість використання нових технологій або обладнання. Під час визначення цінності встановлюється, наскільки важливою є вимога з точки зору досягнення загальних цілей проєкту та користі для кінцевих користувачів, включно з впливом на якість медичних послуг, зручність використання системи й потенційні конкурентні переваги.

Кожній вимозі присвоюється числове значення вартості та цінності, наприклад, за шкалою від 1 до 10, де 1 – мінімальна вартість або цінність, а 10 – максимальна, результати порівняльного аналізу представлено нижче у таблиці 1.2.

Таблиця 1.2. – Функціональні вимоги до програмної системи цифрових двійників медико-біологічних об'єктів

Код вимоги	Вимога			Пріоритет
ФВ1	Імпорт даних з медичних приладів та систем			Високий
ФВ2	3D-моделювання органів на основі медичних зображень			Високий
ФВ3	Інтеграція з електронними медичними картками пацієнтів			Високий
ФВ4	Симуляція фізіологічних процесів у реальному часі			Високий
ФВ5	Візуалізація результатів моделювання у зручному інтерфейсі			Середній
ФВ6	Аналітичні інструменти для прогнозування перебігу захворювань			Середній
ФВ7	Налаштування моделей під індивідуальні особливості пацієнта			Високий
ФВ8	Генерація звітів та документації на основі результатів моделювання			Середній
ФВ9	Підтримка багатомовності інтерфейсу користувача			Низький
ФВ10	Можливість експорту даних у різні формати для подальшого аналізу			Середній
ФВ11	Інтеграція з хмарними сервісами для зберігання та обробки даних			Високий

Продовження таблиці 1.2

ФВ12	Підтримка мобільних пристроїв для доступу до системи			Середній
ФВ13	Автоматичне оновлення моделей на основі нових даних			Високий
ФВ14	Інтеграція з системами віртуальної та доповненої реальності			Середній
ФВ15	Моделювання впливу лікарських препаратів на організм			Високий
ФВ16	Підтримка телемедицини та віддаленого моніторингу пацієнтів			Високий
ФВ17	Інтеграція з науковими базами даних для оновлення моделей та алгоритмів			Високий

Більш детально сформулюємо визначення до обраних оцінок.

ФВ1: вартість реалізації висока через складність інтеграції з різноманітними медичними приладами, але цінність також висока, оскільки це забезпечує актуальні дані для моделей.

ФВ2: найвища цінність, оскільки 3D-моделювання є основою системи; висока вартість через технічну складність.

ФВ3: середня вартість і висока цінність завдяки покращенню точності та інформативності моделей.

ФВ4: висока цінність через можливість реального часу симуляцій; вартість висока через складність реалізації.

ФВ5: середня вартість; цінність висока для користувацького досвіду, але може бути впроваджена після основних функцій.

ФВ6: висока цінність для медичного прогнозування; вартість реалізації значна через складність аналітики.

ФВ7: індивідуалізація моделей підвищує точність; середня вартість реалізації.

ФВ8: генерація звітів має середню цінність; низька вартість реалізації.

ФВ9: підтримка багатомовності корисна для розширення аудиторії; вартість невелика.

ФВ10: експорт даних важливий для подальшого аналізу; середня вартість.

ФВ11: хмарні сервіси підвищують масштабованість; вартість висока.

ФВ12: доступ з мобільних пристроїв покращує зручність; середня вартість.

ФВ13: автоматичне оновлення моделей забезпечує актуальність даних; висока вартість.

ФВ14: інтеграція VR/AR технологій додає інноваційності; вартість дуже висока.

ФВ15: моделювання впливу ліків критично для медицини; висока цінність і вартість.

ФВ16: телемедицина розширює можливості системи; висока цінність, середня вартість.

ФВ17: інтеграція з науковими базами даних підтримує актуальність моделей; висока цінність, середня вартість.

Ці вимоги є важливими для процесу розроблення узагальненої архітектури програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів.

1.5. Висновки до розділу 1

У першому розділі дисертації проведено аналіз наявних технічних та програмних рішень для обробки даних про медико-біологічні об'єкти, з особливим фокусом на цифрових двійниках. Виявлено, що попри значний потенціал цих технологій у медицині, розвиток теоретичних основ та методологій ще перебуває на початковому етапі. Виклики, пов'язані з обробкою та зберіганням даних, складністю інтеграції систем та

забезпеченням безпеки інформації, ускладнюють ефективне впровадження цифрових двійників у медичну практику.

Окрім того, встановлено відсутність універсальних підходів для представлення та обробки мультимодальних даних, що обмежує використання цифрових двійників для детального дослідження медико-біологічних об'єктів. Це вимагає розроблення нових інноваційних рішень, здатних подолати існуючі технічні та методологічні бар'єри.

Таким чином, можна зробити загальний висновок про необхідність подальших досліджень для розроблення ефективних, безпечних та інтегрованих рішень для цифрових двійників у медицині, що включає розвиток стандартів для обміну даними, покращення методів аналізу та моделювання, а також забезпечення сумісності з існуючими медичними системами.

РОЗДІЛ 2. РОЗРОБЛЕННЯ МЕТОДІВ СИНХРОНІЗАЦІЇ ТА АНАЛІЗУ ТЕМПОРАЛЬНИХ МУЛЬТИМОДАЛЬНИХ ДАНИХ

Для створення цифрових двійників медико-біологічних об'єктів необхідний інструментарій, що забезпечує логічне представлення та обробку темпоральних мультимодальних даних на рівнях математики, алгоритмів і програмного забезпечення. Цей підхід має враховувати специфічні особливості таких даних, які визначають їх унікальність і складність обробки. Основними аспектами є мультимодальність і темпоральність [65], які формують основу для побудови математичних моделей і алгоритмів оброблення.

Мультимодальність даних передбачає інтеграцію різнотипної інформації, отриманої з різних джерел, таких як відео, аудіо, сенсори та результати лабораторних досліджень. Ця різноманітність вимагає розробки гнучких підходів до представлення та обробки даних, які можуть адаптуватися до різних форматів і характеристик. Крім того, необхідно враховувати можливі варіації в якості даних, наприклад, різну роздільну здатність відео або неоднорідність сигналів сенсорів, що робить завдання інтеграції ще складнішим.

Темпоральність даних забезпечує можливість аналізу змін об'єкта в часі, що є вирішальним для розуміння динаміки його поведінки. Послідовність подій, зафіксована у темпоральних даних, дозволяє виявляти тренди, аномалії та закономірності, які можуть впливати на висновки досліджень. Наприклад, у цифрових двійниках гортані темпоральність дає змогу відстежувати зміну частоти коливань голосових зв'язок у часі, що може вказувати на прогресування певної патології. Впорядкований характер даних є критично важливим для алгоритмів обробки, які враховують залежність результатів від їхньої хронологічної послідовності.

На основі зазначених характеристик необхідно створювати спеціалізовані математичні моделі, які дозволяють формалізувати мультимодальні темпоральні дані. Ці моделі повинні враховувати як особливості структури кожного окремого типу даних, так і взаємозв'язки між ними. Наприклад, графові моделі можуть бути використані для представлення зв'язків між різними джерелами даних, де вузли графа представляють окремі модальності, а ребра — залежності між ними, такі як часові кореляції або спільні анатомічні характеристики. Для кожного вузла графа, який представляє окрему модальність, можуть бути визначені атрибути, що характеризують його властивості.

Для аналізу таких моделей можна використовувати графові нейронні мережі (GNN) [66], які здатні інтегрувати інформацію з різних вузлів, враховуючи їхні зв'язки, та виявляти складні залежності. Наприклад, GNN дозволяють прогнозувати нові залежності між вузлами, такі як можливий вплив частоти вібрацій голосових зв'язок на параметри аудіосигналу, або виявляти аномальні зв'язки, що можуть вказувати на технічні помилки або патологічні зміни.

Такий комплексний підхід, що поєднує спеціалізовані математичні моделі, адаптивні алгоритми та інструменти програмного забезпечення, стане ключем до створення точних, функціональних і надійних цифрових двійників. Вони зможуть відтворювати поведінку реальних об'єктів з високою точністю, дозволяючи моделювати їхню динаміку, прогнозувати розвиток патологій та оптимізувати підходи до лікування. Це відкриває нові можливості у медичних дослідженнях і персоналізованій медицині, роблячи цифрові двійники незамінним інструментом для аналізу складних біологічних систем.

Синхронізація таких даних також є основою для їх подальшого аналізу. Це включає вирівнювання часових міток, усунення затримок між сигналами різної природи та інтеграцію інформації з декількох джерел у

єдину модель. Без належної синхронізації неможливо досягти точного аналізу, а тим більше — створення функціональних цифрових двійників.

Разом із синхронізацією стає очевидною потреба в додатковій систематизації отриманої інформації. Мультиmodalьні дані не лише взаємодіють у часовій шкалі, але й мають складну семантику, яка визначає їхній контекст, залежності та взаємозв'язки. Саме тому одним з етапів синхронізації стає *семантичний аналіз*, який дозволяє впорядкувати дані, знайти приховані зв'язки між ними та забезпечити їх інтелектуальну інтеграцію у загальну інформаційну систему. Семантичний аналіз не лише покращує якість обробки, але й відкриває нові можливості для створення цифрових двійників, здатних адаптуватися до динамічних змін у даних.

Семантичний аналіз метаданих є одним із ключових інструментів для створення високоточних цифрових двійників у медико-біологічних системах. Завдяки своїй здатності виявляти приховані зв'язки між даними, цей підхід дозволяє не лише впорядковувати інформацію, а й робити складні прогнози та оптимізувати процеси аналізу. Особливо актуальним він стає у випадках, коли дані є мультиmodalьними (відео, аудіо, параметри сенсорів) і темпоральними (відображають динаміку змін).

У цьому розділі розглянуто ключові аспекти синхронізації темпоральних мультиmodalьних даних та їх інтеграції у єдину систему. Основна увага приділяється методам організації часових міток, обробці метаданих та їхньому семантичному аналізу. Це дозволить забезпечити структурованість даних, врахувати динаміку їхніх змін і встановити взаємозв'язки між різними джерелами інформації. Отримані результати спрямовані на створення основи для високоточного аналізу, моделювання та побудови цифрових двійників медико-біологічних об'єктів.

2.1. Організація темпоральних даних медико-біологічних об'єктів

Темпоральні дані є ключовим компонентом у моделюванні цифрових двійників, оскільки вони відображають часову динаміку змін у стані об'єкта дослідження. Ефективна організація та представлення таких даних потребує дотримання специфічних форматів та структур, які забезпечують логічність, сумісність та зручність подальшої обробки.

Одним із найпоширеніших способів організації темпоральних даних є представлення їх у вигляді часових рядів. Часовий ряд — це впорядкована сукупність даних, де кожна точка прив'язана до конкретного моменту часу. Такий підхід дозволяє не лише фіксувати стан об'єкта у певний момент, але й аналізувати динаміку його змін, виявляти закономірності, робити прогнози та ідентифікувати ключові події, які можуть бути критичними для дослідження. Наприклад, часові ряди використовуються для аналізу частоти вібрацій голосових зв'язок, моніторингу змін амплітуди або інших фізіологічних параметрів.

Для представлення часових рядів найчастіше застосовуються формати CSV (лістинг 2.1) або таблиці, у яких кожен рядок відповідає певному часовому моменту, а стовпці містять значення параметрів. Це дозволяє легко працювати з такими даними за допомогою сучасних інструментів для обробки, таких як Python. Крім того, часові ряди можуть бути збережені у форматах, що підтримують додаткові метадані, наприклад, JSON або HDF5 [67], що дозволяє інтегрувати їх із іншими джерелами даних.

Лістинг 2.1. Приклад використання CSV-формату для даних про динаміку роботи голосових зв'язок

```
timestamp,vibration_frequency,amplitude
2024-12-10T10:00:00Z,125,0.85
2024-12-10T10:01:00Z,130,0.88
2024-12-10T10:02:00Z,128,0.86
```

Цей формат дозволяє легко зберігати та аналізувати базові параметри роботи голосових зв'язок, однак для складніших структур даних або додаткової інформації (наприклад, про стан пацієнта чи налаштування апаратури) доцільніше використовувати формат JSON, який наведено на лістингу 2.2.

Лістинг 2.2. Приклад представлення тих самих даних у форматі JSON

```
{
  "patient_id": "56789",
  "measurements": [
    {
      "timestamp": "2024-12-10T10:00:00Z",
      "vibration_frequency": 125,
      "amplitude": 0.85,
      "metadata": {
        "device": "LaryngoScope-X",
        "operator": "Dr. Ivanenko"
      }
    },
    {
      "timestamp": "2024-12-10T10:01:00Z",
      "vibration_frequency": 130,
      "amplitude": 0.88,
      "metadata": {
        "device": "LaryngoScope-X",
        "operator": "Dr. Ivanenko"
      }
    },
    {
      "timestamp": "2024-12-10T10:02:00Z",
      "vibration_frequency": 128,
      "amplitude": 0.86,
      "metadata": {
        "device": "LaryngoScope-X",
        "operator": "Dr. Ivanenko"
      }
    }
  ]
}
```

Формат JSON забезпечує кращу структурування, дозволяючи додавати додаткові атрибути, такі як модель приладу, ім'я оператора чи інші метадані. Це особливо важливо для медичних досліджень, де потрібна точна ідентифікація умов збору даних.

Тепер проаналізуємо дані, отримані у попередньому пункті даного розділу. Дані, отримані після попередньої обробки, можуть бути

організовані у вигляді набору векторів ознак V , де кожен вектор v_i представляє інформацію про певний кадр відео. Ці вектори містять атрибути, які характеризують об'єкт, наприклад, площу сегментованої області, периметр, коефіцієнт форми або інші ключові параметри, що мають значення для подальшого аналізу. Формально це можна описати як наведено у формулі (2.1).

$$V = \{v_1, v_2, \dots, v_n\}, \quad (2.1)$$

де v_i — вектор ознак для i -го кадру.

Кожен атрибут вектора v_i може бути підданий статистичній обробці, наприклад, розрахунку середнього значення, стандартного відхилення, мінімуму, максимуму чи інших показників. Це дозволяє звести набір даних до компактнішої форми, зберігаючи при цьому основну інформацію, необхідну для подальшого аналізу або моделювання.

Отже, для зберігання даних пропонується використати формат JSON, легкий формат обміну даними, який легко читається людиною та машинами. JSON дозволяє ефективно структурувати та зберігати інформацію у вигляді вкладених об'єктів і масивів, що особливо зручно для роботи з мультимодальними темпоральними даними. Його гнучкість дозволяє інтегрувати часові мітки, метадані та параметри з різних джерел в одну структуру, забезпечуючи при цьому зручність імпорту та експорту для подальшого аналізу. Приклад представлення темпоральних мультимодальних даних у форматі JSON наведено у лістингу 2.3.

Лістинг 2.3. Приклад представлення темпоральних мультимодальних даних у форматі JSON

```
{
  "video_id": "larynx_video_001",
  "timestamp": "2024-02-12T10:00:00Z",
  "frame_data": [
    {
      "frame_id": 1,
      "timestamp": "2024-02-12T10:00:00Z",
      "features": {
        "area": 145.67,
        "perimeter": 52.3,
        "shape_coefficient": 0.75
      }
    },
    {
      "frame_id": 2,
      "timestamp": "2024-02-12T10:00:01Z",
      "features": {
        "area": 150.12,
        "perimeter": 54.8,
        "shape_coefficient": 0.78
      }
    }
  ],
  "analysis_summary": {
    "mean_area": 147.9,
    "mean_shape_coefficient": 0.765,
    "total_frames": 2
  }
}
```

У цьому прикладі кожен кадр відео описаний окремим об'єктом із часовою міткою та набором характеристик, які забезпечують точний опис геометричних властивостей зображення, таких як площа сегментованої області (*area*), периметр (*perimeter*) та коефіцієнт форми (*shape_coefficient*). Ці характеристики дозволяють оцінити структурні зміни в динаміці та забезпечують важливу інформацію для діагностики. Крім того, структура даних включає зведену інформацію, яка представлена у вигляді статистичних показників, таких як середнє значення площі (*mean_area*), середнє значення коефіцієнта форми (*mean_shape_coefficient*) та загальна кількість кадрів (*total_frames*). Такий підхід спрощує подальший аналіз, дозволяючи швидко оцінювати основні параметри всієї вибірки, і є

важливим для ефективної обробки великих масивів даних. Отже, перевагами використання JSON є:

- простота у читанні й обробленні;
- легкість інтеграції з мовами програмування (Python, JavaScript);
- підтримка складних ієрархічних структур для метаданих.

Наприклад, середнє значення площі сегментованих областей або інших характеристик кадрів можна обчислити буквально за кілька рядків коду. Для демонстрації цього процесу на лістингу 2.4. наведено приклад обробки даних у Python, де дані завантажуються із JSON-файлу, а потім обчислюється середнє значення площі для всіх кадрів.

Лістинг 2.4. Оброблення даних у Python

```
import json

# Завантаження даних із JSON
with open('larynx_data.json', 'r') as file:
    data = json.load(file)

# Обчислення середнього значення площі
areas = [frame['features']['area'] for frame in data['frame_data']]
mean_area = sum(areas) / len(areas)
print(f"Середня площа: {mean_area:.2f}")
```

Обрана організація даних забезпечує не лише можливість аналізу, а й ефективну інтеграцію з іншими програмними системами, зокрема для створення цифрових двійників. Це також дозволяє легко масштабувати модель, додаючи нові ознаки чи методи обробки без змін у базовій структурі даних.

Також в такій галузі як медицина варто зазначити необхідність збереження конфіденційності даних завдяки їх *анонімізації*. Вона є важливим етапом роботи з темпоральними медичними записами, спрямованим на захист конфіденційності пацієнтів і відповідність міжнародним стандартам, таким як GDPR чи HIPAA [68]. Цей процес

дозволяє забезпечити безпеку даних, зберігаючи їх цінність для наукових досліджень та аналітики.

У медичних темпоральних даних анонімізація зазвичай передбачає видалення ідентифікаційної інформації, такої як ім'я пацієнта, адреса чи точна дата народження. Ці дані замінюються унікальними ідентифікаторами, які не дозволяють безпосередньо ідентифікувати особу. Наприклад, замість імені використовується псевдонім або код: `"patient_id": "ANON_12345"`. Такий підхід забезпечує захист персональних даних без втрати функціональності записів для подальшого аналізу.

Особливістю роботи з анонімізованими темпоральними даними є необхідність збереження контексту для аналізу. Наприклад, часові мітки, характеристики пристрою та параметри запису повинні залишатися доступними, оскільки вони критично важливі для побудови моделей і цифрових двійників. У цьому контексті деперсоналізація є ефективним методом, адже вона видаляє лише ті дані, які безпосередньо ідентифікують пацієнта, зберігаючи технічну та дослідницьку інформацію.

Важливо враховувати можливість деанонімізації, особливо якщо дані поєднуються з іншими наборами. Тому в процесі анонімізації може додатково використовуватися генералізація, коли дані, наприклад про вік, подаються у вигляді діапазонів ("30-40 років"), або шумове маскування, що додає невеликі варіації до числових даних для ускладнення ідентифікації.

Таким чином, анонімізація медичних темпоральних даних дозволяє забезпечити конфіденційність пацієнтів і створює умови для їх безпечного використання у дослідженнях та при побудові цифрових двійників. Це критично важливий процес, який вимагає поєднання автоматизованих технологій і дотримання етичних норм у роботі з медичною інформацією.

2.2. Використання метаданих та часових міток у темпоральних даних

Метадані є важливим елементом темпоральних даних, оскільки вони додають контекст і забезпечують більш глибоку інтерпретацію. У медичних дослідженнях метадані описують ключові аспекти збору даних, зокрема тип та модель обладнання, його налаштування, параметри калібрування, а також зовнішні умови, які могли вплинути на результат. Приклад метаданих пристрою відеоендоскопії наведено нижче у лістингу 2.5.

Лістинг 2.5. Метадані про налаштування пристрою

```
{
  "device_model": "LaryngoScope-X",
  "calibration_date": "2024-01-15",
  "operator": "Dr. Ivanenko",
  "lighting_conditions": "low"
}
```

Ці дані не лише допомагають в аналізі, але й забезпечують відтворюваність експериментів і точність при порівнянні результатів у різних пацієнтів чи дослідженнях.

Часові мітки є невід'ємною частиною метаданих, оскільки вони дозволяють упорядкувати темпоральні дані та синхронізувати їх між різними джерелами. Наприклад, у відео, де аналізується робота голосових зв'язок, кожен кадр отримує часову мітку, яка прив'язує його до конкретного моменту часу, приклад таких даних наведено у лістингу 2.6.

Лістинг 2.6. Темпоральні дані для кадру відео

```
{
  "frame_id": 1,
  "timestamp": "2024-12-10T10:00:00Z",
  "features": {
    "vibration_frequency": 125,
    "amplitude": 0.85,
    "shape_coefficient": 0.78
  }
}
```

Ця структура дозволяє точно зіставляти відеодані з іншими сигналами, наприклад, аудіозаписами або даними з сенсорів.

Синхронізація мультимодальних даних із різних джерел є важливим етапом, особливо для створення цифрових двійників. Для цього використовується універсальний часовий стандарт (UTC), що гарантує узгодженість навіть у складних системах із різними часовими зонами. Наприклад, відеозапис із часовими мітками можна синхронізувати з показниками тиску чи температури, записаними сенсорами. Приклад таких синхронізованих даних наведено у лістингу 2.7.

Лістинг 2.7. Синхронізація мультимодальних даних

```
{
  "timestamp": "2024-12-10T10:00:00Z",
  "video_frame": "frame001.jpg",
  "sensor_data": {
    "pressure": 120,
    "temperature": 36.5
  }
}
```

Для роботи з великими обсягами мультимодальних даних використовуються спеціалізовані формати. Наприклад, DICOM забезпечує зберігання відео разом із вбудованими метаданими, такими як параметри налаштування пристрою. Формат HDF5 дозволяє організовувати великі обсяги мультимодальних даних, таких як відео, аудіо та сенсори, у вигляді ієрархічних структур, що полегшує доступ до конкретних даних. Приклад даних у форматі HDF5 наведено у лістингу 2.8.

Лістинг 2.8. Структура даних у форматі HDF5

```
/Video
  /Frame_001
    VibrationFrequency: 125
    Amplitude: 0.85
  /Frame_002
    VibrationFrequency: 130
    Amplitude: 0.88
  /Sensors
    /Timestamp_001
      Pressure: 120
      Temperature: 36.5
```

Ця структура дозволяє швидко отримати доступ до даних, зберігаючи повну інформацію про їхній контекст. Завдяки правильній організації метаданих і синхронізації інформації створюються умови для точного аналізу динаміки, інтеграції мультимодальних даних і створення високоточних цифрових двійників медико-біологічних об'єктів.

2.3. Основні принципи семантичного аналізу та синхронізації темпоральних мультимодальних даних

Темпоральні мультимодальні дані, що використовуються для побудови цифрових двійників, є складними не лише за своєю структурою, але й за контекстом, у якому вони створюються. Метадані, які супроводжують такі дані, відіграють ключову роль у їхньому аналізі, оскільки містять інформацію про параметри пристроїв, умови запису, характеристики пацієнтів та результати попередньої обробки. Вони дозволяють забезпечити повну картину щодо джерела, якості та релевантності отриманих даних, що є критичним для їх коректної синхронізації та інтеграції в складні інформаційні системи.

Семантичний аналіз метаданих дозволяє виявити приховані залежності між параметрами, врахувати контекст їх створення та забезпечити узгодженість даних із різних джерел. Наприклад, це дозволяє визначити, як параметри калібрування пристрою впливають на точність запису або як умови освітлення змінюють характеристики відеопотоку.

Такий підхід забезпечує не лише організацію даних, але й підвищує якість їхнього аналізу, створюючи основу для ефективного прогнозування та адаптації цифрових двійників до змін у зовнішніх умовах. Таким чином, метадані стають не просто допоміжною інформацією, а критичним інструментом для автоматизації, валідації та моделювання процесів, пов'язаних із побудовою цифрових двійників. Їхнє правильне впровадження та інтеграція значно підвищують надійність і точність кінцевих результатів.

Також він додає додатковий рівень значущості, дозволяючи формалізувати знання про зв'язки між даними, їхній контекст та залежності, які не є очевидними при традиційній обробці. Це відкриває можливості для інтеграції даних у складні моделі, що дозволяють проводити глибокий аналіз, знаходити приховані закономірності та будувати прогнози. Наприклад, семантичний підхід може виявити вплив технічних параметрів пристрою на якість даних або встановити залежності між біологічними характеристиками пацієнтів і результатами діагностики.

Для реалізації семантичного аналізу використовуються сучасні технології, такі як графові моделі даних і бази на їх основі (наприклад, Neo4j). Ці інструменти забезпечують гнучкий підхід до зберігання та обробки інформації у вигляді вузлів (об'єктів) та зв'язків (відношень), що дозволяє моделювати складні взаємозв'язки між даними. Завдяки цьому графові бази даних особливо ефективні для роботи з темпоральними мультимодальними даними, які мають складну ієрархічну структуру та великий обсяг метаданих.

2.4. Семантичні графи

Семантичний граф для аналізу метаданих відеоендоскопії є структурою, яка представляє дані у вигляді **вузлів** (nodes) та **зв'язків** між ними (edges), що дозволяє описувати контекст, характеристики й залежності між елементами даних. Такий підхід спрощує організацію, обробку та аналіз складних мультимодальних даних.

У семантичному графі **вузли** представляють ключові об'єкти, які беруть участь у процесі збору даних і їхнього аналізу:

- **пацієнт** (Patient): вузол, який містить дані про пацієнта, наприклад, ім'я, ідентифікатор, діагноз або історію медичних записів;
- **пристрій** (Device): вузол, який описує технічні характеристики пристрою, зокрема модель, дату калібрування, тип сенсорів та інші параметри;
- **кадр** (Frame): вузол, який представляє окремий кадр відеозапису та пов'язані з ним параметри, наприклад, частоту вібрації, амплітуду або коефіцієнт форми;
- **умови запису** (Recording Conditions): вузол, що включає інформацію про умови, за яких було зібрано дані, такі як освітлення, температура, положення пристрою.

Зв'язки відображають взаємодію між вузлами та дозволяють формалізувати відносини між об'єктами:

- **«записано пристроєм»** (createdBy) – зв'язок між медичним записом і пристроєм, який використовувався для його створення;
- **«отримано за умов»** (hasCondition) – зв'язок, що описує, за яких умов було зроблено запис;
- **«характеризується параметрами»** (hasFeature) – зв'язок, що пов'язує кадр відео із його характеристиками (наприклад, частотою, амплітудою).

Звичайно граф можна представити в JSON-LD (JavaScript Object Notation for Linked Data) – це формат представлення зв’язаної інформації, який легко інтегрується із семантичними вебтехнологіями. Приклад графа у форматі JSON-LD представлено на лістингу 2.9, де ілюструється зв’язок між пацієнтом, пристроєм, записом і параметрами кадру.

Лістинг 2.9 Приклад семантичного графа у форматі JSON-LD (Linked Data)

```
{
  "@context": {
    "@vocab": "http://example.org/ontology#",
    "device": "http://example.org/device#",
    "patient": "http://example.org/patient#"
  },
  "@graph": [
    {
      "@id": "patient:12345",
      "@type": "Patient",
      "hasName": "John Doe",
      "hasRecord": "record:001"
    },
    {
      "@id": "record:001",
      "@type": "Record",
      "createdBy": "device:LaryngoScope-X",
      "hasFeature": "frame:001"
    },
    {
      "@id": "frame:001",
      "@type": "Frame",
      "vibrationFrequency": 125,
      "amplitude": 0.85,
      "shapeCoefficient": 0.78
    },
    {
      "@id": "device:LaryngoScope-X",
      "@type": "Device",
      "calibrationDate": "2024-01-15"
    }
  ]
}
```

Для організації та зберігання даних у вигляді вузлів і зв’язків використовується структура графа, яка дозволяє моделювати складні взаємозв’язки між різноманітними елементами системи. Графова база даних є ефективним рішенням для представлення і взаємодії даних, де зв’язки є такими ж важливими, як і самі дані. Цей підхід забезпечує гнучкість у

зберіганні, пошуку та аналізі даних, що є особливо важливим у медичних системах, які вимагають високої точності та інтеграції даних.

Більш детально розберемо структуру графа, яка використовується для організації та зберігання даних у вигляді вузлів і зв'язків. Такий підхід забезпечує чітке уявлення про відносини між ключовими елементами системи, включаючи пацієнтів, медичні записи, пристрої та дані кадрів:

- пацієнт: вузол із ідентифікатором `patient:12345` описує пацієнта на ім'я "John Doe"; він пов'язаний із записом за допомогою зв'язку
- медичний запис: Вузол із ідентифікатором `record:001` представляє запис, створений пристроєм `device:LaryngoScope-X`; він також пов'язаний із кадром відео через зв'язок `hasFeature`;
- кадр: вузол із ідентифікатором `frame:001` описує окремий кадр відеозапису з параметрами: частотою вібрації 125 Гц, амплітудою 0.85 і коефіцієнтом форми 0.78;
- пристрій: вузол із ідентифікатором `device:LaryngoScope-X` описує пристрій, включаючи дату його останнього калібрування – 15 січня 2024 року.

Для наочності візуалізуємо семантичний граф на рис. 2.1, який демонструє взаємозв'язки між ключовими елементами системи, такими як пацієнт, пристрій, запис і параметри кадру. Цей граф дозволяє чітко відобразити структуру даних, їхні залежності та контекст створення, що є критично важливим для забезпечення коректного аналізу та моделювання. Візуалізація дає змогу легко зрозуміти, як різні елементи взаємодіють між собою, наприклад, як параметри калібрування пристрою впливають на результати запису або як певні характеристики пацієнта корелюють із параметрами відеоданих.

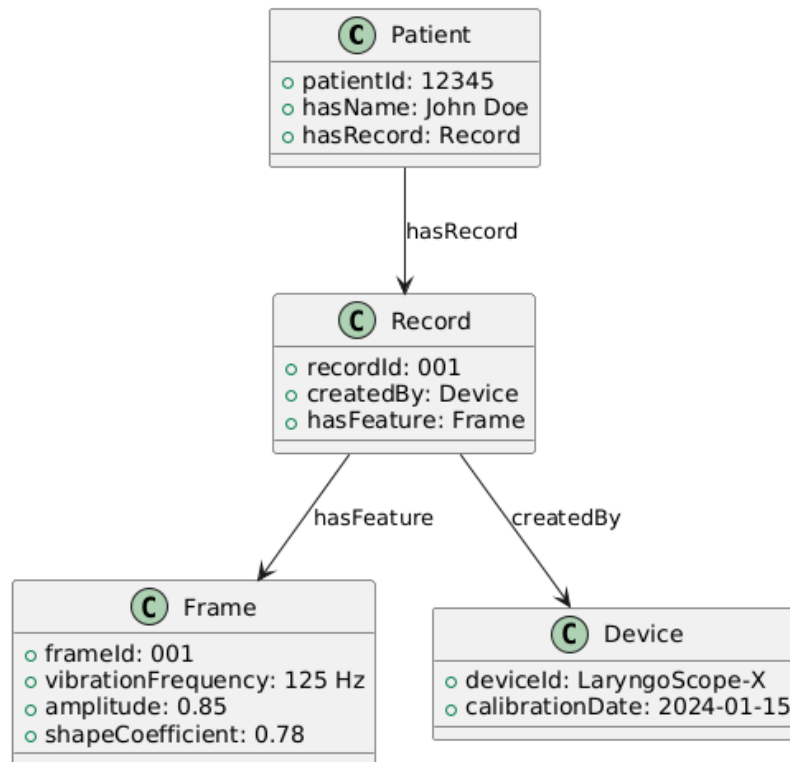


Рис. 2.1. Візуалізація семантичного графу

Граф також дозволяє виконувати гнучкий семантичний пошук, виявляючи приховані закономірності, наприклад, залежність між частотою коливань голосових зв'язок і амплітудою в пацієнтів із певними патологіями. Завдяки цьому дослідники можуть не лише аналізувати окремі вузли та параметри, але й формувати складні запити, які допомагають отримати нову інформацію, що важлива для діагностики та прогнозування. Крім того, така структура графа дозволяє інтегрувати нові дані без необхідності значної перебудови моделі, що робить її масштабованою і зручною для використання в системах цифрових двійників у різних медичних контекстах.

2.5. Метод синхронізації темпоральних мультимодальних даних

Синхронізація аудіо- та відеоданих є одним із ключових завдань при роботі з темпоральними мультимодальними даними, особливо в контексті створення цифрових двійників медико-біологічних об'єктів. Така синхронізація дозволяє поєднувати різноманітні сигнали у спільній часовій шкалі, забезпечуючи їхню узгодженість і коректність для подальшого аналізу. Це важливо не лише для точного відображення динаміки об'єкта дослідження, але й для інтеграції цих даних у семантичні моделі, такі як графові бази даних.

Темпоральні мультимодальні дані, що включають відеопотік, аудіосигнали та метадані про умови запису, часто мають різні частоти дискретизації та часові мітки. Це створює складнощі у їхньому узгодженні. Наприклад, відеодані з високою частотою кадрів можуть не збігатися за часовою шкалою з аудіосигналами, що записані на іншій частоті. Крім того, затримки, викликані апаратними обмеженнями або умовами запису, можуть призводити до значних помилок у подальшому аналізі.

У цьому підрозділі пропонується метод синхронізації темпоральних мультимодальних даних, що базується на використанні часових міток, крос-кореляційного аналізу та нормалізації частот. Метод передбачає узгодження сигналів різної природи та їх подальше збереження у графовій структурі, яка дозволяє відобразити всі зв'язки між елементами даних, такими як пацієнт, пристрій, відеокадри та аудіосегменти. Графова модель забезпечує не лише інтеграцію даних, але й можливість їхнього подальшого аналізу, виявлення прихованих закономірностей і швидкого пошуку релевантної інформації.

Для забезпечення точності синхронізації аудіо- та відеоданих необхідно пройти кілька *основних етапів* обробки, які враховують специфіку темпоральних мультимодальних сигналів. Основна мета методу – створити єдину часову шкалу для всіх типів даних, що дає змогу

проводити комплексний аналіз і візуалізацію змін у гортані під час голосоутворення.

Першим кроком буде перетворення кожного розглянутого кадру відеозапису у сірий колір. Це необхідно для зниження обчислювальної складності, оскільки зображення в градаціях сірого містять лише один канал інтенсивності пікселів, на відміну від трьох (RGB) у кольорових зображеннях. Перетворення значно полегшує подальший аналіз, дозволяючи алгоритмам зосередитися на зміні яскравості, яка має важливе діагностичне значення, а також надає такі переваги, як зменшення розміру даних (менша кількість інформації для обробки) та підготовка до подальшого аналізу (покращення якості трекінгу ключових точок).

Для *реалізації цього етапу* використовується метод `cv2.cvtColor` із бібліотеки OpenCV, що забезпечує швидке та точне перетворення. Наведений нижче лістинг 2.10. демонструє приклад перетворення кольорового зображення у сірий формат.

Лістинг 2.10. Перетворення кадрів у сірий колір

```
import cv2
# Завантаження відеофайлу
video_capture = cv2.VideoCapture('larynx_video.mp4')
while True:
    ret, frame = video_capture.read()
    if not ret:
        break
    # Перетворення кадру в градації сірого
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Подальше оброблення кадру
    cv2.imshow('Gray Frame', gray_frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break;
```

На *другому етапі* виконується масштабування інтенсивності пікселів. Це дозволяє нормалізувати яскравість і контрастність між кадрами, що особливо важливо при змінних умовах освітлення під час запису. Інтенсивність пікселів масштабується до заданого діапазону,

наприклад, $[0, 1]$, що забезпечує узгодженість у подальшому аналізі та підготовку до синхронізації з аудіосигналом. У лістингу 2.11 наведено приклад нормалізації інтенсивності кадрів.

Лістинг 2.11. Перетворення кадрів у сірий колір

```
# Нормалізація інтенсивності пікселів  
normalized_frame = gray_frame / 255.0
```

Третім кроком є виділення характеристичних ознак із кадрів. Для цього використовуються алгоритми комп'ютерного зору, які дозволяють знайти ключові точки або ознаки, що найкраще описують рух об'єкта (наприклад, голосових зв'язок). Ці ключові точки є важливими, оскільки вони фіксують зміни у структурі об'єкта в часі, що дозволяє аналізувати динаміку процесів. Виділені ознаки можуть включати положення, текстуру або локальні зміни інтенсивності пікселів у кадрі. Такі точки є основою для аналізу кореляції між послідовними кадрами та їх синхронізації з аудіоданими.

Для реалізації цього етапу використовується метод із бібліотеки OpenCV, який дозволяє ідентифікувати найбільш значущі точки на зображенні, орієнтуючись на градієнтні зміни. Лістинг 3 демонструє реалізацію цього алгоритму для виявлення ключових точок. Для кращого розуміння результатів, знайдені точки візуалізуються безпосередньо на зображенні, що дозволяє оцінити їхню точність і розташування. Візуалізацію знайдених точок можна побачити на Рис. 2.2, де на кадрі відео показано розташування характеристичних ознак, отриманих після обробки. Це допомагає ідентифікувати найбільш значущі області руху голосових зв'язок та їхню взаємодію з іншими елементами, приклад коду для виділення ключових точок представлено на лістингу 2.12.

Лістинг 2.12. Виділення ключових точок із кадру

```
# Виділення характеристикних точок
corners = cv2.goodFeaturesToTrack(normalized_frame, maxCorners=50,
qualityLevel=0.01, minDistance=10)
for corner in corners:
    x, y = corner.ravel()
    cv2.circle(gray_frame, (int(x), int(y)), 5, 255, -1)

cv2.imshow('Features', gray_frame)
```

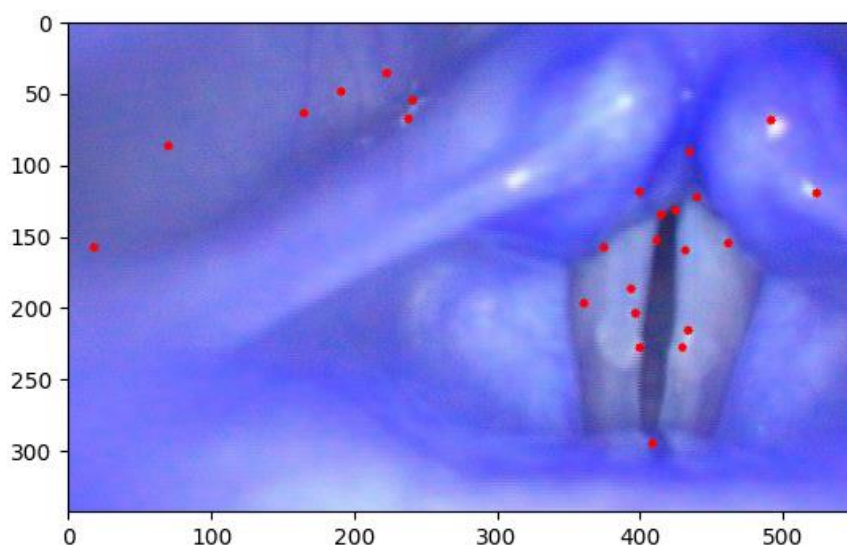


Рис. 2.2. Візуалізація знайдених точок для трекінгу.

На четвертому етапі виконується аналіз кореляції між відео- та аудіосигналами, який є критичним для забезпечення їхньої синхронізації. Цей етап дозволяє вирівнювати дані різної природи (наприклад, зміщення ключових точок у відео та амплітуду аудіосигналу) на спільній часовій шкалі. Суть методу полягає у використанні алгоритмів крос-кореляції для виявлення часових зсувів між сигналами, які виникають через різні частоти запису або затримки у пристроях.

У відеопотоці нормалізованим представленням є зміщення характеристикних точок відносно першого кадру. Цей підхід дозволяє звести весь відеосигнал до компактного набору числових значень, які

відображають зміни об'єкта у часі. Наприклад, динамічне переміщення голосових зв'язок може бути описано координатами їхніх ключових точок у кожному кадрі.

Я

к

щ

$$\Delta_{xy} = x_i - x_0, y_i - y_0 \quad (2.2)$$

о Загальне зміщення для кожного кадру обчислюємо як евклідову відстань між координатами (2.3). i -му кадрі, а x_0 і y_0 — координати тієї ж точки у першому кадрі, то зміщення точки можна обчислити за формулою (2.3)

$$d_i = \sqrt{(\Delta x_i)^2 + (\Delta y_i)^2} \quad (2.3)$$

Нормалізоване значення зміщення для відео отримуємо шляхом масштабування, за формулою (2.4).

$$d_i^{norm} = \frac{d_i}{\max(d)} \quad (2.4)$$

Для аудіосигналу нормалізація проводиться за амплітудою. Якщо a_i — амплітуда сигналу у момент i , то нормалізоване значення амплітуди визначається за формулою (2.5).

$$a_i^{norm} = \frac{(a_i - \min(a))}{(\max(a) - \min(a))} \quad (2.5)$$

Це дозволяє привести дані до діапазону $[0, 1]$, таким чином, відео представляється у вигляді послідовності нормалізованих векторів зміщення, що зменшує обсяг обчислень і спрощує аналіз.

В аудіосигналі нормалізованим представленням є амплітудно-частотні характеристики звукової хвилі, такі як спектральна енергія або частота коливань. Ці параметри відображають зміни у звуковому сигналі, що корелюють із рухом голосових зв'язок, зафіксованим у відео. Нормалізація амплітуди аудіосигналу також виконується для приведення даних до однакового масштабу.

Для виконання аналізу крос-кореляції ці нормалізовані представлення відео- та аудіосигналів використовуються як вхідні дані. Алгоритм

порівнює сигнали у різних часових положеннях, обчислюючи коефіцієнт кореляції для кожного можливого зсуву. Максимальний коефіцієнт кореляції вказує на оптимальний часовий зсув між сигналами, який необхідно компенсувати. Код у лістингу 2.13. демонструє розрахунок крос-кореляції між нормалізованими відео- та аудіоданими.

Лістинг 2.13. Аналіз крос-кореляції між відео- та аудіосигналами

```
import numpy as np

from scipy.signal import correlate

# Приклад нормалізованих даних
video_displacement = np.array([0.1, 0.15, 0.2, 0.25, 0.3]) # Зміщення
ключових точок
audio_amplitude = np.array([0.2, 0.25, 0.28, 0.3, 0.32]) # Амплітуда
аудіосигналу

# Виконання крос-кореляції
correlation = correlate(video_displacement, audio_amplitude, mode='full')
time_shift = np.argmax(correlation) - (len(audio_amplitude) - 1)
print(f"Часовий зсув між сигналами: {time_shift} кадрів/секунд")
```

Виявлений зсув дозволяє скоригувати часові шкали обох сигналів, забезпечуючи їхню синхронізацію. Такий підхід гарантує, що відео- та аудіодані відображають одну й ту ж подію в одному часовому інтервалі. Візуалізація результатів синхронізації може бути використана для перевірки коректності вирівнювання сигналів і подальшого аналізу.

Завершальним етапом є збереження синхронізованих даних у графовій базі, що забезпечує структуроване представлення інформації та зручність для подальшого аналізу. У графовій моделі вузли представляють ключові об'єкти, такі як кадри відео, сегменти аудіо, пацієнт або пристрій, а зв'язки між вузлами фіксують взаємозалежності, наприклад "синхронізовано з" або "записано за допомогою".

Таке збереження не тільки дозволяє інтегрувати дані різної природи, але й забезпечує їхню семантичну пов'язаність, що є критичним для складних систем, таких як цифрові двійники. Використання формату JSON-

LD (Linked Data) у цьому контексті надає можливість зберігати не лише базову інформацію, а й метадані, такі як часові мітки, параметри калібрування пристроїв та інші атрибути, що спрощує інтеграцію даних із зовнішніми системами або їх подальший аналіз.

У лістингу 2.14. наведено приклад створення графової структури у форматі JSON-LD, де синхронізовані відеокадри пов'язані із відповідними аудіосегментами, а також міститься додаткова інформація про умови запису. Така структура є гнучкою, легко розширюється для підтримки нових типів даних і забезпечує високий рівень доступності для різних аналітичних інструментів.

Лістинг 2.14. Створення графа у форматі JSON-LD

```
import json
# Створення графової структури у форматі JSON-LD
graph = {
    "@context": {"@vocab": "http://example.org/ontology#"},
    "@graph": [
        {"@id": "frame:001", "@type": "Frame", "timestamp": "2024-02-12T10:00:00Z", "linkedAudio": "audio:001"},
        {"@id": "audio:001", "@type": "AudioSegment", "timestamp": "2024-02-12T10:00:00Z", "frequency": 440, "amplitude": 0.8}
    ]
}

# Збереження у JSON
with open('synchronized_data.json', 'w') as file:
    json.dump(graph, file, indent=4)
```

Такий покроковий підхід гарантує точну синхронізацію мультимодальних даних, їх інтеграцію в єдину часову шкалу та підготовку до подальшого аналізу. Завдяки використанню нормалізованих представлень відео- та аудіоданих, алгоритм забезпечує зменшення обчислювальної складності та підвищує точність вирівнювання сигналів.

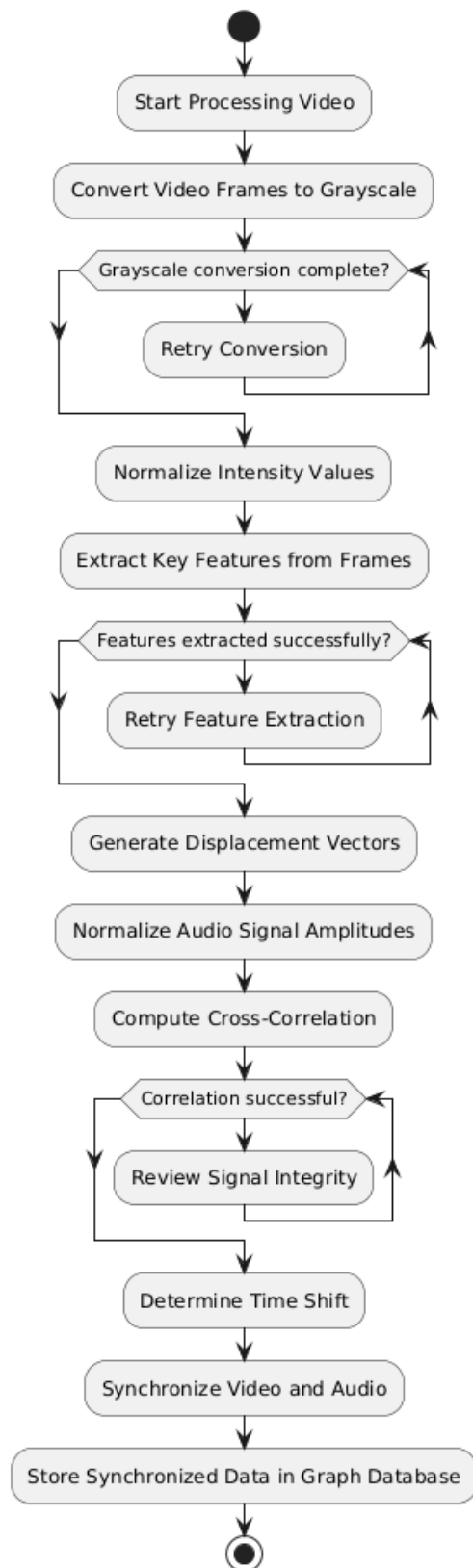


Рис. 2.3. Діаграма активності методу синхронізації темпоральних мультимодальних даних

Усе вищезазначене зображено за допомогою діаграми активності (рис. 2.3), яка відображає основні етапи реалізації пропонованого *методу синхронізації темпоральних мультимодальних даних*.

Отже, розроблений метод синхронізації темпоральних мультимодальних даних забезпечує точне вирівнювання аудіо- та відеосигналів, враховуючи їхню різну природу та частоти дискретизації. Покроковий підхід, який включає попередню обробку, нормалізацію, виділення характеристичних ознак і крос-кореляційний аналіз, дозволяє інтегрувати дані у спільну часову шкалу, готуючи їх до подальшого аналізу. Завдяки використанню нормалізованих представлень, таких як зміщення ключових точок у відео та амплітудні характеристики аудіо, забезпечується висока точність синхронізації. Збереження синхронізованих даних у графовій базі з використанням формату JSON-LD сприяє структурованій інтеграції даних, спрощуючи їхній аналіз і відкриваючи можливості для побудови цифрових двійників.

Попри переваги, метод має певні обмеження. Його ефективність залежить від якості вихідних даних: шум у відео- або аудіосигналах може вплинути на точність розрахунків, а неправильно обрані параметри нормалізації здатні спричинити похибки у вирівнюванні сигналів. Обчислювальна складність методу, особливо на етапах обробки відеопотоку та крос-кореляції, може стати значним бар'єром при роботі з великими обсягами даних. Відсутність автоматизованого вибору параметрів також обмежує масштабованість і універсальність методу.

Для вдосконалення методу доцільно впровадити алгоритми шумозахисту, оптимізувати обчислювальні процеси, зокрема через використання багатопотокової обробки або GPU-акселерації, а також автоматизувати вибір параметрів за допомогою машинного навчання.

Загалом, запропонований метод забезпечує ефективну інтеграцію мультимодальних даних для роботи у складних медичних системах і

побудови цифрових двійників. Його подальше вдосконалення дозволить значно розширити функціональні можливості та підвищити надійність застосування в різних сценаріях аналізу даних.

2.6. Метод семантичного аналізу для виявлення кореляцій між наборами даних та прогнозування поведінки програмно-апаратних компонентів цифрового двійника

Семантичний аналіз метаданих може бути використаний у таких аспектах.

1. Виявлення залежностей – наприклад, як частота вібрацій голосових зв'язок залежить від параметрів освітлення чи калібрування пристрою. Це дозволяє визначати, які саме фактори впливають на характеристики сигналу, та виявляти приховані взаємозв'язки, що не є очевидними при традиційному аналізі.
2. Ідентифікація патернів – автоматичне виявлення патернів у пацієнтів із однаковими діагнозами чи умовами збору даних. Ця інформація може бути використана для створення класифікаційних моделей, які дозволяють ідентифікувати типові випадки або передбачати ймовірність виникнення певних патологій.
3. Аналіз якості даних – тобто визначення, які метадані найбільше впливають на точність отриманих даних. Це допомагає оптимізувати процес збору та попередньої обробки даних, усуваючи ті аспекти, що можуть вносити похибки або знижувати надійність висновків.
4. Побудова прогнозів – на основі зв'язків у графі можна будувати прогнози про можливі діагностичні висновки. Використання зв'язності графа дозволяє виявляти логічні та статистичні закономірності, які сприяють моделюванню поведінки системи за змінних умов.

Він передбачає організацію, зберігання та обробку даних у спосіб, який дозволяє виявляти взаємозв'язки між ними, забезпечувати інтеграцію

з іншими системами та проводити складний аналіз. Цей підхід ґрунтується на використанні семантичних технологій, таких як графові бази даних, RDF/OWL-онтології та SPARQL-запити. Наприклад, онтології можуть описувати терміни та їхні зв'язки в контексті медичних даних, що дозволяє стандартизувати метадані та забезпечити їх сумісність із зовнішніми системами. Графові бази даних, такі як Neo4j, дозволяють зберігати зв'язки між об'єктами (наприклад, пацієнт, пристрій, умови запису) у формі вузлів і зв'язків, що спрощує виявлення складних залежностей. SPARQL-запити використовуються для отримання інформації з графа, наприклад, для пошуку пацієнтів із подібними діагнозами або для аналізу впливу певних умов запису на результати досліджень. Завдяки цьому семантичний аналіз стає потужним інструментом для роботи з великими обсягами медико-біологічних даних, забезпечуючи їхнє інтегроване представлення, автоматизацію аналізу та створення прогнозів.

Тепер перейдемо до опису пропонованого методу, який розпочинається зі створення онтології метаданих. Онтологія виступає основою для формалізації знань і забезпечує чітке визначення типів об'єктів, їх атрибутів та зв'язків між ними. У контексті цифрових двійників ларинкса, онтологія дозволяє систематизувати та структурувати інформацію про пацієнтів, пристрої, відеокадри, їхні характеристики та умови запису.

Онтологія може включати різноманітні елементи. Наприклад, типи об'єктів, такі як пацієнт, пристрій, кадр відео, характеристика кадру, створюють основу для моделювання складних систем. Зв'язки, такі як "записано пристроєм", "характеризується параметром" або "належить пацієнту", описують взаємозалежності між елементами системи. Атрибути, наприклад, дата, час, частота коливань або амплітуда, додають контекст, необхідний для подальшого аналізу даних.

Для збереження цих даних пропонується використати графову базу даних **Neo4j**, яка забезпечує зберігання у вигляді вузлів і зв'язків. Така

модель значно полегшує виконання запитів, виявлення залежностей і аналіз складних зв'язків. Neo4j підтримує мову запитів **Cypher**, яка дозволяє легко створювати, запитувати та аналізувати графові структури.

Наприклад, у лістингу 2.15 наведено приклад створення вузлів і зв'язків у Neo4j для пацієнта, пристрою, запису та характеристик кадру. Зв'язки між вузлами відображають, як дані були отримані, ким і за яких умов, дозволяючи створити цілісну модель системи.

Лістинг 2.15. Завантаження метаданих у Neo4j

```
CREATE (patient:Patient {id: "12345", name: "John Doe"})
CREATE (device:Device {model: "LaryngoScope-X", calibrationDate: "2024-01-15", operator: "Dr. Ivanenko"})
CREATE (record:Record {id: "001"})
CREATE (feature:Feature {id: "001", vibrationFrequency: 125, amplitude: 0.85, timestamp: "2024-12-10T10:00:00Z"})
CREATE (patient)-[:HAS_RECORD]->(record)
CREATE (record)-[:CREATED_BY]->(device)
CREATE (record)-[:HAS_FEATURE]->(feature)
```

Графова структура дозволяє легко здійснювати пошук і аналіз даних через **SPARQL** або **Cypher**-запити. Це забезпечує доступ до глибоких аналітичних висновків і дає змогу виявляти приховані закономірності в метаданих. Наприклад, SPARQL-запит (лістинг 2.16) дозволяє отримати всі записи, створені певним пристроєм, разом із відповідними часовими мітками та частотами коливань.

Лістинг 2.16. SPARQL-запит для отримання всіх записів, створених певним пристроєм

```
SELECT ?record ?timestamp ?frequency
WHERE {
  ?record rdf:type :Record .
  ?record :createdBy ?device .
  ?device :model "LaryngoScope-X" .
  ?record :hasFeature ?feature .
  ?feature :vibrationFrequency ?frequency .
  ?feature :timestamp ?timestamp .
}
```


Методологія, яка базується на онтологіях та графових базах даних, дозволяє ефективно впорядковувати, інтегрувати та аналізувати метадані. Завдяки такій структурі можливе автоматизоване виявлення закономірностей, прогнозування та розширений аналіз для підтримки прийняття рішень у складних медичних системах.

Пропонується розширити традиційний семантичний аналіз, інтегрувавши графові бази даних із алгоритмами машинного навчання. Графові знання можуть бути використані як вхідні дані для графових нейронних мереж (Graph Neural Networks, GNN). Цей підхід дозволяє аналізувати складну структуру метаданих, виявляти закономірності та робити прогнози, наприклад, щодо точності запису або можливих аномалій у даних.

Використання GNN для аналізу метаданих цифрових двійників є інноваційним підходом, оскільки дозволяє:

- враховувати складну структуру залежностей між різними типами даних (пацієнти, пристрої, параметри).
- поєднувати просторові та семантичні аспекти даних, моделюючи взаємозв'язки на графовому рівні.
- інтегрувати мультимодальні дані в єдину структуру, що підвищує точність прогнозів і відкриває нові можливості для створення цифрових двійників.

Розглянемо більш детально принципи роботи графових нейронних мереж (Graph Neural Networks, GNN), які ґрунтуються на ітеративному оновленні представлення вузлів у графі. Графи складаються з вузлів (вершин) і зв'язків між ними (ребер), і кожен вузол містить певну інформацію, яка може змінюватися під час ітерацій.

На кожному ітераційному кроці t GNN оновлює представлення вузла v , враховуючи інформацію, отриману від його сусідніх вузлів. Цей процес відбувається за допомогою спеціальної функції агрегації, яка зводить

інформацію від сусідів до компактного представлення, і функції оновлення, яка об'єднує це представлення із поточним станом вузла, більш формально це можна описати формулою (2.6).

$$h_v^{t+1} = AGGREGATE(\{h_u^t: u \in Neighbors(v)\}) + h_v^t \quad (2.6)$$

де

h_v^t — поточне представлення вузла v на ітерації t ,

$Neighbors(v)$ — множина сусідів вузла v ,

$AGGREGATE$ — функція, що збирає інформацію від сусідніх вузлів.

Цей процес оновлення дозволяє вузлам поступово інтегрувати інформацію з локального контексту, який розширюється з кожною ітерацією. Наприклад, після однієї ітерації вузол містить інформацію про своїх безпосередніх сусідів, після двох — про сусідів другого рівня тощо. Кількість ітерацій T визначає глибину "взаємодії" вузла з іншими частинами графа.

Після завершення T ітерацій, остаточне представлення h_v^T використовується для виконання завдань прогнозування. Залежно від задачі, це представлення може застосовуватися для прогнозування характеристик вузлів (наприклад, класифікація або регресія), зв'язків (передбачення нових зв'язків або атрибутів зв'язків) чи графів у цілому (наприклад, класифікація графів).

Перейдемо до **реалізації та навчання** графової нейронної мережі (GNN), яка є потужним інструментом для роботи зі структурованими даними у вигляді графів. Цей процес включає побудову графа, визначення початкових представлень вузлів, вибір архітектури моделі та її подальше навчання. Завдяки здатності GNN інтегрувати інформацію з локального та глобального контексту графа, вона дозволяє ефективно вирішувати задачі, пов'язані з аналізом зв'язків, прогнозуванням характеристик вузлів і створенням узагальнених представлень для складних систем.

Почнемо з *етанів реалізації*, *першим* буде побудова графа, який складається з вузлів (Nodes) і зв'язків (Edges). Вузли представляють об'єкти, такі як пацієнти, пристрої або записи, а зв'язки визначають взаємозв'язки між ними, наприклад, "створено пристроєм" або "належить пацієнту". Кожен вузол має атрибути, які визначають його унікальні особливості. Наприклад, вузол "Пацієнт" може включати інформацію про вік і стать, "Пристрій" — модель і дату калібрування, а "Ознака" — частоту й амплітуду. Математично граф описується за формулою (2.7).

$$G = (V, E) \quad (2.7)$$

де

$V = \{v_1, v_2, \dots, v_n\}$ — множина вузлів,

$E = \{(v_i, v_j)\}$ — множина зв'язків між вузлами.

На *другому етапі* виконується ініціалізація представлень вузлів, яка є важливим кроком для підготовки даних до обробки графовою нейронною мережею. Кожному вузлу $v \in V$ присвоюється початковий вектор ознак h_v^0 , який відображає його унікальні характеристики та атрибути. Цей вектор є багатовимірним і може містити як числові, так і категоріальні дані, що описують властивості вузла. Початкові представлення є базовими даними, які будуть поступово збагачуватися інформацією від сусідніх вузлів протягом ітерацій навчання.

Наприклад, для вузла типу "Пацієнт" початкове представлення може виглядати як $h_{Patient} = [\text{вік}, \text{стать}]$, де "вік" є числовим параметром, а "стать" може бути закодована у вигляді вектора (наприклад, $[1, 0]$ для чоловіка і $[0, 1]$ для жінки).

Для вузла типу "Пристрій" вектор $h_{Device} = [\text{модель}, \text{дата калібрування}]$ може включати ідентифікатор моделі пристрою, закодований як вектор, і числове значення дати калібрування у форматі, придатному для моделі (наприклад, кількість днів від певної

базової дати). Для вузла типу "Ознака" початковий вектор $h_{Feature} = [\text{частота, амплітуда}]$ може складатися з числових параметрів, які описують сигнал або об'єкт дослідження.

Ці вектори ознак створюють основу для навчання моделі, адже вони відображають властивості вузлів, які безпосередньо впливають на результати прогнозування. Під час навчання GNN ці початкові вектори будуть оновлюватися через процес агрегації, поступово включаючи інформацію від сусідніх вузлів у графі. Таким чином, початкові представлення є стартовими точками, від яких модель розпочинає своє навчання.

Процес ініціалізації представлень також залежить від задачі. У задачах класифікації вузлів важливо, щоб вектори відображали найбільш релевантні атрибути, що допоможуть диференціювати вузли між собою. У задачах прогнозування зв'язків важливо врахувати атрибути, які найбільше впливають на наявність або відсутність зв'язку між вузлами.

На третьому етапі необхідно обрати архітектуру GNN, яка найкраще відповідає поставленій задачі та специфіці даних. Наприклад, для задачі класифікації вузлів може бути використана GCN (Graph Convolutional Network), яка обчислює середнє представлення сусідів, інтегруючи локальну інформацію графа. Це робить її ефективною для аналізу вузлів у графах зі стабільною та рівномірною структурою. Якщо важливо враховувати вагомість зв'язків між вузлами, можна обрати GAT (Graph Attention Network), яка використовує механізм уваги для динамічного визначення важливості сусідів, забезпечуючи адаптивний підхід до агрегації інформації. Для роботи з великими графами, де повний аналіз всіх сусідів кожного вузла може бути обчислювально затратним, підходить GraphSAGE, яка ефективно агрегує інформацію з підвибірок сусідів, зберігаючи при цьому ключові характеристики графа. Цей етап є ключовим для адаптації моделі до конкретних умов задачі та графової структури.

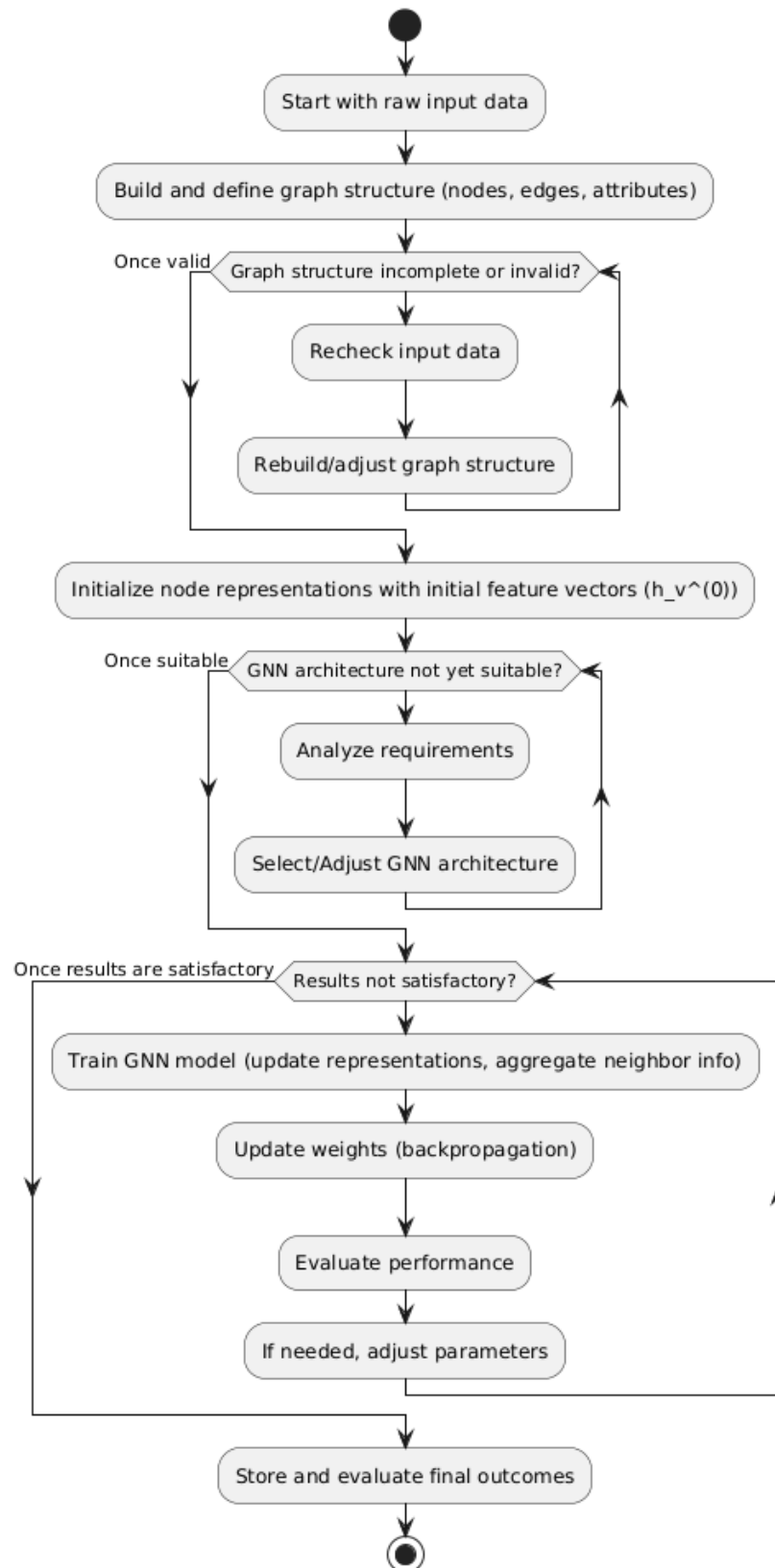


Рис. 2.3. Діаграма активності методу синхронізації темпоральних мультимодальних даних

Тепер зобразимо етапи реалізації графової нейронної мережі на *діаграмі активності* (рис 2.4). Вона чітко відображає послідовність кроків, включаючи побудову графа, ініціалізацію представлень вузлів, вибір архітектури GNN, навчання моделі та прогнозування результатів. Такий візуальний підхід дозволяє краще зрозуміти основні етапи процесу та їхню взаємозалежність.

Після побудови графа та ініціалізації представлень вузлів модель переходить до етапу **навчання**, який є центральною частиною процесу роботи GNN. У процесі навчання модель ітеративно оновлює представлення вузлів, використовуючи інформацію від їхніх сусідів. На кожному кроці t вузол v отримує локальну інформацію від своїх сусідів, яка агрегується та інтегрується в його поточне представлення. Цей механізм забезпечує поступове збагачення представлень вузлів інформацією з дедалі ширшого контексту графа. Після виконання T ітерацій представлення вузла h_v^T містить інформацію не лише про самого вузла, але й про його сусідів кількох рівнів. Це дозволяє моделі враховувати як локальні, так і глобальні взаємозв'язки в графі, що є критично важливим для складних структурованих даних. Кінцеві представлення вузлів використовуються для виконання завдань, таких як класифікація вузлів, кластеризація або прогнозування зв'язків.

Наступним важливим етапом є прогнозування зв'язків або атрибутів графа. На основі кінцевих представлень вузлів h_v^T модель обчислює ймовірність існування зв'язків між парами вузлів або передбачає інші характеристики графа. Прогнозування зв'язків зазвичай здійснюється за допомогою функції сумісності, наведеної у формулі (2.8).

де сама f представлена скалярним добутком, однак для прогнозування також можна використовувати нейронну мережу, яка дозволяє враховувати

нелінійні залежності між представленнями вузлів і більш точно моделювати взаємодії між ними, хоча і за більші витрати часу.

Практичне застосування прогнозування зв'язків у контексті цифрових двійників медико-біологічних об'єктів, таких як ларинкс, може включати:

1. Виявлення аномалій у роботі пристроїв. Наприклад, прогнозування зв'язків між вузлом "Пристрій" і вузлом "Кадр" може дозволити виявити випадки, коли пристрій дає некоректні або спотворені дані через неправильне калібрування.
2. Аналіз взаємозв'язку між пацієнтом і діагностичними параметрами. Наприклад, модель може передбачити, чи певні параметри, такі як частота вібрацій голосових зв'язок, відповідають типовим значенням для пацієнтів із певними діагнозами, використовуючи зв'язки між вузлами "Пацієнт" і "Ознака".
3. Прогнозування можливих ускладнень. Наприклад, прогнозування зв'язків між вузлами, що представляють пацієнта та його історію обстежень, дозволяє оцінити ймовірність розвитку ускладнень на основі кореляції між минулими результатами і поточними даними.

Окрім прогнозування зв'язків, GNN може бути використана для визначення атрибутів графа. Наприклад, у вузлі "Ознака", який містить параметри відеокадру, можна передбачити пропущені значення, такі як амплітуда чи частота, на основі даних про сусідні вузли. Це корисно у випадках, коли дані з пристроїв є неповними через технічні помилки або артефакти. Таким чином, прогнозування зв'язків або атрибутів графа не тільки допомагає у виявленні закономірностей і залежностей у даних, але й сприяє підвищенню точності та повноти аналізу у складних медико-біологічних системах. Ці практичні сценарії демонструють потужність GNN у реальних умовах застосування.

Завершальним етапом є оптимізація моделі. Для цього використовується функція втрат, яка оцінює, наскільки точними є

передбачення моделі порівняно з реальними даними. У випадку прогнозування зв'язків використовується логістична функція втрат (лог-лосс), зображена на формулі (2.9).

$$L = - \left(\frac{1}{|E|} \right) \sum_{v_i, v_j \in E} [y_{ij} \cdot \ln(\hat{y}_{ij}) + (1 - y_{ij}) \cdot \ln(1 - \hat{y}_{ij})] \quad (2.9)$$

де

E — множина пар (v_i, v_j) за якими ми рахуємо втрати (наприклад, пар вершин, між якими може бути або не бути зв'язок);

y_{ij} — істинна мітка (1, якщо зв'язок існує, і 0, якщо не існує);

\hat{y}_{ij} — прогнозована ймовірність існування зв'язку.

Для підвищення точності навчання використовуються додаткові методи. Наприклад, регуляризація допомагає запобігти перенавчанню моделі, зменшуючи вплив складності. Один із підходів — регуляризація ваг, яка додає штрафи до великої складності параметрів, тим самим стабілізуючи модель.

Окрім цього, важливим аспектом є вибір гіперпараметрів, таких як кількість ітерацій T , що визначає глибину нейронної мережі, та розмір векторів представлень вузлів $\dim(h_v)$. Оптимальні значення цих параметрів часто визначаються через перевірку моделі на валідаційній вибірці. Використання таких метрик, як точність, повнота або специфічність, дозволяє оцінити, наскільки добре модель справляється зі своєю задачею. Також важливо враховувати баланс вибірки зв'язків, особливо якщо позитивні та негативні приклади представлені нерівномірно.

Завдяки поєднанню цих методів модель здатна досягти високої точності у передбаченні зв'язків, забезпечуючи її ефективне використання в задачах аналізу даних, таких як побудова цифрових двійників та виявлення складних взаємозв'язків у графах.

Тепер створимо *діаграму потоку даних* (рис 2.4.), яка ілюструє основні етапи обробки даних у графовій нейронній мережі.

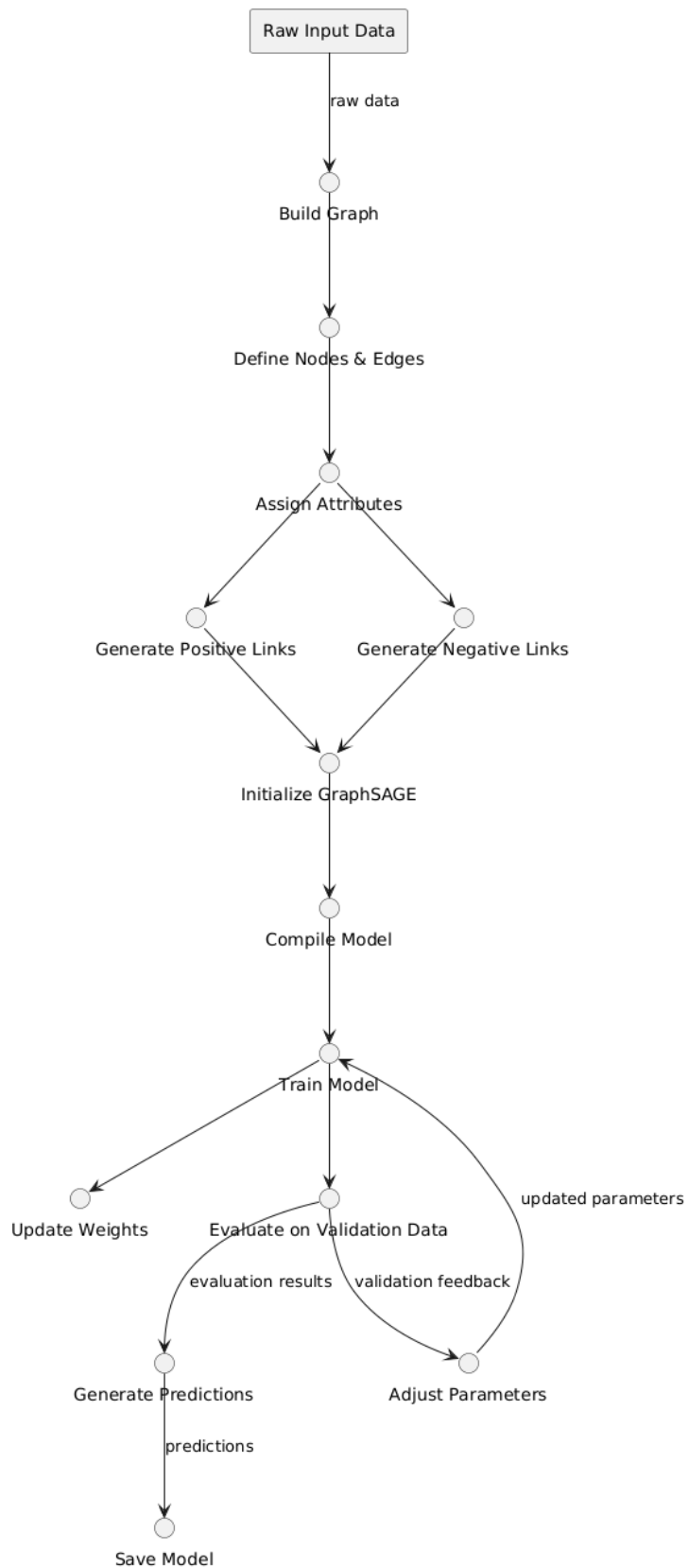


Рис. 2.4. Діаграма потоку даних для навчання графової нейронної мережі (GNN)

Приклад тренування подібної мережі мовою Python представлено у лістингу 2.17. Цей код демонструє всі ключові етапи реалізації GraphSAGE, включаючи створення графа, генерацію прикладів зв'язків, ініціалізацію архітектури моделі, її компіляцію та навчання. Представлений підхід дозволяє ефективно працювати з графовими структурами, поєднуючи вузли та зв'язки в єдину систему для прогнозування. Цей приклад є базою, яку можна адаптувати для вирішення конкретних завдань, таких як аналіз медичних даних або моделювання взаємозв'язків у цифрових двійниках.

Лістинг 2.17. Реалізація в Python

```
import tensorflow as tf
from stellargraph import StellarGraph
from stellargraph.layer import GraphSAGE, link_classification
from stellargraph.mapper import GraphSAGELinkGenerator
from tensorflow.keras import Model

# Створення графа
stellar_graph = StellarGraph.from_networkx(graph, node_features="type")

# Генерація позитивних і негативних прикладів зв'язків
edge_gen = GraphSAGELinkGenerator(stellar_graph, batch_size=32,
num_samples=[10, 5])
train_gen = edge_gen.flow(train_edges, train_labels)

# Архітектура GraphSAGE
graphsage = GraphSAGE(layer_sizes=[16, 16], generator=edge_gen,
dropout=0.5)
x_inp, x_out = graphsage.in_out_tensors()

# Модель прогнозування зв'язків
prediction = link_classification(output_dim=1, output_act="sigmoid",
edge_embedding_method="dot")(x_out)
model = Model(inputs=x_inp, outputs=prediction)
model.compile(optimizer="adam", loss="binary_crossentropy",
metrics=["accuracy"])

# Навчання моделі
history = model.fit(train_gen, epochs=10, validation_data=val_gen,
verbose=1)
```

Таким чином, графові нейронні мережі (*GNN*) є перспективним інструментом для семантичного аналізу метаданих завдяки їхній здатності моделювати складні залежності між об'єктами та параметрами в графових структурах. Вони дозволяють ефективно інтегрувати та аналізувати

мультимодальні дані, поєднуючи інформацію з різних джерел, таких як відео, аудіозаписи, сенсорні дані та текстові метадані. Це забезпечує глибше розуміння зв'язків між елементами системи, що є критично важливим для побудови високоточних цифрових двійників.

2.7. Висновки до розділу 2

У цьому розділі розроблено та детально розглянуто два основних методи, які забезпечують ефективну роботу з темпоральними мультимодальними даними для створення цифрових двійників медико-біологічних об'єктів: метод синхронізації мультимодальних даних та семантичний аналіз метаданих із використанням графових моделей.

Перший метод, синхронізація мультимодальних даних, дозволяє інтегрувати відео- та аудіопотоки в єдину часову шкалу. Цей підхід включає кілька етапів: попередню обробку сигналів, нормалізацію характеристик, виділення ключових ознак та кореляційний аналіз для компенсації часових зсувів між сигналами. Ефективність методу забезпечується завдяки використанню алгоритмів крос-кореляції, що дозволяють точно визначати та коригувати часові невідповідності. Результатом є високоточна синхронізація даних, яка готує їх до подальшого аналізу та моделювання.

Другий метод базується на семантичному аналізі метаданих із застосуванням графових баз даних, таких як Neo4j. Цей підхід дозволяє організовувати дані у вигляді семантичного графа, де вузли представляють об'єкти (пацієнтів, пристрої, відеокадри), а зв'язки між ними описують їх взаємодію та контекст створення. Графові моделі, збережені у форматі JSON-LD, забезпечують можливість ефективного зберігання, виконання складних запитів та автоматизованого аналізу, що дозволяє виявляти приховані закономірності та прогнозувати динаміку об'єктів.

Обидва методи працюють у тісній інтеграції: синхронізація мультимодальних даних створює основу для формування якісних вхідних

даних для графових моделей, тоді як семантичний аналіз дозволяє досліджувати взаємозв'язки між параметрами та прогнозувати їхній вплив на функціонування об'єкта. Ця комбінація підходів значно підвищує точність та функціональність цифрових двійників, роблячи їх цінним інструментом для біомедичних досліджень та персоналізованої медицини.

Водночас методи мають певні обмеження. Наприклад, обчислювальна складність синхронізації може зростати зі збільшенням обсягів даних, а якість семантичного аналізу залежить від повноти та структурованості вхідних метаданих. Подальший розвиток має бути спрямований на оптимізацію обчислень, автоматизацію налаштування параметрів та інтеграцію технологій машинного навчання для підвищення ефективності обох методів.

Таким чином, представлені методи забезпечують не лише інтеграцію та аналіз мультимодальних даних, але й створюють основу для високоточних цифрових двійників, які здатні відображати складні взаємозв'язки між характеристиками об'єкта та їх динамікою у реальному часі.

РОЗДІЛ 3. РОЗРОБЛЕННЯ МЕТОДУ АДАПТАЦІЇ БАЗОВОЇ МОДЕЛІ МЕДИКО-БІОЛОГІЧНОГО ОБ'ЄКТА

Цифрові двійники медико-біологічних об'єктів стають невід'ємним інструментом сучасної медицини, дозволяючи не лише вивчати анатомічні структури, але й моделювати їх функціонування в динаміці. Одним із ключових завдань є створення цифрового аналога реального органу, що максимально відповідає індивідуальним характеристикам пацієнта. Для цього базова модель повинна адаптуватися до конкретних особливостей об'єкта, таких як форма, структура та динаміка рухів, отриманих із мультимодальних даних.

Адаптація базової моделі передбачає використання алгоритмів аналізу відеопотоків та інших даних, отриманих з діагностичного обладнання. Відеопотік, який відображає динаміку роботи органу в реальному часі, є одним із найважливіших джерел інформації. Він дозволяє зафіксувати як макроскопічні характеристики, наприклад, загальну форму органу, так і дрібні зміни, які можуть вказувати на патологічні процеси. Ці дані інтегруються з базовою моделлю для створення точного цифрового двійника.

Особливу роль у цьому процесі відіграють тривимірні згорткові нейронні мережі (3D-CNN), які забезпечують одночасну обробку просторових і часових ознак. На відміну від звичайних алгоритмів, які працюють із статичними зображеннями, 3D-CNN здатні аналізувати відеопотік як об'єм даних, враховуючи зміну характеристик органу у часі. Це дозволяє визначити такі критично важливі ознаки, як періодичність рухів, амплітуду коливань чи зміни текстури, які можуть бути використані для моделювання реального функціонування органу. Наприклад, у випадку гортані алгоритми дозволяють моделювати рух голосових зв'язок, виявляти аномалії чи прогнозувати розвиток патологій.

Процес адаптації базової моделі передбачає кілька етапів. Спочатку виконується попереднє оброблення відеоданих для покращення їхньої якості, зокрема нормалізація яскравості, корекція контрасту та усунення шуму. Потім 3D-CNN виділяє ключові ознаки, що характеризують об'єкт, які використовуються для внесення змін у базову модель. Накладання цих характеристик дозволяє моделі відображати як індивідуальні анатомічні особливості, так і динамічні процеси.

Застосування методів адаптації має низку переваг. По-перше, це дозволяє створювати цифрові двійники, які є максимально точними копіями реальних органів, включаючи їхню динамічну поведінку. По-друге, така інтеграція мультимодальних даних із базовою моделлю забезпечує високу функціональність цифрових двійників, що може бути використано для діагностики, віртуальних експериментів та прогнозування. По-третє, цифрові двійники, створені на основі цього підходу, стають ефективними інструментами для персоналізованої медицини, що враховує особливості кожного пацієнта.

Таким чином, метод адаптації базової моделі відкриває нові перспективи у створенні цифрових двійників, здатних моделювати складні біологічні процеси. Використання алгоритмів глибокого навчання, таких як 3D-CNN, та мультимодальних даних дозволяє значно підвищити точність і адаптивність цифрових моделей, що має важливе значення для подальшого розвитку медицини та біологічних досліджень.

3.1. Теоретичні засади оброблення даних для створення цифрових двійників

Створення цифрових двійників медико-біологічних об'єктів вимагає високої точності у відображенні індивідуальних характеристик пацієнта. Вирішальну роль у цьому відіграє класифікація та оброблення даних, які забезпечують повноцінну адаптацію базової моделі до конкретного органу. Медичні прилади генерують широкий спектр мультимодальної інформації, включаючи відео, зображення, аудіосигнали та параметри сенсорів. Серед цих даних відеопотік є найбільш значущим джерелом, оскільки він дозволяє фіксувати як структурні, так і функціональні особливості органу в динаміці.

Оброблення поточкових даних є надзвичайно складним завданням через їхню мінливість, великий обсяг і потребу у синхронізації. Відеодані, отримані за допомогою діагностичного обладнання, мають високу роздільну здатність, але часто містять шум, змінні умови освітлення та інші артефакти, які необхідно усунути під час попередньої обробки. Ця складність зростає, коли необхідно враховувати часові залежності між кадрами та їх інтеграцію з іншими джерелами інформації, такими як аудіо чи параметри сенсорів.

Особливу увагу приділяють використанню тривимірних згорткових нейронних мереж (3D-CNN), які забезпечують обробку просторово-часових залежностей у поточкових даних. На відміну від класичних методів, які аналізують статичні зображення, 3D-CNN дозволяють враховувати динаміку змін у часі, що є критично важливим для відображення роботи органу в реальних умовах. Наприклад, вони допомагають моделювати періодичність рухів голосових зв'язок чи аналізувати їхню змінену геометрію при патологіях.

Процес адаптації базової моделі до потокових даних передбачає кілька етапів:

1. **Попереднє оброблення даних.** Виконується видалення шуму, нормалізація яскравості та корекція контрасту для підвищення якості відеопотоку. Це включає алгоритми фільтрації, які усувають вплив артефактів і забезпечують стабільність зображення в умовах змінного освітлення.
2. **Аналіз просторово-часових залежностей.** За допомогою 3D-CNN виділяються ключові ознаки, такі як амплітуда та частота рухів, які є важливими для точного моделювання динаміки органу.
3. **Інтеграція виділених ознак у базову модель.** Отримані характеристики накладаються на існуючий шаблон органу, створюючи цифровий двійник, який відображає як анатомічні особливості, так і поведінкові зміни.

Особливістю обробки потокових даних є необхідність синхронізації мультимодальних джерел. Наприклад, відеопотік може бути поєднаний з аудіоданими, щоб аналізувати зв'язок між рухами голосових зв'язок і акустичними характеристиками. Для цього використовуються алгоритми, які вирівнюють часові шкали різних сигналів, усуваючи затримки та забезпечуючи їхню узгодженість.

Складність обробки потокових даних також полягає у високих вимогах до обчислювальних ресурсів, оскільки великі обсяги інформації потребують швидкої обробки в режимі реального часу. Для цього застосовуються оптимізовані методи попередньої обробки, які зменшують обсяг даних без втрати ключових ознак, а також розподілені обчислення для паралельного аналізу великих відеопотоків.

Унікальність підходу до адаптації базової моделі полягає у поєднанні мультимодальних даних, таких як відео та аудіо, з методами глибокого навчання, що дозволяє створювати цифрові двійники з високим ступенем

точності. Такі двійники стають не лише інструментами для вивчення органу, але й інтерактивними моделями для прогнозування його роботи в різних умовах, що має важливе значення для персоналізованої медицини.

3.2. Попереднє оброблення даних

Попереднє оброблення відеоданих є важливим етапом у підготовці інформації для створення цифрових двійників медико-біологічних об'єктів. Якість отриманого відеопотоку значно впливає на точність аналізу та адаптації базової моделі. Відеодані, які надходять з діагностичних приладів, часто містять артефакти, спричинені низькою якістю запису, шумом, неоднорідністю освітлення та іншими факторами, що можуть спотворити результати обробки.

Для забезпечення коректності та узгодженості даних застосовуються методи нормалізації інтенсивності пікселів, видалення шуму і пулінгу. Нормалізація дозволяє усунути відмінності у яскравості й контрасті між кадрами, що особливо важливо для аналізу послідовностей відеопотоку. Видалення шуму спрямоване на покращення якості зображення шляхом усунення цифрових артефактів і стабілізації зображення. Пулінг допомагає зменшити обсяг даних, зберігаючи ключові ознаки, що дозволяє оптимізувати обчислювальні ресурси та зменшити вплив несуттєвих деталей на результати аналізу.

У цьому розділі розглядаються основні підходи до нормалізації інтенсивності пікселів, методи зменшення шуму та роль пулінгу у підготовці відеоданих до подальшого аналізу.

Нормалізація інтенсивності пікселів є одним із ключових етапів попередньої обробки відеоданих. Відеодані, отримані з медичного обладнання, часто характеризуються неоднорідністю інтенсивності пікселів через змінні умови запису. Наприклад, нерівномірне освітлення, відблиски або тінь можуть створювати суттєві перешкоди для точного аналізу. Такі

артефакти можуть негативно впливати на алгоритми, що використовуються для обробки зображень, оскільки вони спричиняють похибки під час виділення ключових ознак та подальшої адаптації базової моделі.

Тому нормалізація є одним із найважливіших етапів попередньої обробки, який забезпечує узгодженість і стабільність відеоданих. Вона дозволяє усунути розбіжності у яскравості та контрасті між кадрами, спричинені різними зовнішніми умовами, і приводить інтенсивність пікселів до стандартизованого діапазону. Це полегшує подальший аналіз відеопотоку, зокрема застосування алгоритмів глибокого навчання, таких як тривимірні згорткові нейронні мережі (3D-CNN).

Процес нормалізації базується на статистичній обробці інтенсивності пікселів кожного кадру. В результаті інтенсивність кожного пікселя приводиться до форми, яка дозволяє зменшити вплив умов запису і підвищити точність виділення ознак. Формально нормалізацію можна описати формулою (3.1).

$$\bar{I}_{ij} = \frac{(I_{ij} - \mu)}{\sigma} \quad (3.1)$$

де

\bar{I}_{ij} — нормалізована інтенсивність пікселя з координатами i, j ;

I_{ij} — оригінальна інтенсивність пікселя;

μ — середнє значення інтенсивності пікселів у кадрі;

σ — стандартне відхилення інтенсивності пікселів.

Середнє значення інтенсивності обчислюється за формулою (3.2).

$$\mu = \left(\frac{1}{N}\right) \sum (I_i) \quad (3.2)$$

де

N — загальна кількість пікселів у кадрі,

I_i — інтенсивність кожного пікселя.

Стандартне відхилення визначається за формулою (3.3).

$$\sigma = \text{sqrt}\left(\left(\frac{1}{N}\right)\Sigma(I_i - \mu)^2\right) \quad (3.3)$$

Нормалізація забезпечує стандартизацію яскравості кожного кадру, що дозволяє зменшити вплив артефактів, таких як неоднорідне освітлення або цифровий шум. Це сприяє коректному аналізу відеопотоку, поліпшує точність алгоритмів обробки та забезпечує узгодженість даних для подальшого використання в адаптації базової моделі. Процедура включає обчислення середнього значення та стандартного відхилення інтенсивності пікселів із подальшим приведенням їх до стандартизованого діапазону. Для демонстрації цього підходу приклад реалізації наведено у лістингу 3.1.

Лістинг 3.1. Нормалізація інтенсивності пікселів

```
import cv2
import numpy as np

# Завантаження кадру з відео
frame = cv2.imread('frame.png', cv2.IMREAD_GRAYSCALE)

# Обчислення середнього значення та стандартного відхилення
mean_intensity = np.mean(frame)
std_intensity = np.std(frame)

# Нормалізація інтенсивності пікселів
normalized_frame = (frame - mean_intensity) / std_intensity

# Масштабування до діапазону [0, 255] для візуалізації
scaled_frame = cv2.normalize(normalized_frame, None, 0, 255,
cv2.NORM_MINMAX)
scaled_frame = scaled_frame.astype(np.uint8)

# Збереження результату
cv2.imwrite('normalized_frame.png', scaled_frame)
```

Видалення шуму є наступним важливим кроком попередньої обробки відеоданих, спрямованим на зменшення артефактів і небажаних варіацій у зображеннях. Шум у відеопотоках є поширеною проблемою, яка може значно ускладнити аналіз медико-біологічних даних. Він може бути

спричинений багатьма факторами, такими як обмеження роздільної здатності або чутливості медичного обладнання, вплив зовнішніх умов, наприклад, змінного освітлення, або особливості методу зйомки, зокрема, швидкість запису відео та тип сенсорів, що використовуються. Ці фактори часто створюють зернистість, розмитість чи цифрові артефакти, які можуть маскувати важливі деталі об'єкта дослідження.

Такі артефакти впливають на здатність автоматичних алгоритмів, зокрема згорткових нейронних мереж (CNN), точно ідентифікувати ключові ознаки, такі як краї, текстурні чи динамічні зміни об'єкта. Це, у свою чергу, може призводити до помилкових висновків або неточних діагностичних рішень. Крім того, шум у відеопотоках може спотворювати часо-просторові зв'язки між кадрами, що є критично важливим для аналізу динамічних характеристик, таких як рух голосових зв'язок чи зміни структури тканин. Для усунення цих проблем використовуються різноманітні методи обробки, які дозволяють зменшити вплив шуму на відеодані, зберігаючи при цьому їх ключову інформативність. Ефективне видалення шуму є необхідним етапом підготовки даних, забезпечуючи надійність і точність подальшого аналізу.

Медіанна фільтрація базується на заміні інтенсивності пікселя середнім (медіанним) значенням пікселів у локальному вікні. Уявімо, що піксель і його сусіди представлені квадратною областю розміром 3×3 , де значення інтенсивності кожного пікселя в околі позначені як P_1, P_2, \dots, P_9 . Медіанна фільтрація передбачає сортування цих значень за зростанням і вибір середнього елемента як нового значення для центрального пікселя:

- формування вікна навколо пікселя;
- сортування значень інтенсивності пікселів;
- вибір медіанного значення як нового значення для центрального пікселя.

Цей підхід дозволяє ефективно усувати артефакти, такі як імпульсний шум, не впливаючи на краї та деталі зображення. Процес медіанної фільтрації можна описати формулою (3.4).

$$I'_{ij} = \text{median}(P_1, P_2, \dots, P_n) \quad (3.4)$$

де

I'_{ij} — інтенсивність центрального пікселя після фільтрації;

(P_1, P_2, \dots, P_n) — інтенсивності пікселів у локальному вікні.

Для реалізації видалення шуму можна використовувати бібліотеку OpenCV. Нижче у лістингу 3.2. наведено приклад використання медіанного фільтра для обробки зображення.

Лістинг 3.2. Реалізація медіанного фільтра

```
import cv2

# Завантаження кадру з відео
frame = cv2.imread('frame.png', cv2.IMREAD_GRAYSCALE)

# Застосування медіанного фільтра
filtered_frame = cv2.medianBlur(frame, 3)

# Збереження результату
cv2.imwrite('filtered_frame.png', filtered_frame)

# Відображення оригінального і фільтрованого кадру
cv2.imshow('Original Frame', frame)
cv2.imshow('Filtered Frame', filtered_frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Якісне видалення шуму забезпечує поліпшення візуальних характеристик відеопотоку, що є вирішальним для подальшого аналізу. У випадку використання згорткових нейронних мереж (CNN), зокрема 3D-CNN, очищення зображень від шуму сприяє більш точному виділенню ознак. Це дозволяє зменшити ймовірність помилкових висновків і забезпечує стабільність роботи моделі.

Максимальний пулінг (Max Pooling) є одним із найпоширеніших методів зменшення розмірності даних у згорткових нейронних мережах. Він

зберігає максимальне значення інтенсивності у кожному локальному вікні, що дозволяє виділяти найбільш значущі ознаки, такі як краї, текстури або локальні контрастні елементи. Цей метод ідеально підходить для обробки відеопотоків, де важливо зберігати ключові зміни, такі як рух голосових зв'язок або динамічні зміни структури тканин.

Під час виконання максимального пулінгу кожне локальне вікно обчислення, зазвичай розміром 2×2 або 3×3 пікселі, зсувається по зображенню з певним кроком (stride). Для кожного вікна обирається найбільше значення, яке записується у вихідний масив. Загальну схему такої операції можна переглянути на рис 3.1. Обчислення максимального пулінгу наведено у формулі (3.5).

$$P_{ij} = \max(I_{i1}, I_{i2}, \dots, I_{in}) \quad (3.5)$$

де

P_{ij} — результат пулінгу для області (i, j) ,

$I_{i1}, I_{i2}, \dots, I_{in}$ — значення інтенсивності пікселів у локальному вікні.



Рис. 3.1. Усереднення значень інтенсивності за допомогою пулінгу

Максимальний пулінг використовується для зменшення розмірності даних, що дозволяє:

- виділяти найбільш значущі ознаки: зберігаються тільки найбільш яскраві деталі, які є найбільш інформативними для аналізу;

- зменшувати вплив шуму: оскільки слабкі сигнали ігноруються, максимальний пулінг дозволяє зробити модель менш чутливою до незначних варіацій;
- оптимізувати обчислення: зменшення розмірності входів до наступних шарів мережі значно скорочує витрати на обчислення.

У контексті аналізу медико-біологічних відеопотоків максимальний пулінг дозволяє виділяти динамічні зміни, наприклад, амплітуду рухів голосових зв'язок, ігноруючи менш важливі деталі. Реалізацію максимального пулінгу для аналізу відеоданих можна переглянути на лістингу 3.3.

Лістинг 3.3. Приклад застосування максимального пулінгу для аналізу відеоданих у Python

```
from tensorflow.keras import layers, models

# Створення моделі з максимальним пулінгом
model = models.Sequential([
    layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
input_shape=(128, 128, 1)),
    layers.MaxPooling2D(pool_size=(2, 2)), # Максимальний пулінг
    layers.Conv2D(filters=64, kernel_size=(3, 3), activation='relu'),
    layers.MaxPooling2D(pool_size=(2, 2)), # Другий шар максимального
пулінгу
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])

# Виведення архітектури моделі
model.summary()
```

Хоча максимальний пулінг ефективно зберігає ключові ознаки, він має певні обмеження, зокрема втрату інформації про точне розташування деталей у даних. Це може бути важливим у задачах, де критичним є збереження локальних просторових залежностей, наприклад, при аналізі анатомічних структур або виявленні дрібних патологічних змін. Однак його переваги роблять максимальний пулінг незамінним у більшості задач. Він виділяє найзначущіші ознаки, такі як контрастні краї чи інтенсивні

текстурні зміни, що є ключовими для діагностики. Крім того, максимальний пулінг суттєво зменшує обсяг даних, переданих до наступних шарів моделі, скорочуючи обчислювальні витрати та підвищуючи продуктивність. Це особливо важливо при роботі з високороздільними відеопотоками.

Отже результатом пропонованого методу попередньої обробки є відеоматеріал, який позбавлений надлишкових перешкод, вирівняний за інтенсивністю та адаптований за розміром для подальшої автоматизованої обробки. Отримані таким чином покращені відеодані створюють сприятливі умови для підвищення точності та швидкодії наступних етапів дослідження, аналізу та класифікації. У результаті знижується ймовірність спотворення змісту відеоматеріалу та зменшується кількість хибнопозитивних чи хибнонегативних визначень під час подальшого використання обробленого відео. Візуалізацію пропонованого методу можна переглянути на рис. 3.2.

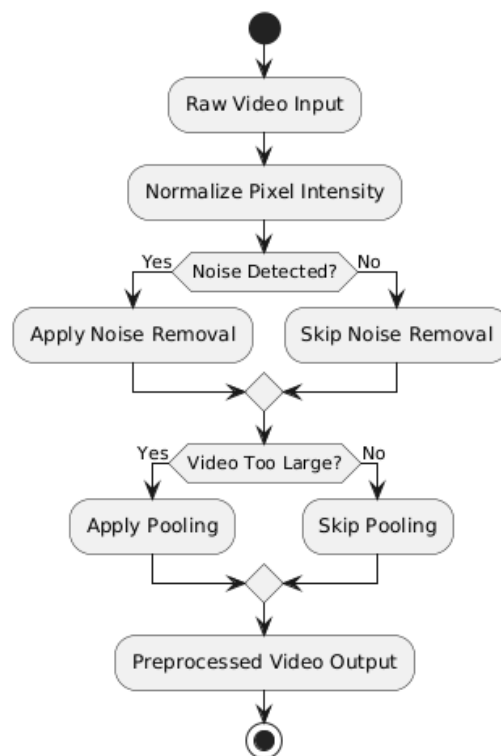


Рис. 3.2 Діаграма процесу попередньої обробки відео

3.3. Побудова цифрового двійника

Однією з важливих задач, які вирішується у цьому дослідженні, є розроблення методів і алгоритмів для створення цифрового двійника гортані, здатного адаптуватися до потокових мультимодальних даних (відео, аудіо), що надходять із діагностичних приладів. Такий підхід дозволяє точно відтворювати як анатомічні особливості, так і динаміку роботи гортані, що є критично важливим для діагностики та прогнозування стану пацієнта.

Сучасні підходи до моделювання цифрових двійників зосереджені на інтеграції даних із різних джерел. Особливу увагу приділяють синхронізації відео- та аудіопотоків у реальному часі, що забезпечує їх узгодженість і коректність аналізу. Це дозволяє використовувати відео для детального відображення структурних змін, а аудіодані — для оцінки функціональних характеристик, таких як робота голосових зв'язок.

У цьому розділі також розглядається архітектура нейронної мережі, яка поєднує здатність 3D-згорткових мереж аналізувати просторово-часові дані відеопотоків із можливістю рекурентних мереж обробляти послідовності аудіосигналів. Така комбінація дозволяє врахувати як просторові, так і часові залежності в даних, створюючи комплексну модель роботи гортані.

Ще одним важливим аспектом є адаптація базової тривимірної моделі гортані до індивідуальних характеристик пацієнта. Це досягається шляхом інтеграції результатів аналізу даних із деформацією базового шаблону, що дозволяє максимально точно відобразити анатомічні особливості та динамічну поведінку органу. Такий підхід забезпечує створення інтерактивної 3D-моделі, яка може використовуватися для діагностики та планування лікування.

Побудова цифрового двійника гортані включає кілька ключових етапів, починаючи з аналізу та обробки мультимодальних даних, що надходять із діагностичних приладів, і закінчуючи створенням

візуалізованої 3D-моделі. В попередніх підрозділах було детально описано збір і попередню обробку даних, які стали основою для подальшого моделювання цифрового двійника.

Авторський внесок у пропонований в розділі метод полягає у розробці інтегрованої архітектури, яка поєднує загальновідомі підходи оброблення відео (та аудіо) даних за пропонованою схемою схемою синхронізації та конкатенації ознак для адаптування базової 3D-моделі гортані до індивідуальних характеристик пацієнта шляхом.

Першим ключовим кроком пропонованого методу є виконання **згорткових операцій**, які дозволяють виділити значущі ознаки з відеоданих для подальшого моделювання. **Згорткові операції** (Convolution) є основою обробки зображень і відео за допомогою згорткових нейронних мереж. Згорткові операції виконуються шляхом застосування фільтра (ядра згортки) до вхідного зображення. Формально, згортка описується формулою (3.6).

$$O(x,y) = \sum \sum I(x + i, y + j) \times K(i,j) , \quad (3.6)$$

де

$O(x,y)$ — вихідне значення пікселя в точці (x, y) після згортки;

$I(x + i, y + j)$ — значення інтенсивності пікселя з вхідного зображення у відповідній точці;

$K(i, j)$ — значення ядра згортки (фільтра) у точці (i, j) ;

$\sum \sum$ — подвійна сума, яка виконується по всіх значеннях i та j , що визначають розмір ядра.

Фільтр є невеликою матрицею ваг, яка ковзає по зображенню, обчислюючи зважену суму інтенсивностей пікселів для кожної області. Наприклад, фільтр для виявлення країв виділяє різкі зміни яскравості, які відповідають межах об'єктів.

Для аналізу відеопотоків пропонується використати розширений підхід, що враховує не лише просторові, але й часові залежності. Це

досягається за допомогою тривимірних згорток (3D Convolution), які обробляють послідовності кадрів як єдиний об'єм даних. Обчислення 3D-згортки представлено у формулі (3.7).

$$O(x,y,t) = \sum \sum \sum I(x + i, y + j, t + k) \times K(i,j,k), \quad (3.7)$$

де

t — часовий індекс;

$\sum \sum \sum$ — потрійна сума, яка виконується по всіх значеннях i, j (просторові координати) і k (часовий зсув).

Така згортка дозволяє враховувати зміни інтенсивності пікселів у часі, що є критично важливим для аналізу динаміки, наприклад, руху голосових зв'язок або інших анатомічних структур.

Згорткові шари CNN включають:

1. **Фільтри (ядра).** Визначають, які ознаки будуть виявлятися, наприклад, краї, текстури чи форми.
2. **Крок згортки (stride).** Визначає, наскільки ядро зсувається під час обчислень. Менший крок забезпечує детальніший аналіз, але збільшує обчислювальну складність.
3. **Доповнення (padding).** Використовується для збереження розмірів вхідного зображення після згортки.

Згорткові операції дозволяють автоматично виділяти ознаки, які важко або неможливо виявити вручну, особливо в умовах складної структури медико-біологічних об'єктів.

- у випадку аналізу гортані CNN можуть виділяти ознаки патологій, таких як вузли чи поліпи голосових зв'язок;
- у динамічних даних 3D-згортки враховують часові зміни, що дозволяє відстежувати прогресування патологій або оцінювати ефективність лікування.

У лістингу 3.4 наведено приклад створення простого згорткового шару в Python за допомогою бібліотеки TensorFlow.

Лістинг 3.4. Реалізація згорткового шару

```
from tensorflow.keras import layers, models

# Створення моделі з 2D-згортковим шаром
model = models.Sequential([
    layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
input_shape=(128, 128, 1)),
    layers.MaxPooling2D(pool_size=(2, 2)),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])

# Виведення архітектури моделі
model.summary()
```

Отже, першим ключовим кроком у процесі створення цифрового двійника є вилучення з вхідних відеоданих найінформативніших ознак, необхідних для точної адаптації базової моделі. Цю задачу вирішують згорткові операції, що реалізуються за допомогою згорткових нейронних мереж (CNN). Застосування 2D-згорток до окремих кадрів відео дозволяє автоматично виділяти суттєві просторові ознаки — краї, текстури, характерні геометричні елементи. Однак статичний аналіз окремих кадрів не враховує зміни, які відбуваються в часі, а для медико-біологічних об'єктів саме динаміка рухів є критичною: періодичні коливання, зміни форми чи структури в процесі функціонування органу.

Саме тому наступним логічним кроком стає *аналіз відеоданих за допомогою тривимірних згорткових нейронних мереж (3D-CNN)*. На відміну від класичних 2D-CNN, які працюють зі статичними зображеннями, 3D-CNN дозволяють враховувати часовий вимір, обробляючи відео не як послідовність окремих кадрів, а як цілісний об'єм даних (spatio-temporal volume). Застосування тривимірних згорток забезпечує виділення просторово-часових ознак, що дає змогу моделювати такі характеристики, як амплітуда та частота рухів гортані, наявність патологічних змін, які проявляються у динаміці, або інші важливі часові патерни. Таким чином, перехід від 2D до 3D-згорток логічно виправданий необхідністю врахувати

часову складову даних та підвищити точність цифрового двійника за рахунок детального аналізу динамічних процесів.

Для моделювання динаміки гортані та виявлення просторово-часових залежностей у відеопослідовностях застосовуються тривимірні згорткові нейронні мережі (3D-CNN). На відміну від двовимірних згорток, які обробляють кожен кадр незалежно, 3D-згортки розглядають відеоряд як об'єм даних із тривимірною структурою (висота, ширина, глибина) та часовою складовою. Це дозволяє враховувати зміни у часі та моделювати динамічну поведінку органу.

Якщо вхідний відеоблок представлено у вигляді чотиривимірного тензора I розмірністю $T \times H \times W \times C$, де:

T – кількість кадрів у блоці;

H, W – висота та ширина кадру;

C – кількість каналів (наприклад, три для RGB).

То тривимірна згортка полягає у зваженому сумуванні пікселів у локальній тривимірній області (у просторі та часі) із відповідним ядром згортки. Таким чином, кожна операція згортання враховує не лише сусідні пікселі у просторі, а й сусідні кадри у часі. Це дає змогу виділяти ознаки, пов'язані з рухом, змінами текстур та іншими динамічними характеристиками, важливими для аналізу гортані. Приклад архітектури 3D-CNN наведено у лістингу 3.5.

Лістинг 3.5 – Приклад архітектури 3D-CNN мовою Python

```
from tensorflow.keras import layers, models

model = models.Sequential([
    layers.Conv3D(32, (3, 3, 3), activation='relu', input_shape=(16, 128,
128, 3)),
    layers.MaxPooling3D((2, 2, 2)),
    layers.Conv3D(64, (3, 3, 3), activation='relu'),
    layers.MaxPooling3D((2, 2, 2)),
    layers.Conv3D(128, (3, 3, 3), activation='relu'),
    layers.Flatten(),
    layers.Dense(256, activation='relu'),
    layers.Dense(3, activation='linear') # Наприклад, координати адаптації
])
```

У наведеній архітектурі аналізуються відеоблоки з 16 кадрів розміром 128×128 пікселів та 3 каналами (RGB). Перші 3D-згорткові шари (Conv3D) виділяють елементарні просторово-часові ознаки, починаючи з базових (краї, локальні контрастні елементи, прості рухи) і поступово переходячи до більш складних структур (зміни форми, текстури та періодичні коливання гортані). Шари підвибірки (MaxPooling3D) зменшують розмір даних, зберігаючи найбільш значущі ознаки та знижуючи обчислювальну складність. Отримані ознаки потім перетворюються у вектор (шар Flatten) для інтеграції за допомогою повнозв'язних шарів (Dense).

Останній шар з лінійною активацією призначений для регресійних завдань, наприклад, визначення параметрів деформації моделі гортані. Таким чином, поєднання 3D-згорток, пулінгу та щільних шарів дає змогу не лише відобразити статичні просторові особливості об'єкта, а й зрозуміти динаміку його функціонування, що є критично важливим для точного та персоналізованого цифрового двійника.

Додатковим джерелом інформації для підвищення точності адаптації цифрового двійника до медико-біологічного об'єкта **можуть слугувати аудіодані**, синхронізовані з відеопотоком (див. попередню главу). На відміну від статичних зображень, аудіосигнал є динамічним у часі, а його характеристики (частота, амплітуда, гармонічна структура) відображають функціональні аспекти роботи органу, наприклад, голосових

зв'язок. Припустімо, що аудіодані перетворено на послідовність ознак довжиною L , де кожен із L часових кадрів містить вектор із M акустичних параметрів (спектральні коефіцієнти, мел-кепстральні ознаки тощо). Така представлення дає змогу застосовувати рекурентні нейронні мережі, зокрема архітектури типу LSTM (Long Short-Term Memory). Основна перевага LSTM полягає у здатності утримувати довгострокові залежності та моделювати часові закономірності, уникаючи проблеми затухання чи вибуху градієнтів, що притаманна класичним RNN.

Використання LSTM для обробки аудіоданих дозволяє виявити приховані патерни, пов'язані з акустичними характеристиками органу. Наприклад, можна виявити специфічні зміни голосових характеристик, пов'язані з періодичністю рухів голосових зв'язок, а також слідкувати за довготривалими варіаціями, що можуть свідчити про патології чи функціональні відхилення. Отримані результати інтегруються з просторово-часовою інформацією, виділеною за допомогою 3D-CNN із відеопотоку, утворюючи мультимодальний підхід до формування цифрового двійника.

Оскільки аудіодані є опціональним компонентом, блок LSTM включають до загальної архітектури лише у тих випадках, коли акустична інформація дійсно сприяє підвищенню точності чи інформативності моделі. Якщо ж внесок аудіо не є суттєвим, цей блок можна пропустити. Таким чином, забезпечується гнучкість та адаптивність системи, що є важливим для її універсального застосування у різноманітних клінічних сценаріях. Реалізацію можна переглянути у лістингу 3.6.

Лістинг 3.6. Оброблення аудіоданих

```
audio_model = models.Sequential([
    layers.LSTM(128, return_sequences=True, input_shape=(None,
audio_features)),
    layers.LSTM(64),
    layers.Dense(32, activation='relu'),
    layers.Dense(3, activation='linear') # Параметри деформації моделі
])
```

У наведеному прикладі аудіодані подаються до вхідного шару LSTM у вигляді послідовності фреймів. Перший LSTM-шар із 128 одиницями обробляє ці дані та зберігає часові залежності, повертаючи повну послідовність вихідних станів. Другий LSTM-шар із 64 одиницями стискає інформацію до більш абстрактних ознак. Потім застосовуються щільні шари (Dense), щоб інтегрувати виділені часові патерни в компактні представлення, придатні для прогнозування параметрів адаптації моделі. Вихідний шар з лінійною активацією призначений для регресійних завдань, наприклад, оцінювання координат чи інших числових параметрів, необхідних для деформації базової моделі. Такий підхід до опрацювання аудіоданих розширює функціональні можливості цифрового двійника, дозволяючи йому відображати не лише структурну та динамічну конфігурацію органу (отриману з відео), але й акустичні характеристики його роботи, що є важливим аспектом персоналізованої медицини.

Процес створення адаптивного цифрового двійника медико-біологічного об'єкта передбачає інтеграцію різних типів інформації. Відеодані забезпечують детальний опис просторово-часових характеристик органу, тоді як аудіо дають змогу аналізувати акустичні аспекти, що відображають динаміку його функціонування. Окремий аналіз цих каналів корисний, проте поєднання відео- та аудіоінформації дозволяє створити повнішу модель, здатну одночасно враховувати структурні та функціональні особливості.

Ідея мультимодального підходу полягає у тому, що різні типи даних несуть взаємодоповнюючу інформацію. Наприклад, рух голосових зв'язок, зафіксований у відео, може корелювати із змінами частоти чи амплітуди акустичного сигналу. Врахування обох аспектів дає змогу точніше виявити закономірності та передбачити стан органу. Це особливо важливо для персоналізованої медицини, де необхідна комплексна оцінка, що охоплює як морфологічні, так і функціональні характеристики.

Після *виділення просторово-часових ознак із відео* за допомогою 3D-CNN та *часових патернів із аудіо за допомогою LSTM* обидва вектори ознак об'єднуються (конкатенуються) в єдиний простір. Така операція дозволяє моделі розглянути зв'язки між різними модальностями та виявити складні кореляції, недоступні під час аналізу поодиноких потоків. У результаті формується мультимодальне представлення, яке можна використовувати для прогнозування параметрів адаптації базової моделі, уточнення деформацій тривимірного двійника або інших релевантних цільових змінних.

Після конкатенації ознак створюється фінальна модель, яка навчається на мультимодальних даних. На етапі навчання ваги оптимізуються так, щоб мінімізувати функцію втрат, забезпечуючи найкращу відповідність між прогнозованими параметрами та реальними даними.

Таким чином, модель стає здатною враховувати одночасно візуальну (структурну) та акустичну (функціональну) інформацію, що підвищує точність та надійність одержаних результатів, реалізацію цього можна переглянути на лістингу 3.7.

Лістинг 3.7. Комбінування вихідних представлень відео та аудіо

```
from tensorflow.keras import layers, Model, Input
video_input = Input(shape=(16, 128, 128, 3))
audio_input = Input(shape=(None, audio_features))

video_features = model(video_input)
audio_features = audio_model(audio_input)

combined = layers.concatenate([video_features, audio_features])
output = layers.Dense(3, activation='linear')(combined)

final_model = Model(inputs=[video_input, audio_input], outputs=output)
final_model.compile(optimizer='adam', loss='mse')
```

У представленому прикладі ознаки, виділені 3D-CNN із відеопотоку та LSTM з аудіопотоку, конкатенуються в один вектор. Далі щільний шар (Dense) формує прогноз адаптаційних параметрів. Завдяки такому підходу модель аналізує обидва типи даних одночасно, що забезпечує більш інформативний прогноз, враховуючи як структурно-анатомічні, так і функціонально-акустичні аспекти роботи гортані. Такий мультимодальний підхід є важливим кроком до створення повноцінних цифрових двійників, здатних точно моделювати реальні медико-біологічні процеси.

Після завершення процесу навчання мультимодальної моделі, яка враховує як візуальну (відео), так і акустичну (аудіо) інформацію, **настає ключовий етап** — використання отриманих параметрів для анімації узагальненої тривимірної моделі гортані. На початку дослідження маємо лише статичну 3D-модель, що описує загальну форму та структуру органу. Завдання полягає в тому, щоб «оживити» цю базову модель, передати їй індивідуальні риси пацієнта та відтворити динаміку, зафіксовану у відеопотоці та аудіосигналах.

Завдяки алгоритмам глибокого навчання та мультимодальній інтеграції мережа тепер може прогнозувати параметри, які визначають зміни форми, позиції та деформації окремих елементів гортані у часі. Ці

параметри включають координати точок, що відповідають за геометрію голосових зв'язок, амплітуди коливань та часові затримки, а також інші ключові ознаки, пов'язані з рухами та акустичною активністю органу. Оскільки модель була навчена на реальних даних, отриманих з відеозаписів роботи гортані та супутніх аудіосигналів, вона здатна відтворити природну поведінку органу, а не просто абстрактні рухи.

Інтегруючи прогнозовані параметри у вихідну 3D-модель, ми поступово змінюємо її геометрію у відповідності до динаміки, зафіксованої на відео, та акустичних патернів, виявлених у аудіоданих. Наприклад, якщо відео показує періодичні рухи голосових зв'язок з певною частотою та амплітудою, а аудіо свідчить про відповідні зміни у звукових характеристиках, модель враховує ці дані, коригуючи положення поверхонь гортані в кожному часовому кроці. Таким чином, статична модель перетворюється на анімований цифровий двійник, який відображає не лише анатомію, а й функціональність органу.

Нижче у псевдокоді лістингу 3.8 наведено приклад коду, який демонструє, як після отримання прогнозованих параметрів (наприклад, координат для деформацій, значень амплітуд чи фреймів анімації) можна застосувати їх до узагальненої 3D-моделі гортані. Зверніть увагу, що це демонстраційний код — у реальних умовах буде потрібна інтеграція зі спеціалізованими бібліотеками для 3D-графіки та медичної візуалізації (наприклад, VTK, Mayavi чи PyVista), а також наявність відповідної 3D-моделі гортані у вигляді сітки (mesh).

Лістинг 3.8. Приклад застосування прогнозованих параметрів деформації до узагальненої 3D-моделі гортані для отримання анімованого цифрового

```
import numpy as np

def deform_model(params, base_mesh):
    # Припустимо, params - (N×3) із зсувами (dx, dy, dz) для кожної з N
    # вершин
    deformed_mesh = base_mesh.copy()
    deformed_mesh += params
    return deformed_mesh

def render(mesh):
    # Псевдокод для візуалізації/рендерингу 3D-сцени
    pass

# Завантаження базової 3D-моделі
base_mesh = np.load('base_mesh.npy') # масив розміром (N×3)

# Завантаження набору зсувів, передбачених мультимодальною моделлю
predicted_vertex_shifts = np.load('predicted_deformations.npy') # (T×N×3)

# Для кожного кадру оновлюємо сітку та візуалізуємо
for t in range(predicted_vertex_shifts.shape[0]):
    current_params = predicted_vertex_shifts[t] # (N×3)
    deformed_mesh = deform_model(current_params, base_mesh)
    render(deformed_mesh)

print("Анімація завершена. Цифровий двійник відтворено на основі відео- та
аудіоданих.")
```

Загальна ідея пропонованого методу полягає у завантаженні базової 3D-моделі гортані та послідовності деформацій, отриманих мультимодальною моделлю, зокрема 3D-CNN + LSTM. Базова модель зберігається у масиві `base_mesh` розміром N на 3 , де N — кількість вершин, а кожен рядок відповідає координатам (x , y , z) певної вершини. Зсуви (`predicted_vertex_shifts`) мають формат T на N на 3 , де T — кількість кадрів анімації, а кожен блок N на 3 містить зсуви dx , dy та dz для всіх вершин. У циклі перебираються всі часові кроки, і для кожного з них програма бере поточний зріз масиву зсувів та додає його до координат вершин базової сітки, створюючи змінену форму гортані. Функція `render(mesh)` відповідає за відтворення сцени на кожному кроці, демонструючи, як змінюється модель у часі.

Функція `deform_model(params, base_mesh)` отримує на вхід параметри зсувів (N на 3) та вихідну сітку (N на 3), повертаючи результат їх додавання. Функція `render(mesh)` у реальному проєкті може використовувати сторонні бібліотеки для відображення змінених координат вершин.

Основний цикл анімації включає початкове завантаження базової моделі й масиву зсувів, а також послідовне додавання зсувів на кожному кроці з подальшим рендерингом отриманої деформованої сітки. Це дає змогу побачити, як форма гортані змінюється відповідно до даних мультимодальної системи.

Мультимодальна модель (3D-CNN + LSTM) виконує аналіз відео й аудіо, прогнозуючи, як саме рухається гортань у часі. Всі обчислення залишаються “за лаштунками”, оскільки в коді вже використовуються готові зсуви, збережені у `predicted_vertex_shifts`.

Практично цей метод дає змогу створювати медичні симуляції, де рухи гортані візуалізують у реальному часі, аналізувати акустичні ефекти, пов’язані зі змінами у формі гортані, а також застосовувати анімовану модель у віртуальних середовищах чи навчальних матеріалах. У цьому прикладі показано спрощений спосіб переходу від статичної 3D-моделі до динамічного цифрового двійника, що змінює форму відповідно до реальних або змодельованих даних.

Таким чином, код ілюструє концепцію переходу від статичної 3D-моделі до анімованого цифрового двійника гортані, динамічні характеристики якого визначаються попередньо проаналізованими відео-та аудіоданими. Загальну візуалізацію сформульованого методу можна побачити на рис. 3.3.

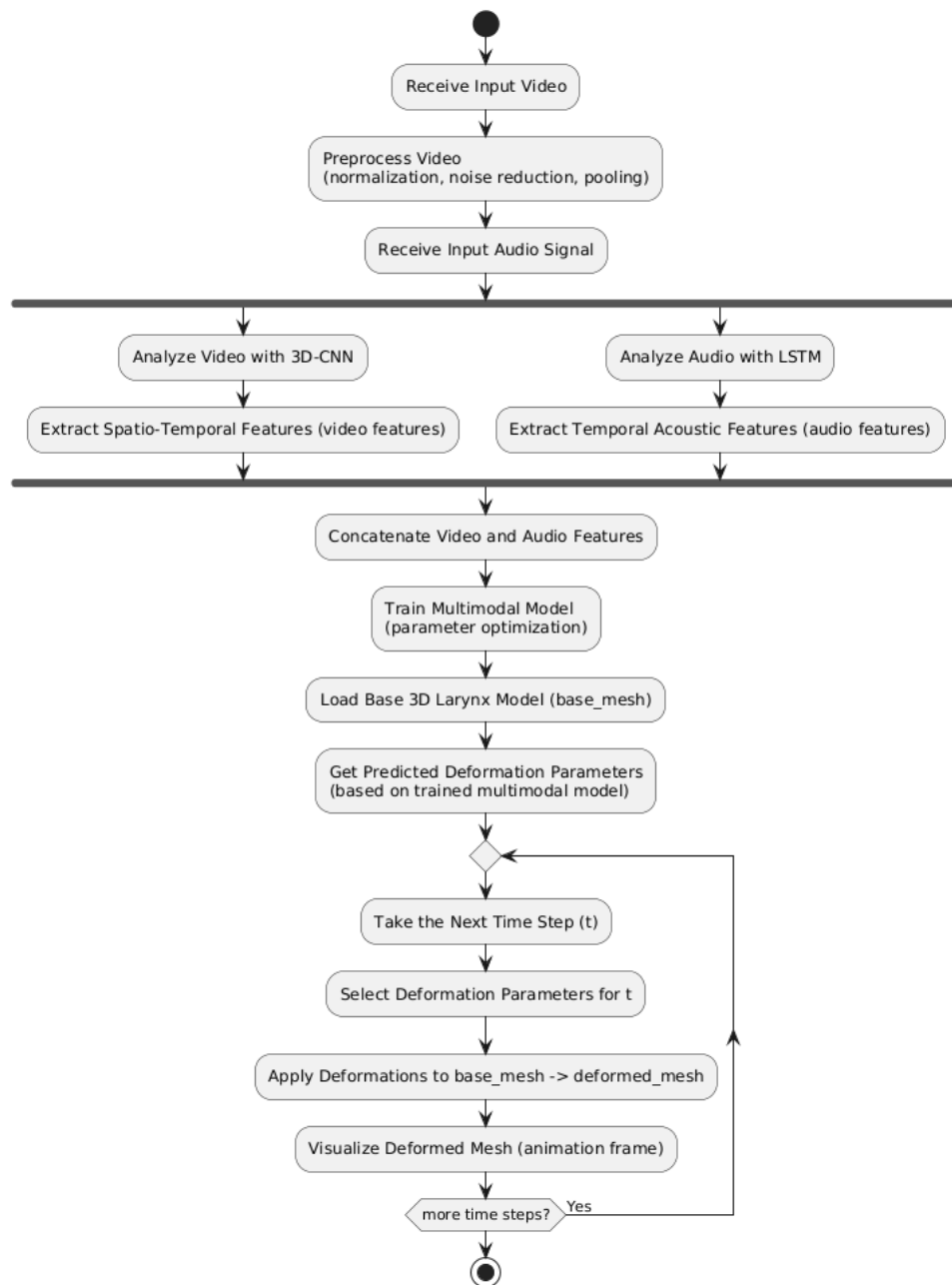


Рис. 3.3 Діаграма активності методу програмної адаптації базової моделі медико-біологічного об'єкта

3.4. Висновки до розділу 3

У цьому розділі представлено методи обробки, візуалізації та інтеграції мультимодальних даних для створення цифрових двійників медико-біологічних об'єктів. Увага акцентується на особливостях мультимодальних даних, таких як їхня інтеграція у часі та просторі, що відіграє важливу роль у створенні високоточної моделі гортані. Розглядаються підходи до узгодження відео- та аудіопотоків, отриманих у режимі реального часу, з метою забезпечення точності та узгодженості аналізу.

На основі проведеного аналізу сформовано методологічну базу для адаптації базового тривимірного шаблону гортані до реальних даних. У розділі представлено комбіновані підходи до обробки відео- та аудіоданих: використання сучасних нейронних мереж для аналізу просторово-часових залежностей у відеопотоках та виявлення послідовностей у звукових сигналах. Продемонстровано, що інтеграція різних типів даних у єдину модель дозволяє максимально ефективно використовувати доступну інформацію, забезпечуючи точне відображення динаміки роботи гортані.

Особливу увагу приділено методам корекції та нормалізації вхідних даних, що сприяє підвищенню якості аналізу. Також описано принципи адаптації тривимірної моделі гортані до індивідуальних особливостей пацієнтів, з урахуванням як геометричних, так і текстурних характеристик. Це дозволяє створювати точні цифрові двійники, здатні відображати складні біологічні процеси з високою деталізацією.

У результаті розроблено метод, який підтримує масштабованість і можливість інтеграції нових типів даних, що відкриває перспективи для персоналізованої медицини. Запропоновані підходи спрямовані на забезпечення точності діагностики, покращення прогнозування стану пацієнтів і створення основ для ефективного планування лікувальних заходів.

РОЗДІЛ 4. РОЗРОБЛЕННЯ АРХІТЕКТУРИ МЕДИЧНОЇ ПРОГРАМНОЇ СИСТЕМИ НА ОСНОВІ ТЕХНОЛОГІЇ ЦИФРОВИХ ДВІЙНИКІВ

Цифровізація охорони здоров'я є одним із ключових напрямків розвитку медичної галузі, що сприяє підвищенню ефективності досліджень, діагностики та лікування. Одним із перспективних підходів у цьому контексті є впровадження цифрових двійників медико-біологічних об'єктів, таких як гортань людини. Цифрові двійники дозволяють точно відтворювати фізіологічні та функціональні характеристики органів, створюючи можливості для їх детального вивчення та моделювання в умовах, наближених до реальних. Проте цей процес супроводжується низкою технічних та організаційних викликів.

Одним із головних завдань є забезпечення якості та верифікації цифрових двійників. Оскільки такі системи складаються з численних компонентів, які виконують окремі функції, критично важливо забезпечити їх правильну роботу та надійність даних. Найменші помилки можуть призвести до некоректних висновків або невірних медичних рішень, що підкреслює необхідність ретельного тестування кожного компонента та всієї системи загалом.

Модульність та ізоляція компонентів є ще однією складністю. Сучасні системи будуються на основі розподіленої архітектури, що передбачає незалежну роботу модулів. Це ускладнює процес верифікації, оскільки необхідно перевіряти не лише функціональність окремих частин, а й їхню сумісність у межах єдиної системи.

Для вирішення цих завдань важливим є автоматизоване тестування. Ручне тестування у динамічних системах, що постійно оновлюються, є трудомістким і неефективним. Використання автоматизованих тестових сценаріїв дозволяє прискорити процес перевірки, забезпечуючи при цьому високу точність і повторюваність результатів.

Також значну роль відіграє контейнеризація та мікросервісна архітектура. Контейнеризація дає змогу створювати ізольовані середовища для розробки, тестування та розгортання, що забезпечує гнучкість системи та спрощує її верифікацію. Мікросервісна архітектура дозволяє проводити незалежне тестування кожного модуля та пришвидшує процес виявлення й усунення помилок.

Додатково до цього, інтеграція систем безперервної інтеграції та доставки (CI/CD), фреймворків для тестування, а також інструментів моніторингу та логування є критичною. Вони дозволяють автоматизувати процеси тестування та розгортання, а також забезпечують моніторинг роботи системи для своєчасного виявлення проблем.

Таким чином, створення архітектури медичної програмної системи на основі цифрових двійників медико-біологічних об'єктів вимагає комплексного підходу, який поєднує модульність, автоматизацію тестування, контейнеризацію, мікросервісну архітектуру та сучасні інструменти для забезпечення якості. Лише дотримання цих принципів дає змогу створювати надійні системи, що сприяють підвищенню точності діагностики, прогнозуванню стану пацієнтів та ефективному плануванню лікувальних заходів.

4.1. Архітектурні шаблони проєктування для розроблення мультимедійного програмного забезпечення

Архітектурні шаблони проєктування становлять собою базовий інструмент сучасної розроблення програмного забезпечення. Вони представляють стандартизовані та параметризовані підходи до побудови архітектури систем, що дозволяє ефективно вирішувати типові завдання розроблення. Завдяки використанню цих шаблонів, розробники отримують структуровані рекомендації для організації архітектури та коду, що значно

спрощує процес адаптації, розширення та масштабування програмних систем.

Однією з ключових переваг застосування архітектурних шаблонів є зниження складності розроблення, що безпосередньо впливає на підвищення якості кінцевого продукту. Стандартизована організація коду сприяє уникненню повторень, підвищує його читабельність та спрощує подальшу підтримку. Крім того, шаблони стимулюють повторне використання перевірених рішень, що є важливим чинником у створенні модульних та гнучких систем, а також зменшує ризик виникнення помилок під час розроблення.

Особливу актуальність архітектурних шаблонів має у сфері мультимедійного програмного забезпечення. Оброблення мультимедійних даних вимагає високого рівня абстракції та інтеграції різних модальностей, що створює додаткові виклики для розробників. Впровадження спеціалізованих шаблонів проєктування дозволяє стандартизувати підхід до таких систем, сприяючи оптимізації процесу створення продукту та забезпечуючи його надійність і ефективність.

Сформулюємо такі шаблони для використання під час подальшого розроблення узагальненої архітектури. Нижче наведено формулювання схеми для опису шаблонів проєктування:

- Intent (Намір). Відповідає на запитання, яку проблему вирішує даний шаблон.
- Motivation (Мотивація). Опис сценарію або ситуації, яка демонструє архітектурну проблему, а також пояснює, як конкретні елементи шаблону сприяють її розв'язанню.
- Applicability (Застосовність). Визначення тих умов або контекстів, у яких шаблон може бути застосований, а також ознак, що дозволяють його розпізнати.

- Structure (Структура). Графічне представлення компонентів шаблону у вигляді діаграми.
- Consequences (Наслідки). Аналіз переваг та можливих недоліків застосування шаблону.
- Implementation (Реалізація). Рекомендації та поради щодо практичного впровадження шаблону.
- Related Patterns (Пов'язані шаблони). Перелік шаблонів, тісно пов'язаних із даним, з описом їх відмінностей та взаємозв'язків.

.1 Шаблон «Мультимедійний Конвеєр»

Intent (Намір). Шаблон «Мультимедійний конвеєр» орієнтований на розбиття складної задачі обробки мультимедійних даних на незалежні, спеціалізовані модулі. Основна ідея полягає в тому, щоб кожен етап обробки, від декодування до постобробки, виконував свою чітко визначену функцію. Це дозволяє підвищити якість обробки, спростити тестування і налагодження, а також забезпечити масштабованість системи, де окремі модулі можна оновлювати або замінювати без впливу на загальну роботу платформи.

Motivation (Мотивація). Уявимо систему, яка обробляє потокове відео з камер спостереження. Сирі дані можуть містити численні шуми, артефакти, нерівномірне освітлення, що знижує їхню якість і ускладнює подальший аналіз. Якщо всю обробку виконувати монолітно, це призведе до затримок, складнощів із масштабуванням і підтримкою. Завдяки застосуванню «Мультимедійного конвеєра» процес розбивається на кілька етапів: спочатку дані декодуються і перетворюються у внутрішній уніфікований формат із базовою корекцією, далі застосовуються фільтри для усунення шумів, потім проводиться аналіз і розпізнавання ознак за допомогою спеціалізованих алгоритмів, і завершальною стадією є постоброблення, що готує дані для збереження або подальшого

використання. Такий підхід дозволяє оптимізувати кожен етап окремо і забезпечити високу ефективність системи.

Applicability (Застосовність). Шаблон «Мультимедійний конвеєр» є корисним у випадках, коли система повинна обробляти потокові мультимедійні дані в режимі реального часу, або коли дані надходять із різних джерел і мають різноманітні формати. Він підходить для систем відеоспостереження, аудіопроектингу, онлайн-трансляцій, а також для проектів, де важлива гнучкість і можливість масштабування окремих компонентів без перебудови всієї архітектури.

Structure (Структура). Структура шаблону організована як послідовний ланцюг модулів, що починається з прийому сирих даних і завершується формуванням остаточного вихідного сигналу. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 4.1. Нижче більш детально розпишемо структурні частини пропонованого шаблону.

- Data. Виконує роль контейнера або структури для зберігання мультимедійних даних. Може містити різні формати (відео, аудіо, зображення, метадані тощо) і додаткову інформацію про потік даних (наприклад, часові мітки).
- Source. Відповідає за отримання початкових даних із певного джерела: файлів, мережевого потоку, апаратних пристроїв на кшталт камер чи мікрофонів. Формує об'єкт Data, який потім передається далі по “конвеєру”.
- IFilter (інтерфейс або базовий клас фільтрів). Містить метод `process(data: Data): Data` (або аналогічний), у якому виконується оброблення вхідних даних. Реальні підкласи можуть виконувати різні функції: декодування, фільтрацію шуму, аналіз, зміну формату тощо.
- Sink. “Кінцева точка” обробки, де результат виводиться або зберігається. Забезпечує виклик методу `output(data: Data)`, який

завершує ланцюг обробки. Може зберігати готові дані у файл, передавати їх по мережі або відправляти на відтворення.

- PipelineManager. Відповідає за керування всіма етапами: створює та конфігурує Source, IFilter та Sink, організовує їх у послідовність, ініціалізує та завершує обробку (start/stop), а також відстежує помилки. Може забезпечувати паралельну роботу, масштабування та черги даних між етапами.

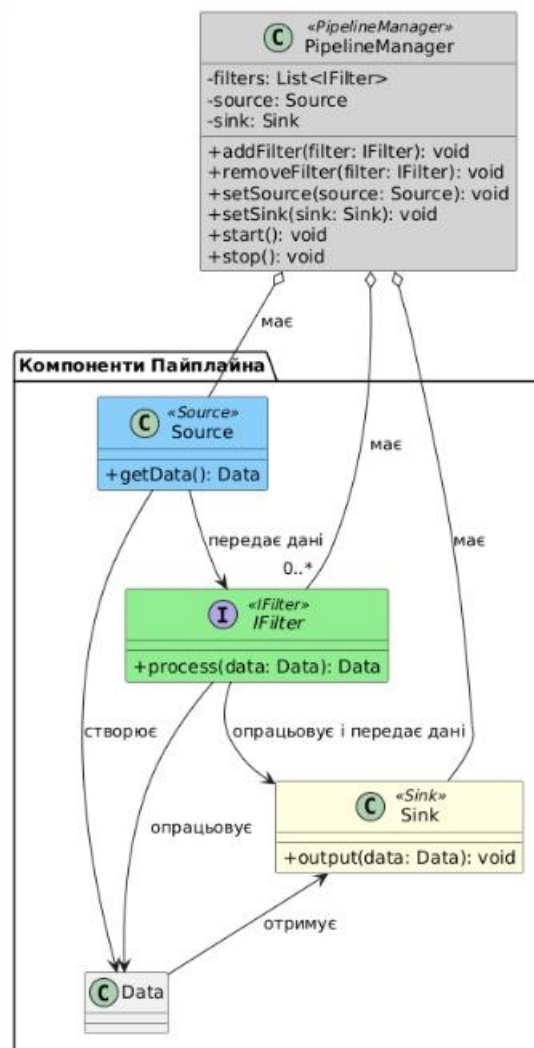


Рис. 4.1. Структурна діаграма запропонованого архітектурного шаблону проектування «Мультимедійний Конвеєр»

Під час ініціалізації PipelineManager отримує або створює Source, один чи декілька IFilter і Sink, а також визначає порядок їх використання.

Зчитування даних відбувається в Source: він формує об'єкт Data і може повертати його PipelineManager або напрямку передавати наступному етапу.

Проходження через фільтри реалізується викликом методу process на кожному IFilter, де дані змінюються або аналізуються. Після обробки результат надходить до наступного фільтру або до фінальної точки.

Вивід чи збереження даних здійснюється у Sink: метод output завершує обробку, записуючи готові результати на диск, передаючи в мережу чи відтворюючи їх у реальному часі.

Закриття та оброблення помилок координується PipelineManager. Якщо фільтр або джерело даних повідомляють про помилку, він може перезапустити етап чи зупинити весь процес. Після штатного завершення обробки PipelineManager викликає stop, звільняючи всі ресурси.

Додаткові сценарії передбачають паралельне або розподілене виконання фільтрів, використання буферів між етапами для кращої продуктивності, а також гнучке масштабування у разі потреби обробки великих потоків або багатьох джерел одночасно.

Implementation (Реалізація). При створенні Мультимедійного Конвеєра корисно почати з визначення базових інтерфейсів або абстрактних класів для кожного типу компонента (Source, Filter, Sink). Такий підхід чітко розмежовує відповідальність: Source отримує дані, Filter обробляє, а Sink виводить або зберігає. Також варто продумати, чи працюватимуть фільтри послідовно (кожен наступний отримує результат попереднього), чи передача даних між ними відбуватиметься асинхронно.

Для керування виконанням можна створити спеціальний модуль або клас (PipelineManager), який ініціалізує Source і Sink, формує список або ланцюг фільтрів, а потім послідовно чи паралельно передає між ними дані.

Це допоможе централізовано керувати запуском, зупинкою, обробкою помилок та моніторингом. Лістинг пропонованого шаблону наведено у додатку 1.

Related Patterns (Пов'язані шаблони). Мультимедійний Конвеєр може бути поєднаний зі «Спостерігачем» (Observer) або «Публікація-Підписка» (Publisher-Subscriber), коли потрібно повідомляти про зміну стану даних або завершення обробки. Це особливо корисно, якщо дані можуть надходити нерегулярно і необхідно динамічно реагувати на появу нових фреймів чи пакетів.

Шаблон «Мікросервіси» (Microservices) може бути дотичний у тих випадках, коли кожен фільтр або група фільтрів розгортається як окремий сервіс, що працює в контейнерному середовищі. Тоді PipelineManager або аналогічний диспетчер виступає в ролі оркестратора, керуючи з'єднаннями між сервісами.

Шаблон «Фільтр-Ланцюг» (Filter-Chain або Chain of Responsibility) має схожий механізм «проходження» через послідовність обробників. Проте, на відміну від Мультимедійного Конвеєра, в Filter-Chain кожен обробник вирішує, чи передавати дані наступному, чи перервати ланцюг. У конвеєрі ж, як правило, кожен фільтр виконує обов'язкову операцію над даними і передає результат наступному етапу.

Consequences (Наслідки). Застосування «Мультимедійного Конвеєра» розподіляє обробку мультимедійних даних на чітко визначені етапи, кожен з яких легко адаптувати та розширювати. Це спрощує розробку і тестування, адже можна зосередитися на логіці одного етапу, не зачіпаючи інші. Такий підхід також сприяє масштабуванню: окремі фільтри можна розташувати в різних потоках чи навіть на різних серверах, збільшуючи пропускну здатність системи. Архітектура шаблону полегшує заміну або додавання нового фільтра без необхідності переписувати всю систему.

До можливих недоліків належать ускладнення налагодження, якщо система має велику кількість фільтрів з різними алгоритмами та форматами даних. У таких випадках потрібно відстежувати, де саме виникає збій, і керувати чергами даних між етапами. Крім цього, модель конвеєра може викликати затримки в реальному часі, якщо один із фільтрів працює повільніше за інші, тож інколи потрібні додаткові заходи (буферизація або асинхронна оброблення) для збалансування швидкості виконання.

Шаблон «Мультимедійний Плагін»

Intent (Намір). Шаблон Plugin-based Multimedia Architecture (або «Архітектура з плагінами для мультимедіа») призначений для побудови системи, де легко додавати або оновлювати підтримку нових мультимедійних форматів, кодеків, фільтрів та функціональних можливостей без змін у базовому ядрі додатка. Основна ідея полягає у відокремленні ядра системи (модуля, що керує завантаженням мультимедійних даних і надає загальні механізми) від спеціалізованих плагінів, кожен з яких реалізує обробку певного типу файлу чи потоку.

Motivation (Мотивація). У мультимедійних системах постійно зростає кількість різноманітних форматів даних, які можуть значно відрізнятися за кодуванням, контейнерами та специфікаціями. Коли система потребує підтримки багатьох таких форматів одночасно, традиційна монолітна модель призводить до значного розростання коду та ускладнює його підтримку. Натомість плагінний підхід відокремлює все, що стосується обробки конкретного формату чи групи форматів, у незалежні модулі. Це дає змогу оновлювати або підміняти підтримку одного формату без ризику порушити стабільність усієї системи.

Ще одним чинником є потреба регулярного розширення функціональності, особливо коли потрібно оперативно реагувати на нові стандарти або вимоги ринку. Плагіни можна розробляти паралельно

командою чи сторонніми розробниками, водночас ядро системи залишається незмінним. Достатньо створити та підключити новий модуль, який дотримується встановлених інтерфейсів. Такий метод полегшує також процес тестування: замість перевірки всієї програми можна сфокусуватися на роботі нового плагіна у визначеному середовищі.

У великих корпоративних чи спеціалізованих системах (наприклад, у медицині) різні заклади можуть мати власні вимоги до форматів та алгоритмів обробки. Плагінна архітектура дає їм можливість використовувати загальне ядро, додаючи специфічні плагіни, які задовольняють локальні потреби. Це прискорює впровадження та спрощує підтримку, оскільки користувачі можуть отримувати оновлення й нові модулі без критичного впливу на решту функцій системи.

Applicability (Застосовність). Модель плагінів стає особливо корисною, якщо програмі доводиться працювати з динамічним набором мультимедійних форматів або способів обробки, коли з часом може виникнути потреба швидкого підключення додаткових алгоритмів. Застосування актуальне для мультимедійних плеєрів і редакторів, яким постійно доводиться оновлюватися під нові кодеки та розширення. Такий самий підхід дієвий у вебзастосунках, де сервер або клієнт можуть завантажувати плагіни для специфічної обробки аудіо- й відеопотоків, включно з накладанням субтитрів або додаткових ефектів.

У медичних та наукових сферах великою перевагою плагінної архітектури є можливість інтеграції з різними спеціалізованими форматами зображень або відео, наприклад, з даними різних типів томографії чи ультразвукових досліджень. Завдяки єдиному ядру і незалежним модулям легко обслуговувати й розвивати систему, додаючи обробку нових форматів чи інструментів аналізу. Ігрові рушії також виграють від цієї концепції, адже графічні та аудіоефекти часто вимагають оновлень, а відтак

функціональність можна розширювати невеликими плагінами без зміни усієї програми.

Structure (Структура). Структура патерну організована як ядро системи, яке оперує мультимедійними даними, та сукупність автономних плагінів, кожен із яких реалізує певний тип обробки чи підтримку конкретного формату. Ядро містить менеджер плагінів, що завантажує, реєструє та вивантажує окремі модулі. Завдяки цьому плагіни взаємодіють із базовим інтерфейсом (наприклад, `IPlugin`), але не впливають безпосередньо одне на одного чи на саме ядро. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 4.2. Нижче більш детально розпишемо структурні частини пропонованого шаблону.

- `MediaData`. Загальний контейнер або структура для зберігання медіаданих (аудіо, відео, зображення), метаданих (назва, розмір, часова позначка) та інформації про формат.
- `IPlugin`. Інтерфейс або абстрактний клас для плагінів. Містить методи `supportsFormat(data: MediaData): bool` та `process(data: MediaData): MediaData`. Завдяки цьому інтерфейсу різні реалізації можуть перевіряти, чи підтримується файл (потік), та виконувати потрібну обробку (декодування, рендеринг, фільтрацію).
- `AudioPlugin`, `VideoPlugin` (як приклади реалізацій `IPlugin`). Конкретні плагіни, що спеціалізуються на певному форматі або групі форматів. Наприклад, `AudioPlugin` може знати, як обробляти MP3/FLAC, а `VideoPlugin` — H.264/VP9 тощо.
- `PluginManager`. Збирає всі наявні плагіни і керує ними: додає, видаляє, індексує. За запитом (коли в систему надходить новий `MediaData`) визначає, який плагін може його обробити. За потреби ініціалізує чи звільняє ресурси плагіна.
- `MultimediaEngine`. Основне ядро або «двигун» мультимедійної програми. Має посилання на `PluginManager` і викликає його, коли

потрібно відкрити чи обробити певний файл/потік. Відповідає за оркестрацію: отримати від плагіна оброблені дані, передати їх далі (на відтворення, зберігання).

Перший етап передбачає завантаження та реєстрацію плагінів. Менеджер плагінів ініціює процес додавання кожного нового модуля, викликаючи метод, що повідомляє про типи або формати, які цей плагін підтримує. Після реєстрації інформація про плагін стає доступною для пошуку та використання.

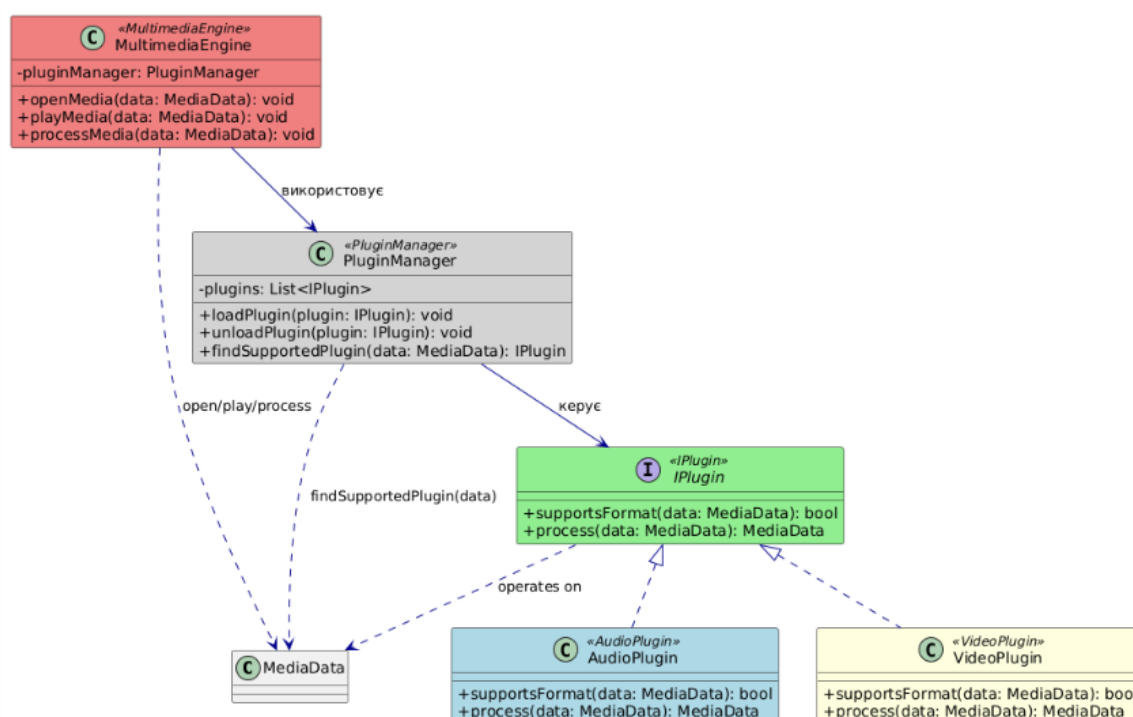


Рис. 4.2. Структурна діаграма запропонованого архітектурного шаблону проектування «Мультимедійний Плагін»

Далі, коли надходить мультимедійний контент, ядро (MultimediaEngine) викликає метод менеджера плагінів для визначення, чи є плагін, який підтримує вказаний формат або тип обробки. Якщо підходящий модуль знайдено, відбувається виклик його методу `process`, у

межах якого виконується декодування, аналіз чи інша спеціалізована дія над даними.

Після завершення обробки ядро може надати користувачеві результат або передати його іншій підсистемі. Якщо формат чи оброблення не підтримуються жодним із плагінів, ядро повідомляє про відсутність сумісного модуля.

Вивантаження та оновлення плагінів відбувається за запитом. Менеджер плагінів перестає пропонувати вилучений модуль у списку доступних, і ядро не буде викликати його методи. У разі потреби модуль можна замінити або оновити новішою версією, не зачіпаючи решту системи.

Implementation (Реалізація). Під час упровадження плагінної архітектури для мультимедіа варто передусім визначити базовий інтерфейс або абстрактний клас (наприклад, `IPlugin`), що уніфікує підхід до обробки різних форматів даних. Ядром системи зазвичай слугує окремий компонент (`MultimediaEngine`), який виконує ключові операції з мультимедійним контентом і делегує специфічні завдання плагінам. Менеджер плагінів (`PluginManager`) зберігає інформацію про всі доступні модулі, забезпечує їх завантаження чи вивантаження, а також обирає відповідний плагін залежно від типу або формату даних. Завдяки такій структурі можна додавати нові можливості, не змінюючи логіку ядра: достатньо підготувати окрему бібліотеку чи пакет із необхідним плагіном і зареєструвати його в системі. Поряд з цим можна організувати динамічне завантаження, що дозволить увімкнути або вимкнути конкретну обробку «на льоту». Приклад з реалізацією основних класів і взаємодії між ними розміщено у додатку 2.

Related Patterns (Пов'язані шаблони). Мікросервісна архітектура доповнює плагінний підхід у тих випадках, коли кожен плагін можна розгортати в окремому сервісі з власною інфраструктурою. `Factory Method` дає змогу автоматизувати створення об'єктів плагінів, особливо коли їх слід

завантажувати з динамічних бібліотек. Chain of Responsibility є подібним у сенсі послідовної обробки, але там зазвичай існує логіка передачі запиту далі або припинення ланцюга. Мультимедійний Пайплайн добре інтегрується з плагінною моделлю, коли кожен плагін можна підключати як окремий етап обробки у пайплайні.

Consequences (Наслідки). Плагінна модель спрощує адаптацію системи до нових форматів і технологій, адже форматоспецифічна логіка зберігається окремо від ядра. Це підвищує розширюваність і прискорює доставку оновлень, проте може ускладнити керування сумісністю та версіями: що більше плагінів, то уважніше потрібно відстежувати конфлікти. Окремою проблемою стає контроль якості: якщо плагіни розробляються сторонніми командами, ядро не завжди гарантує стабільну роботу кожного модуля. Крім цього, діагностика помилок може ускладнитися, бо збій у плагіні іноді впливає на загальний перебіг процесу. Проте гнучкість, можливість швидкої інтеграції нових функціональних блоків та ізольованість форматоспецифічного коду зазвичай переважають ризики, особливо в середовищах із часто змінюваними вимогами та динамічною специфікацією мультимедійних даних.

Шаблон «Мультимедійний Шлюз»

Intent (Намір). Мультимедійний шлюз (Media Gateway) передбачає створення єдиної точки входу для мультимедійних потоків і даних, яка приймає та маршрутизує інформацію з різних джерел у внутрішні сервіси чи підсистеми. Метою є спростити інтеграцію з зовнішніми пристроями, сторонніми платформами та різноманітними форматами, а також централізувати політики безпеки, конверсію форматів і контроль за навантаженням.

Motivation (Мотивація). У розподілених мультимедійних системах джерела інформації можуть бути доволі різноманітними за форматом,

протоколом передачі, регіональними особливостями. Сервіси, що відповідають за обробку, зберігання або трансляцію, нерідко розробляються окремо і потребують єдиного узгодженого інтерфейсу взаємодії. Шлюз слугує проміжною ланкою, через яку до системи надходять аудіопотоки, відеопотоки, метадані та інша супровідна інформація. Система «шлюзу» може здійснювати первинну обробку або фільтрацію даних, регулювати навантаження і розподіляти потоки по мікросервісах чи підсистемах. Завдяки цьому відокремлюється логіка інтеграції від логіки внутрішньої обробки, а також спрощується забезпечення безпеки, оскільки зовнішні підключення контролюються в єдиному місці. Це важливо і для масштабних проєктів, які мають декілька точок входу, і для менших систем із різноманітними типами пристроїв, бо дозволяє централізовано управляти форматами, протоколами і користувацькими правами доступу. Водночас шлюз стає тим шаром, де можна вбудувати механізми логування та моніторингу, відстежувати помилки та збої в мережевій частині. Така гнучкість дає можливість швидко вмикати чи вимикати нові канали і спрощує подальше розширення системи.

Applicability (Застосовність). Такий шаблон застосовують, коли важливо мати єдиний точний контроль доступу для різних потоків та джерел, а також спростити конфігурацію зовнішніх інтеграцій. У складних архітектурах на кшталт відеоспостереження, поточкових медичних систем, онлайн-конференцій чи великих відеоплатформ шлюз виконує роль фасаду, відділяючи внутрішню інфраструктуру від зовнішніх клієнтів. Він особливо доречний, коли потрібно підтримувати різні протоколи (RTMP, WebRTC, RTSP, SRT) і зводити їх до єдиного формату, керувати авторизацією та застосовувати загальні політики безпеки. У невеликих застосунках це може бути корисно при поступовому нарощуванні функціоналу, коли є ризик неоднорідності інтерфейсів чи виникнення конфліктів між компонентами.

Structure (Структура). Структура патерну полягає в існуванні спеціального компонента Gateway, який приймає мультимедійні дані з різних джерел і протоколів, перетворює їх або виконує первинну обробку, а потім передає у внутрішні сервіси. Gateway працює разом із менеджером протоколів чи перетворень та забезпечує логування, авторизацію й інші крос-системні сервіси. Структурна UML-діаграма запропонованого архітектурного шаблону зображена на рис. 4.3. Нижче більш детально розпишемо структурні частини пропонуваного шаблону.

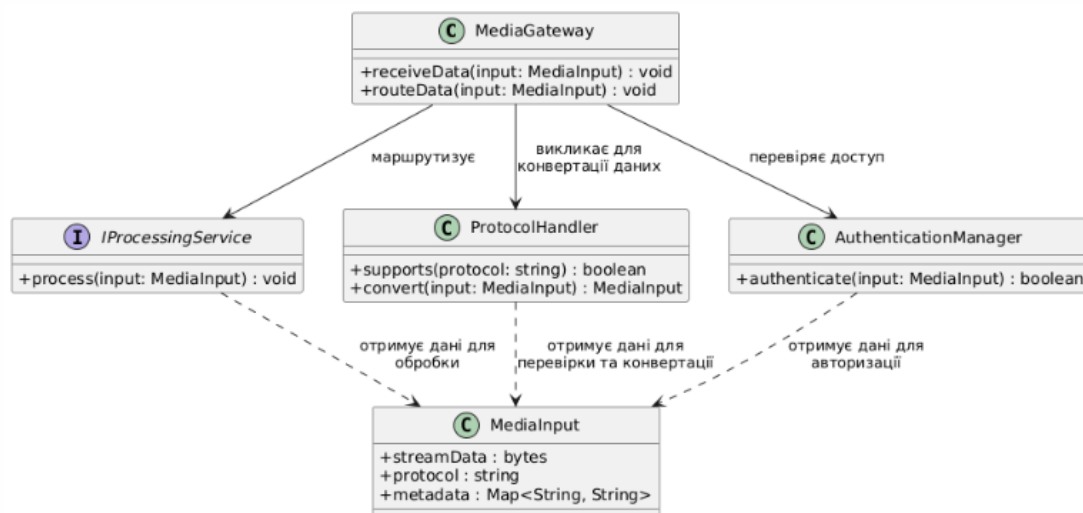


Рис. 4.3. Структурна діаграма запропонованого архітектурного шаблону проєктування «Мультимедійний Шлюз»

- **MediaInput.** Це сутність, яка містить бінарні або потокові дані мультимедіа, а також метадані та інформацію про протокол, через який дані надходять. Завдяки цьому об’єкту шлюз має можливість аналізувати тип і формат вхідних даних, визначати, які кроки обробки чи конвертації слід виконати, а також коректно маршрутизувати дані далі. MediaInput часто містить додаткові поля для відстеження

джерела (наприклад, ідентифікатор камери або пристрою), часових міток та інших контекстних параметрів.

- **MediaGateway.** Центральний компонент патерну, який отримує мультимедійні потоки чи файли з різних джерел і визначає, як саме з ними взаємодіяти. По-перше, він може звернутися до **AuthenticationManager** для перевірки прав доступу та безпеки. По-друге, **MediaGateway** викликає **ProtocolHandler**, щоб привести дані до уніфікованого формату чи протоколу. Насамкінець, він передає результат відповідному внутрішньому сервісу, що реалізує **IProcessingService**. Цей модуль відповідає і за контроль пропускну здатності, базове логування чи фільтрацію, а також за випадки помилок, коли відбувається сповіщення чи відхилення запиту.
- **ProtocolHandler.** Модуль, що спеціалізується на роботі з протоколами передачі даних, а також на початковій конвертації мультимедійних форматів. Його завдання — перевірити, чи підтримується конкретний протокол (RTSP, RTMP, SRT тощо) та виконати базову або глибшу адаптацію даних (наприклад, привести заголовки до необхідних стандартів чи перекодувати потік). Якщо протокол чи формат не підтримується, **ProtocolHandler** повертає відповідь про неможливість опрацювати вхідні дані, і далі **MediaGateway** може повідомити про помилку. У великих системах може існувати декілька реалізацій **ProtocolHandler**, об'єднаних у певну фабрику або плагінний механізм.
- **AuthenticationManager.** Компонент, що відповідає за перевірку автентифікації та, за необхідності, авторизацію запитів. Він обробляє метадані або додаткову інформацію у **MediaInput** (наприклад, маркер доступу, ключ авторизації, електронний підпис тощо) та визначає, чи має клієнт право надсилати або переглядати конкретні потоки. Якщо перевірка не проходить, шлюз може негайно відхилити запит. Крім простого порівняння токенів, цей модуль може містити складніші

механізми безпеки: двофакторну авторизацію, перевірку IP-адрес чи інтеграцію зі зовнішньою системою керування доступом.

- `IProcessingService`. Інтерфейс або абстрактний опис сервісів, здатних обробляти мультимедійні дані після їхньої первинної обробки у шлюзі. Кожен сервіс реалізує метод `process`, який отримує вже «приведені» чи уніфіковані дані. Реальні реалізації можуть охоплювати відеоаналітику, трансляцію користувачам, зберігання у сховище чи подальше накладання фільтрів. За рахунок спільного інтерфейсу Gateway має змогу не прив'язуватись до конкретної логіки обробки, а просто викликати відповідний сервіс на підставі метаданих чи типу даних.

MediaGateway приймає потік чи файл і визначає потрібного ProtocolHandler за типом протоколу або медіа. Якщо дані вимагають автентифікації, виконується запит до AuthenticationManager. Після успішної авторизації викликається метод `convert`, щоб отримати уніфікований формат, а потім MediaGateway передає його відповідному сервісу, реалізованому IProcessingService. Сервіс може бути мікросервісом, що опрацьовує відео чи аудіо, а шлюз виконує завдання маршрутизації й безпеки.

Implementation (Реалізація). Під час реалізації спочатку створюється компонент, який виконує роль шлюзу. Він слухає зовнішні підключення через потрібні протоколи (TCP, WebSockets, HTTP, RTMP чи інші), зчитує вхідні дані і визначає, який ProtocolHandler слід застосувати. В ProtocolHandler можна реалізувати необхідне декодування або перетворення, а також запровадити механізми визначення, чи підтримується конкретний формат. AuthenticationManager можна зберігати в ядрі шлюзу, щоб забезпечити єдину точку керування правами доступу й контролем безпеки. Далі, після валідації доступу й успішної конверсії даних, шлюз надсилає адаптований потік чи повідомлення внутрішнім

сервісам. Ці сервіси можна розробити як окремі мікросервіси чи модулі, кожен із яких відповідає за подальшу обробку або збереження даних. Приклад з реалізацією основних класів і взаємодії між ними розміщено у додатку 3.

Related Patterns (Пов'язані шаблони). Поняття «мультимедійний шлюз» тісно переплітається з концепцією API Gateway у мікросервісній архітектурі, де шлюз виступає єдиною точкою доступу до безлічі внутрішніх сервісів, забезпечуючи балансування навантаження, кешування та автентифікацію. У більш загальному плані можна провести паралель із шаблоном Facade (Фасад), що, як і шлюз, спрощує складну внутрішню структуру для зовнішніх клієнтів, але при цьому часто обмежується єдиним інтерфейсом без урахування специфіки протоколів. Якщо ж система вимагає адаптації різноманітних форматів чи протоколів, то проглядається схожість із Adapter (Адаптер), де кожен обробник може виступати адаптером для перетворення даних під потрібний формат. У випадках, коли потрібна послідовна оброблення або перевірка відповідності різним умовам, доречним буде посилання на Chain of Responsibility, яка дозволяє проходити ланцюгом обробників (хоча у «мультимедійному шлюзі» ключову роль грає централізований контроль, а не передавання відповідальності «далі»). Іноді інтегрують Observer (Спостерігач) або Event-Driven підхід, особливо коли шлюз потребує надсилати повідомлення про нові дані багатьом зацікавленим підсистемам і відстежувати події в реальному часі. Якщо всередині шлюзу чи після нього мультимедійні дані проходять крізь серію фільтрів, можна поєднати його з патерном Multimedia Pipeline, адже шлюз відповідає за маршрутизацію потоків, а конвеєр фільтрів — за покрокову обробку. Усе це демонструє, що «Media Gateway» є доволі універсальним будівельним блоком, котрий можна з успіхом об'єднати з іншими шаблонами, залежно від потреби в розподіленості, розширюваності або специфіки мережевої взаємодії.

Consequences (Наслідки). Використання мультимедійного шлюзу централізує роботу з вхідними потоками, що спрощує керування безпекою, автентифікацією та уніфікацією форматів. Це робить архітектуру більш гнучкою: нові протоколи або зовнішні пристрої можна підключати, модифікувавши лише невеликий компонент або використавши плагінну модель. З іншого боку, шлюз стає одним із найбільш навантажених елементів системи, де накопичуються виклики, пов'язані з продуктивністю, відмовостійкістю та захистом від атак. Якщо в системі планується різке збільшення обсягу потоків, доведеться розглянути можливість масштабування самого шлюзу, наприклад за рахунок використання балансувальників навантаження або контейнеризації та оркестрації. Хоча запровадження єдиного входу суттєво спрощує узгодження з іншими сервісами, воно також робить цю точку критичною для функціонування, що вимагає додаткових інструментів моніторингу та резервування.

4.2. Узагальнена архітектура програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів

Розвиток технологій цифрових двійників у медицині потребує архітектури, здатної забезпечувати високу точність і достовірність даних, масштабованість і адаптивність, інтеграцію з клінічними процесами, безперервне оновлення в реальному часі, а також відповідний рівень безпеки та конфіденційності. Зокрема, для цифрових двійників медико-біологічних об'єктів (МБО), таких як гортань, важливим є уніфікований підхід до збирання, обробки та аналізу різномірних медичних даних (відеоендоскопія, аудіозаписи голосу, показники медичних сенсорів, томограми, електронні медичні записи). Метою даного дослідження є розроблення архітектурних принципів, що забезпечать високу якість створення цифрових двійників медико-біологічних об'єктів, зокрема гортані людини. Сформулюємо основні нефункціональні вимоги до такої системи з

посиланням на досліджені раніше у першому розділі та описані у Таблиці 1.2 функціональні вимоги (ФВ).

Точність та уніфікація медичних даних. Архітектура передбачає збір та нормалізацію вхідної інформації з медичних приладів, інформаційних систем та документів, використовуючи єдині формати зберігання та обміну. Це підвищує достовірність цифрового двійника й дозволяє коректно реалізувати функціональні вимоги, пов'язані з імпортом даних (ФВ1) та інтеграцією з електронними медичними картками (ФВ3). Крім того, можливість експорту у різні формати (ФВ10) і взаємодія з хмарними сервісами (ФВ11) також покладаються на єдиний підхід до уніфікації даних.

Масштабованість та адаптивність. Система має легко розширюватися зі збільшенням кількості пацієнтів та типів оброблюваних сигналів. Це особливо актуально для моделювання різноманітних органів чи складніших фізіологічних процесів, таких як вплив лікарських препаратів (ФВ15) або симуляція в реальному часі (ФВ4). Адаптивність забезпечується модульною структурою та гнучкими механізмами обробки даних, що дозволяє налаштовувати моделі під індивідуальні особливості пацієнта (ФВ7) і автоматично оновлювати їх за надходження нових даних (ФВ13).

Інтеграція з клінічними процесами. Для ефективного впровадження у медичну практику цифровий двійник має бути сумісним із внутрішніми лікарняними системами (LIS, PACS, RIS) та підтримувати уніфіковані стандарти обміну (HL7, FHIR). Це дає змогу в повній мірі реалізувати функціональність телемедицини та віддаленого моніторингу пацієнтів (ФВ16), а також інтегруватися з науковими базами даних (ФВ17) для актуалізації моделей.

Безперервність оновлень у реальному часі. Цифровий двійник має оперативно реагувати на зміни в стані пацієнта, особливо у відділеннях

інтенсивної терапії чи за критичних станів. Таким чином, реалізуються функції симуляції фізіологічних процесів (ФВ4) та моделювання перебігу захворювань (ФВ6), що можуть бути розраховані в реальному часі для ухвалення клінічних рішень.

Безпека та конфіденційність. Обробка персональних медичних даних вимагає дотримання жорстких вимог до збереження та передачі інформації. Система має забезпечувати шифрування, багаторівневу аутентифікацію, анонімізацію (псевдонімізацію) та аудит доступу. Це є критичним компонентом усіх етапів роботи із системою і стосується як взаємодії з клінічною інформацією, так і з результатами моделювання.

Модульність та ізолюваність компонентів. Кожна функціональна частина системи виконує окремі завдання, що спрощує розробку, тестування та оновлення. Використання підходів на кшталт мікросервісів чи контейнеризації сприяє стабільній роботі й підвищує надійність. Модульна організація також спрощує інтеграцію з технологіями доповненої та віртуальної реальності (ФВ14), підтримку багатомовного інтерфейсу (ФВ9) та доступ до системи з мобільних пристроїв (ФВ12).

Тепер на основі комбінації вимог наведено розроблені ключові компоненти пропонованої архітектури та приклади того, які саме функціональні вимоги (ФВ) вони реалізують.

Модуль автентифікації та управління користувачами. Забезпечує вхід у систему пацієнтів, лікарів та адміністраторів з використанням облікових записів, багатоетапну перевірку доступу та налаштування прав. Зберігає інформацію про користувачів і контекстні сесії, а також виконує аудит звернень. Хоча цей модуль напряду не відповідає за конкретні функціональні вимоги з табл. 1.2, він створює основу для безпечного використання усіх інших функцій системи.

Модуль інтеграції та уніфікації даних. Збирає та нормалізує дані з різних джерел: відеоендоскопія, аудіозаписи голосу, медичні прилади

(сенсори тиску, респіраторні показники тощо), електронні медичні документи. Конвертує різнорідні дані у єдиний системний формат та записує їх до центрального сховища. Цей компонент реалізує ФВ1 (імпорт даних із приладів), ФВ3 (інтеграція з електронними картками), ФВ10 (експорт у різні формати), ФВ11 (взаємодія з хмарними сервісами) та частково ФВ17 (інтеграція з науковими базами).

Модуль обробки медичних даних. Виконує фільтрацію шумів, нормалізацію сигналів, сегментацію, екстракцію ознак та попередній аналіз за допомогою нейронних мереж (наприклад, 3D-CNN для відео, рекурентних мереж для аудіо, CNN для статичних зображень). Оброблені дані надсилаються до сховища для подальшого використання, зокрема для автоматичного оновлення моделей (ФВ13) та аналітичних інструментів (ФВ6). Модуль також закладає основу для симуляції фізіологічних процесів у реальному часі (ФВ4).

Модуль створення цифрового двійника. На основі оброблених даних формує цифровий двійник гортані чи іншого органу. Проводить аналіз відео, аудіо та додаткової клінічної інформації, прогнозує фізіологічні зміни, параметри деформацій та функціональні характеристики. Забезпечує 3D-моделювання (ФВ2), симуляцію в реальному часі (ФВ4), налаштування моделей під індивідуальні особливості (ФВ7) і моделювання впливу лікарських препаратів (ФВ15). Цей же модуль може бути розширений для інтеграції з VR/AR-технологіями (ФВ14).

Модуль візуалізації та аналітики. Надає графічне відображення цифрового двійника, зокрема у 3D-форматі, і дає змогу лікарям аналізувати зміни в динаміці, планувати лікування, прогнозувати наслідки втручань. Дозволяє переглядати результати моделювання (ФВ5), використовувати аналітичні інструменти (ФВ6), формувати звіти й документацію (ФВ8), а за потреби розгортати віртуальні або доповнені середовища (ФВ14). Підтримка багатомовності інтерфейсу (ФВ9) та доступ із мобільних

пристроїв (ФВ12) також реалізуються цим компонентом за рахунок відповідних інтерфейсних рішень.

Модуль безпеки та анонізації. Реалізує анонімізацію (псевдонімізацію) персональних даних, шифрування під час передачі даних мережею, моніторинг та блокування несанкціонованого доступу. Завдяки розподіленому зберіганню й мінімізації обсягу ідентифікаційних даних у разі компрометації одного з компонентів уся система лишається захищеною. Безпека є наскрізною вимогою до всієї архітектури та критично важливою при роботі з персональними і клінічними відомостями.

Система контролю якості та тестування (CI/CD). Автоматизує процеси тестування, виявлення помилок та внесення виправлень, використовуючи модульне, інтеграційне та регресійне тестування. Безперервна інтеграція (CI) і доставка (CD) гарантують стабільність та відповідність системи вимогам, включно з підтриманням необхідного рівня покриття тестами. Хоча цей компонент безпосередньо не відображений у функціональних вимогах з табл. 1.2, він забезпечує високу якість і надійність реалізації таких вимог, як симуляції в реальному часі (ФВ4) чи аналітичні інструменти (ФВ6).

Таким чином, запропонована архітектура поєднує принципи модульності, уніфікації та інтелектуальної обробки даних, одночасно враховуючи вимоги безпеки, масштабованості й адаптивності. Усі її компоненти можуть працювати незалежно і взаємодіяти один з одним через стандартизовані інтерфейси. Це дає змогу задовольнити як нефункціональні вимоги (точність, безпека, безперервність оновлень, інтеграція з клінічними системами тощо), так і широкий спектр функціональних вимог (ФВ1–ФВ17), включно з імпортом та експортом даних, 3D-моделюванням, симуляцією фізіологічних процесів, багатомовним та мобільним доступом, а також телемедициною й інтеграцією з науковими базами даних. Спільне використання зазначених модулів забезпечує високоякісне створення

цифрових двійників медико-біологічних об'єктів та ефективного застосування їх у клінічній практиці.

Для того, аби всі модулі могли послідовно взаємодіяти між собою й обмінюватися результатами без втрати точності та актуальності, необхідно забезпечити єдиний підхід до представлення медичних даних. Одним із ключових факторів успішної реалізації цифрових двійників є впровадження уніфікованих форматів і стандартів, що дають змогу адекватно поєднувати вхідну інформацію з різних джерел, зберігати її в централізованому сховищі та швидко передавати між компонентами системи. Це дозволяє мінімізувати дублювання даних, запобігти можливим конфліктам при обробці та, як наслідок, гарантувати безперервність та оперативність оновлення цифрових моделей у реальному часі.

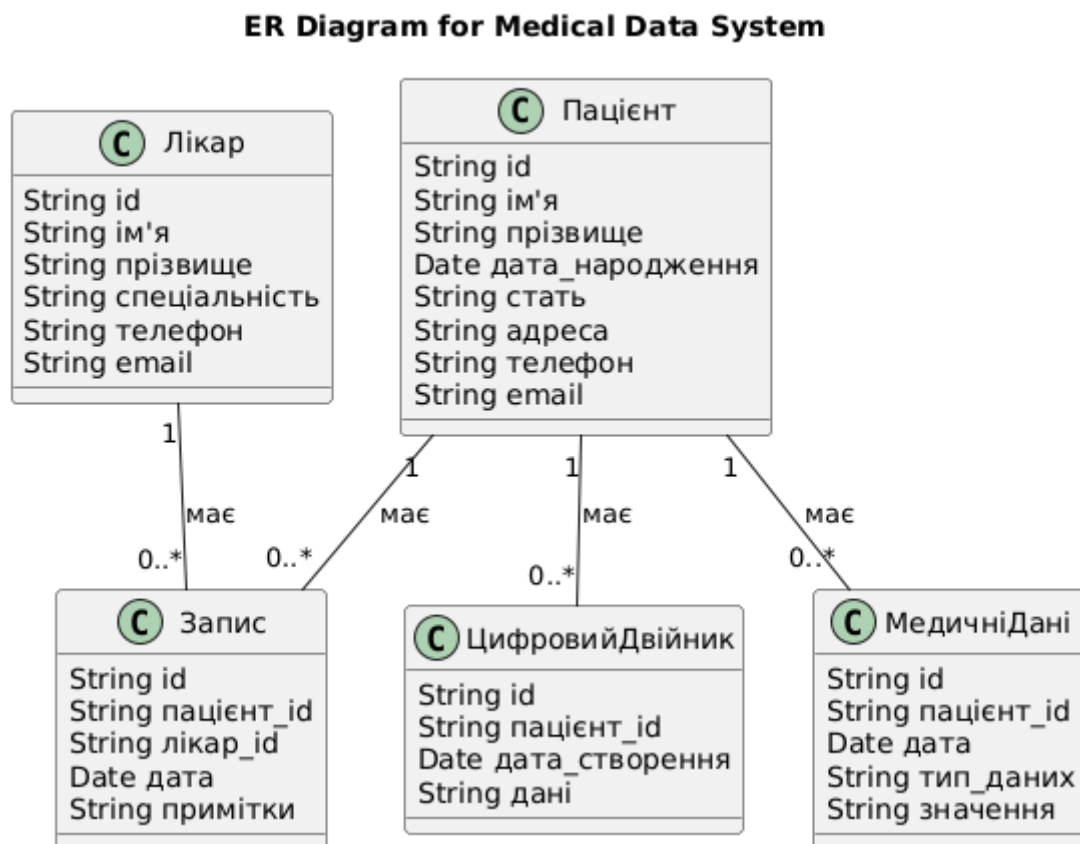


Рис. 4.4. Пропонований шаблон бази для збереження даних (ER діаграма)

Отже уніфікований формат медичних даних і їх стандартизація є ключовим аспектом запропонованої архітектури. На рис. 4.4. наведено приблизну структуру бази даних, яка включає сутності «Лікар», «Пацієнт», «Запис» (медичний запис), «Цифровий двійник» та «Медичні дані». Така модель спрощує пошук, оновлення, аналіз та інтеграцію даних. Подібна структура забезпечує логічну впорядкованість, уніфікацію та спрощення операцій над даними. Завдяки ній можна легко інтегрувати нові типи даних або алгоритми їх обробки.

Тепер маючи основні модулі зобразимо розроблену *узагальнену архітектуру програмної системи для створення і використання цифрових двійників медико-біологічних об'єктів*. Її можна переглянути на рис. 4.6. Вона на модульному підході та мікросервісній концепції. Кожна функціональна частина виконується як окремий сервіс, а міжсервісна взаємодія реалізується через централізовану інфраструктуру обміну даними. Такий підхід дає змогу:

- Спрощувати інтеграцію та адаптацію завдяки автономності модулів.
- Замінювати чи оновлювати окремі сервіси без потреби внесення змін до решти системи.
- Забезпечувати високу надійність і відмовостійкість за рахунок швидкого виявлення й усунення несправностей у межах конкретного модуля.
- Легко масштабувати систему, додаючи нові екземпляри сервісів при зростанні навантаження.

Коротко розпишемо послідовність взаємодії пропонованих модулів розробленої архітектури. Графічну реалізацію у вигляді діаграми активності системи, що ілюструє послідовність обробки даних і взаємодію між модулями наведено на рис. 4.5.

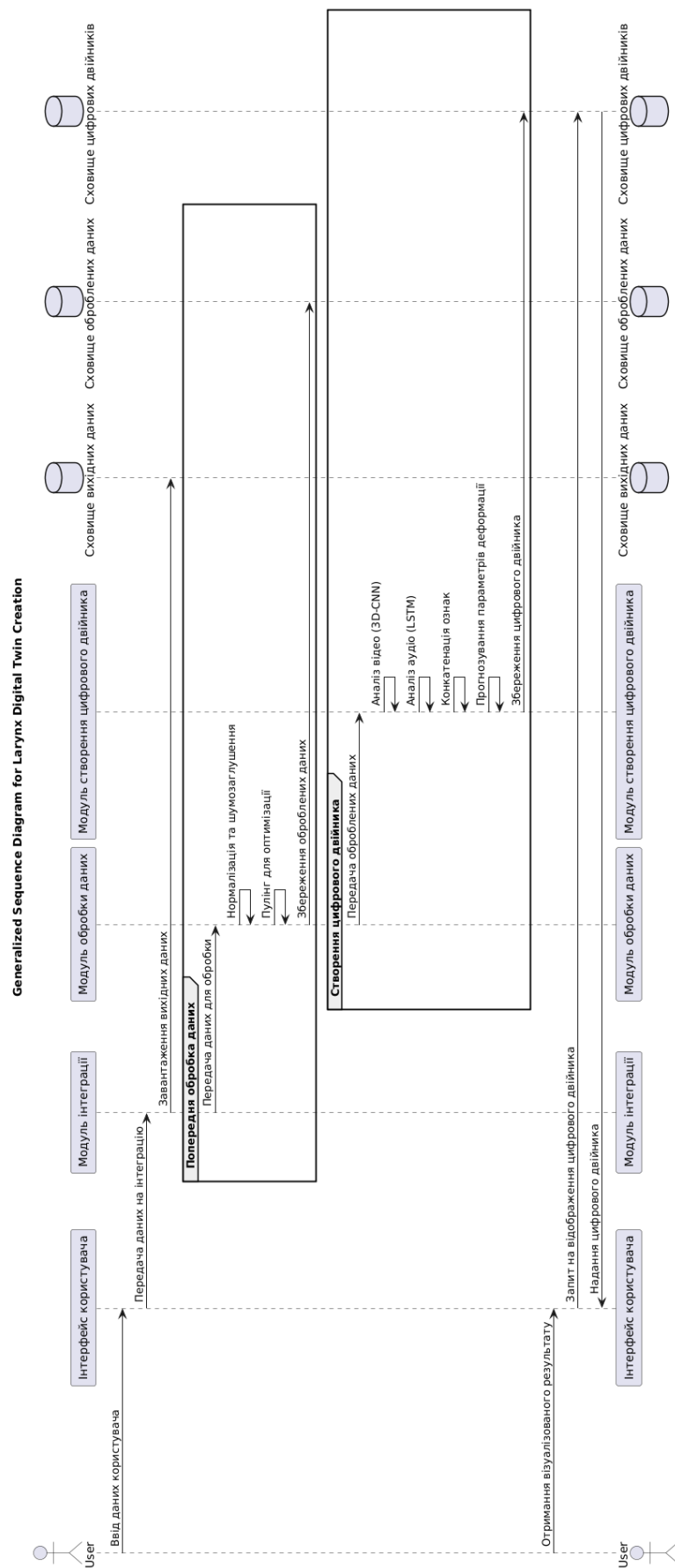


Рис. 4.5. Діаграма активності пропонованої системи

Модуль обробки даних відповідає за первинне перетворення, очистку та уніфікацію інформації, що надходить від сторонніх пристроїв (ендоскопічних камер, аудіореєстраторів, сенсорів тощо). Далі модуль «цифрових двійників» взаємодіє з уже обробленими вхідними даними та формує віртуальні представлення конкретного медико-біологічного об'єкта (МБО). Окремий блок «управління й користування» реалізує створення облікових записів, надає інструменти призначення аналізів і доступу до історії пацієнтів. Важливу роль відіграє підсистема безпеки, що слідкує за автентифікацією та правами користувачів, а також система автоматизованого тестування, яка контролює якість роботи всіх мікросервісів.

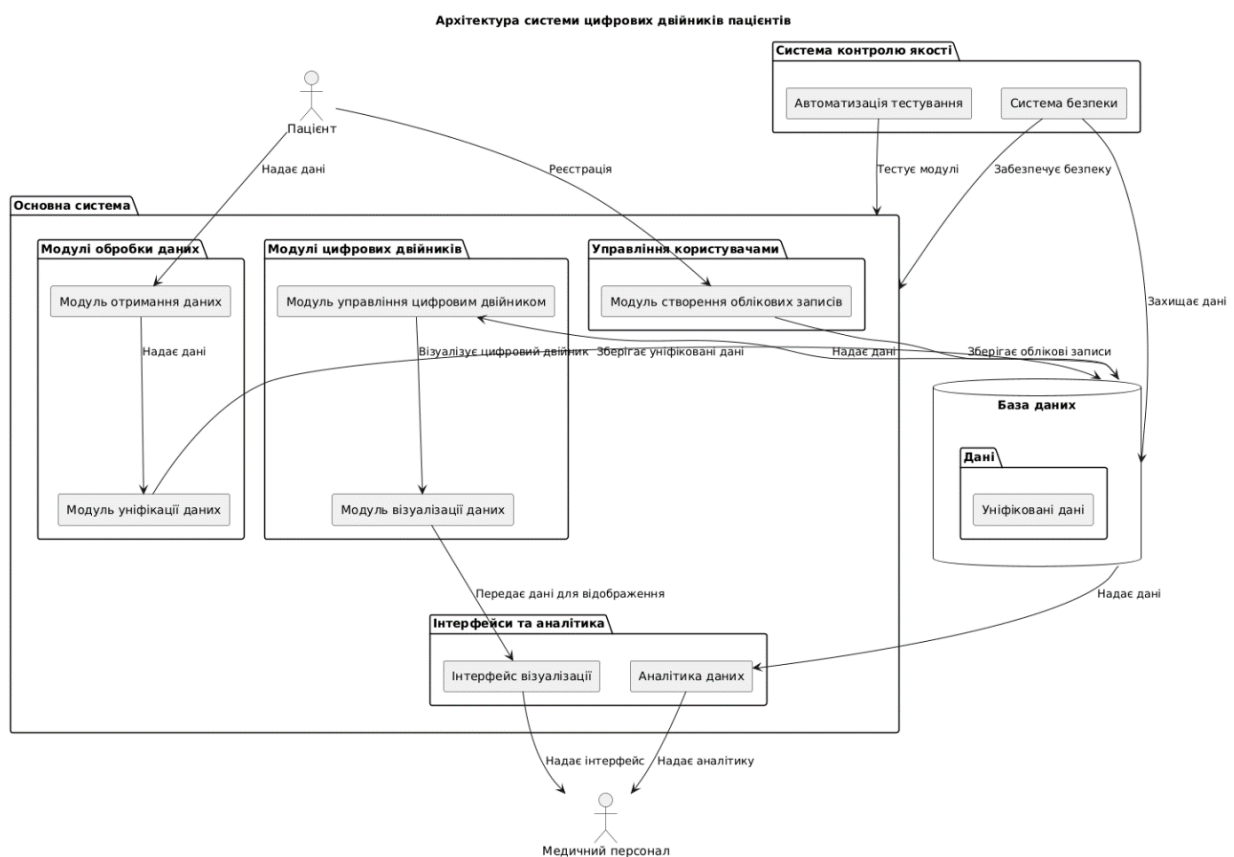


Рис. 4.6 Архітектура розроблюваної системи

Завдяки такому модульному устрою зручно впроваджувати **архітектурні патерни**, що були розроблені та запропоновані раніше у цьому розділі. Для покрокової обробки вхідних мультимедійних потоків доцільно застосувати шаблон **«Мультимедійний Конвеєр»**, де джерела даних (Source) послідовно передають інформацію через низку фільтрів (IFilter), а результат формування цифрового двійника надходить у кінцеву точку (Sink). Це знижує складність системи, бо кожен фільтр виконує лише одну чітко визначену функцію (наприклад, видалення шумів чи конвертацію формату), а зміни в одному етапі не впливають на інші. Якщо ж до системи час від часу потрібно «підключати» нові типи даних або спеціалізовані алгоритми обробки, ефективним стає **«Мультимедійний Плагін»**.

У такій моделі ядро залишається незмінним, а підтримка додаткових форматів реалізується у вигляді незалежних модулів. Це особливо актуально для системи цифрових двійників, де можуть виникати нові методи 3D-візуалізації чи специфічні дані різної природи (наприклад, ультразвукові знімки). Крім того, у великих розподілених сценаріях, коли дані надходять одночасно з багатьох пристроїв (ендоскопи, мікрофони, моніторингові сенсори тощо), зручно використовувати підхід **«Мультимедійного Шлюзу»**, що виконує роль єдиної точки входу. Такий шлюз приймає потоки з різних джерел, виконує первинну обробку (перевірку прав доступу, конвертацію протоколів) і лише потім розподіляє дані далі по спеціалізованих сервісах для моделювання та візуалізації.

4.3. Інтерфейси та взаємодія з користувачами

Для підвищення зручності роботи медичного персоналу архітектура передбачає інтуїтивно зрозумілі інтерфейси візуалізації. ***Графічні інструменти та панелі керування*** забезпечують швидкий доступ до ключових параметрів двійника; лікар може легко досліджувати анатомічні

деталі, функціональні параметри, переглядати динаміку змін у часі, порівнювати стан до та після лікувальних процедур. А **інтерактивні 3D-моделі** дозволяють вивчати структуру гортані, оцінювати вплив різних факторів на функціонування голосового апарату та тестувати різні сценарії лікування.

У медичній програмній системі, що заснована на технології цифрових двійників, взаємодіють три основні категорії користувачів: пацієнти, лікарі та адміністратори, а також сторонні пристрої (наприклад, діагностичні прилади, датчики тощо).

Пацієнти мають змогу отримувати інформацію про свій стан, переглядати результати досліджень, наданих лікарями, а також в перспективі ознайомлюватися з рекомендаціями. Пацієнти можуть вводити окремі дані (наприклад, відповіді на анкети про самопочуття) через інтерфейс користувача. Доступ пацієнта до системи обмежений переглядом власної інформації та результатів аналізів.

Лікарі забезпечують введення та контроль якості медичних даних, призначають аналізи, формують цифрові двійники пацієнтів. Лікарі використовують інструменти візуалізації для деталізованого дослідження стану гортані та інших МБО, можуть моделювати вплив різних лікувальних заходів. Їх доступ розширений: вони можуть не лише переглядати історію пацієнтів, а й створювати та оновлювати цифрові двійники, призначати обстеження, отримувати аналітичні звіти.

Адміністратори відповідають за налаштування системи, управління правами доступу користувачів, оновлення програмного забезпечення, контроль політик безпеки, інтеграцію з іншими інформаційними системами закладу. Адміністратори взаємодіють з інтерфейсом керування для конфігурування системи та моніторингу логів безпеки, виконання аудиту.

Сторонні пристрої автоматично збирають та передають медичні дані в систему. Це можуть бути ендоскопічні камери, аудіореєстратори для

аналізу голосових сигналів, датчики життєвих показників. Дані з таких пристроїв надходять у модуль інтеграції та обробки, після чого використовуються для оновлення цифрових двійників.

На рис. 4.7 наведено діаграму випадків використання (Use Case Diagram), що відображає типи взаємодій різних акторів із системою.

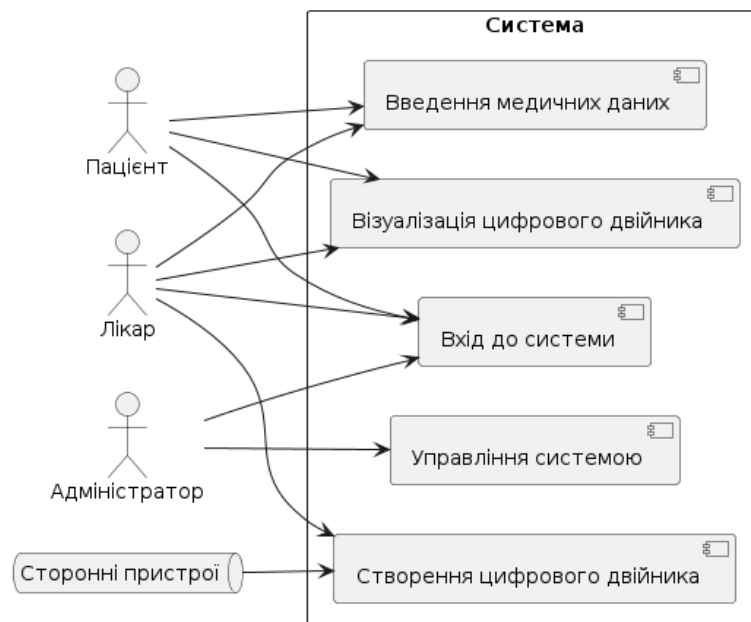


Рис. 4.7. Діаграма використання пропонованої системи

Як видно з діаграми, пацієнти та лікарі можуть вводити медичні дані (UC1), переглядати цифрові двійники (UC2) та входити до системи (UC3). Лікарі мають можливість створювати цифрові двійники пацієнтів (UC5). Адміністратори здійснюють управління системою (UC4), а сторонні пристрої забезпечують автоматичне надходження даних, необхідних для створення та оновлення двійників (UC5).

4.4. Особливості впровадження

При впровадженні запропонованої архітектури важливо врахувати низку ключових аспектів, що сприяють сумісності з існуючими системами, ефективному налаштуванню виробничого середовища, високому рівню

кваліфікації персоналу та постійному моніторингу роботи. Інтеграція з медичними інформаційними ресурсами здійснюється через використання загальновизнаних стандартів HL7 і FHIR, що забезпечує органічну взаємодію з електронними медичними картами, лабораторними інформаційними системами, радіологічними сервісами та іншими додатками шляхом стандартизованого обміну даними.

Сучасні підходи до забезпечення якості програмного забезпечення орієнтовані на автоматизацію тестування, безперервну інтеграцію та доставку, а також збір метрик з логуванням. Автоматизація тестування полягає у створенні автотестів для кожного функціонального модуля, що дозволяє оперативно виявляти помилки на ранніх стадіях розроблення та зменшувати ризик людської помилки, адже при внесенні змін до коду чи архітектури тести запускаються автоматично. Використання інструментів CI/CD, таких як Jenkins, GitLab CI/CD або GitHub Actions, гарантує стабільне збирання, розгортання та оновлення сервісів без зупинки роботи системи, що є критично важливим для медичних застосунків, де безперервний доступ до даних є пріоритетним. Регулярний аналіз показників покриття тестами, часу відгуку, затримок та надійності мікросервісів дозволяє вчасно виявляти потенційні недоліки та оперативно реагувати на них.

Налаштування виробничого середовища базується на впровадженні технологій контейнеризації, зокрема Docker і Kubernetes, що сприяє гнучкому розгортанню, оперативному оновленню та ефективному управлінню компонентами системи. Такий підхід значно прискорює процеси інтеграції та доставки, що є необхідним для своєчасного впровадження нових функціональних можливостей і підтримання високої продуктивності.

Підготовка медичного персоналу є ключовим чинником успішної експлуатації архітектури, оскільки ефективне використання інструментів

для візуалізації та інтерпретації даних цифрових двійників, а також управління процесами діагностики і лікування вимагає спеціалізованих знань і навичок. Організоване навчання дозволяє персоналу повноцінно засвоїти технологічні рішення, що сприяє підвищенню якості медичного обслуговування.

Система моніторингу та аналітики, заснована на регулярному аналізі логів, звітів про використання системи та показників тестування, відіграє важливу роль у безперервному вдосконаленні архітектури. Такий підхід дозволяє своєчасно виявляти потенційні недоліки, вносити необхідні корективи та підтримувати відповідність системи сучасним вимогам медичної практики.

4.5. Висновки до розділу 4

У цьому розділі розроблена архітектура медичної програмної системи для створення цифрових двійників медико-біологічних об'єктів, яка базується на принципах модульності, уніфікації вхідних даних, масштабованості, безпеки та якості програмного забезпечення.

Розроблена архітектура:

- забезпечує стандартизацію та уніфікацію процесів збору даних із різноманітних джерел;
- гарантує безперервне оновлення цифрових двійників у реальному часі, що підвищує точність діагностики та ефективність лікування;
- використовує сучасні методи автоматизації тестування (CI/CD), що дозволяє оперативно виявляти та виправляти помилки;
- реалізує надійну систему безпеки та анонімізації, захищаючи конфіденційність особистих даних пацієнтів;
- надає інтуїтивно зрозумілі інтерфейси для візуалізації цифрових двійників та аналізу медичних даних, сприяючи підвищенню якості медичного обслуговування.

Практичне застосування цієї архітектури, продемонстроване на прикладі цифрового двійника гортані, підтверджує ефективність використання описаних підходів та інструментів у медичній галузі. Таким чином, розроблені архітектурні принципи сприяють підвищенню якості медичних послуг та оптимізації процесів обробки й аналізу біологічних даних, що зрештою веде до покращення здоров'я пацієнтів та ефективнішого використання ресурсів медичних установ.

ВИСНОВКИ

У дисертаційній роботі розв'язано науково-технічну задачу підвищення ефективності створення та використання цифрових двійників медико-біологічних об'єктів завдяки розробленню універсального алгоритмічного й програмного забезпечення для обробки мультимодальних темпоральних даних. Отримані результати мають важливе значення для підвищення точності діагностики, оптимізації медичних процедур та формування підґрунтя для персоналізованої медицини та освіти.

Основні наукові та практичні результати полягають у наступному.

1. **Виявлено** необхідність розвитку концепції цифрових двійників для медико-біологічних об'єктів із урахуванням людськоцентричної специфіки таких галузей, як медицина та освіта. Виконаний аналіз наявних систем засвідчив, що існуючі рішення орієнтовані на вузькоспеціалізовані сфери, мають обмежені можливості масштабування та не забезпечують комплексний підхід до інтеграції різноманітних джерел даних.
2. **Розроблено** універсальну архітектуру програмної системи, здатну ефективно опрацьовувати мультимодальні темпоральні дані медико-біологічних об'єктів. Запропонована архітектура підтримує збір, синхронізацію, семантичний аналіз, інтеграцію та візуалізацію даних із різноманітних джерел, забезпечуючи можливість створення повноцінних цифрових двійників, які оновлюються в реальному часі.
3. **Розроблено** методи синхронізації та семантичної інтеграції даних, що ґрунтуються на застосуванні семантичних графів, графових нейронних мереж та тривимірних згорткових нейронних мереж. Використання цих підходів дало змогу підвищити точність визначення зв'язків між мультимодальними сигналами (відео, аудіо, сенсори) та виявити

приховані закономірності, важливі для діагностики стану пацієнта або поведінки складної біосистеми.

4. **Розроблено** механізми адаптації базової 3D-моделі медико-біологічних об'єктів за допомогою глибинних нейронних мереж, що враховують індивідуальні характеристики та динаміку фізіологічних процесів. Застосування 3D-CNN та рекурентних архітектур забезпечило розпізнавання просторово-часових патернів і персоналізацію цифрових двійників відповідно до реальних даних про конкретного пацієнта.
5. **Запропоновано** формат представлення темпоральних мультимодальних даних та **розроблено** низку архітектурних шаблонів проєктування, які полегшують інтеграцію, обробку та повторне використання отриманих даних, а також спрощують розроблення програмних систем, орієнтованих на роботу з цифровими двійниками в медичній сфері.
6. **Удосконалено** теоретичні засади оброблення просторово-часових параметрів медико-біологічного об'єкта для побудови його цифрового двійника, у тому числі, за рахунок застосування тривимірних згорткових нейронних мереж та рекурентних архітектур для оброблення темпоральних мультимодальних медичних даних.

Таким чином, отримані в дисертаційній роботі результати формують комплексне теоретичне та практичне підґрунтя для створення високоефективного алгоритмічного та програмного забезпечення, яке підтримує інтеграцію мультимодальних темпоральних даних та масштабованість. Це відкриває перспективи для застосування розроблених підходів у медичних, освітніх та інших людиноцентризованих галузях, де точне моделювання, діагностика та прогнозування стану складних об'єктів мають вирішальне значення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

wska, A., 2024. Advanced computational methods for modeling, prediction and optimization—a review. *Materials*, 17(14), p.3521.

9. Liu, Y., Ilyas, M., Khosa, S.K., Muhmoudi, E., Ahmad, Z., Khan, D.M.

a

n

11. Шесчанський В.Ю., Сулема Є.С. Методи створення цифрових двійників медико-біологічних об'єктів на прикладі отоларингології.

«Системні технології» № 6 (149) за 2023 рік

a

m

e

d

a

n

i

,

G

.

G

.

,

2

hnology—Theory and Application: Stuttgart Conference on Automotive

ДОДАТКИ

Додаток 1. Реалізація запропонованого архітектурного шаблону проєктування «Мультимедійний Конвеєр»

```
class Data:
    def __init__(self, raw_bytes):
        self.raw = raw_bytes

class Source:
    def get_data(self):
        # У реальному випадку метод захоплення аудіопотоку з мікрофона
        # чи файлу
        raw_bytes = b"audio_stream_chunk"
        return Data(raw_bytes)

class IFilter:
    def process(self, data: Data) -> Data:
        raise NotImplementedError("This method should be
overridden.")

class NoiseReductionFilter(IFilter):
    def process(self, data: Data) -> Data:
        # Приклад "очищення" аудіосигналу
        print("NoiseReductionFilter: applying noise reduction")
        # У реальному випадку застосування алгоритму фільтрації шуму
        return data

class CompressionFilter(IFilter):
    def process(self, data: Data) -> Data:
        print("CompressionFilter: compressing data")
        # У реальному випадку стискання аудіо
```

```

        return data

class Sink:
    def output(self, data: Data):
        print("Sink: outputting or storing data")

class PipelineManager:
    def __init__(self):
        self.filters = []
        self.source = None
        self.sink = None

    def set_source(self, source: Source):
        self.source = source

    def set_sink(self, sink: Sink):
        self.sink = sink

    def add_filter(self,flt: IFilter):
        self.filters.append(flt)

    def start(self):
        if not self.source or not self.sink:
            raise RuntimeError("Source or Sink not configured.")
        # Умовно: отримаємо один "кадр" чи "порцію" даних
        data = self.source.get_data()
        for flt in self.filters:
            data = flt.process(data)
        self.sink.output(data)

# Приклад використання
if __name__ == "__main__":
    manager = PipelineManager()

```



```
manager.set_source(Source())  
manager.add_filter(NoiseReductionFilter())  
manager.add_filter(CompressionFilter())  
manager.set_sink(Sink())  
manager.start()
```

Додаток 2. Реалізація запропонованого архітектурного шаблону проєктування «Мультимедійний Плагін»

```
class MediaData:
    def __init__(self, data_bytes, format_name):
        self.data_bytes = data_bytes
        self.format_name = format_name

class IPlugin:
    def supportsFormat(self, media: MediaData) -> bool:
        raise NotImplementedError

    def process(self, media: MediaData) -> MediaData:
        raise NotImplementedError

class AudioPlugin(IPlugin):
    def supportsFormat(self, media: MediaData) -> bool:
        return media.format_name in ("mp3", "wav", "flac")

    def process(self, media: MediaData) -> MediaData:
        print(f"AudioPlugin: Processing {media.format_name}")
        # Тут може бути складна логіка декодування або обробки
        return media

class VideoPlugin(IPlugin):
    def supportsFormat(self, media: MediaData) -> bool:
        return media.format_name in ("mp4", "avi", "mkv")

    def process(self, media: MediaData) -> MediaData:
        print(f"VideoPlugin: Processing {media.format_name}")
        # Тут може бути логіка декодування відео, фільтрація тощо
        return media
```

```

class PluginManager:
    def __init__(self):
        self.plugins = []

    def loadPlugin(self, plugin: IPlugin):
        self.plugins.append(plugin)

    def unloadPlugin(self, plugin: IPlugin):
        self.plugins.remove(plugin)

    def findSupportedPlugin(self, media: MediaData) -> IPlugin:
        for plg in self.plugins:
            if plg.supportsFormat(media):
                return plg
        return None

class MultimediaEngine:
    def __init__(self, plugin_manager: PluginManager):
        self.plugin_manager = plugin_manager

    def openMedia(self, media: MediaData):
        plugin = self.plugin_manager.findSupportedPlugin(media)
        if not plugin:
            print(f"No plugin found for format {media.format_name}")
            return None
        return plugin.process(media)

# Приклад використання
if __name__ == "__main__":
    manager = PluginManager()
    manager.loadPlugin(AudioPlugin())
    manager.loadPlugin(VideoPlugin())

```

```
engine = MultimediaEngine(manager)

audio_data = MediaData(b"some audio bytes", "mp3")
engine.openMedia(audio_data)

video_data = MediaData(b"some video bytes", "avi")
engine.openMedia(video_data)

unknown_data = MediaData(b"unknown bytes", "xyz")
engine.openMedia(unknown_data)
```

Додаток 3. Реалізація запропонованого архітектурного шаблону проєктування «Мультимедійний Шлюз»

```
class MediaInput:
    def __init__(self, stream_data: bytes, protocol: str, metadata:
dict):
        self.stream_data = stream_data
        self.protocol = protocol
        self.metadata = metadata

class ProtocolHandler:
    def supports(self, protocol: str) -> bool:
        return protocol in ["RTSP", "RTMP", "SRT"]

    def convert(self, input_data: MediaInput) -> MediaInput:
        print(f"ProtocolHandler:          converting          from
{input_data.protocol}")
        # Тут може бути специфічна логіка для уніфікації форматів
        return input_data

class AuthenticationManager:
    def authenticate(self, input_data: MediaInput) -> bool:
        token = input_data.metadata.get("auth_token", "")
        if token == "valid_token_example":
            return True
        return False

class VideoProcessingService:
    def process(self, input_data: MediaInput):
        print(f"VideoProcessingService: processing video stream of
size {len(input_data.stream_data)}")

class MediaGateway:
```

```

def __init__(self, auth_manager: AuthenticationManager,
protocol_handler: ProtocolHandler):
    self.auth_manager = auth_manager
    self.protocol_handler = protocol_handler
    self.video_service = VideoProcessingService()

def receiveData(self, input_data: MediaInput):
    if not self.auth_manager.authenticate(input_data):
        print("MediaGateway: authentication failed")
        return

    if self.protocol_handler.supports(input_data.protocol):
        unified_data = self.protocol_handler.convert(input_data)
        self.routeData(unified_data)
    else:
        print("MediaGateway: unsupported protocol")

def routeData(self, input_data: MediaInput):
    # Спрощений приклад, де припускаємо, що всі дані направляються
у VideoProcessingService
    self.video_service.process(input_data)

if __name__ == "__main__":
    auth_manager = AuthenticationManager()
    protocol_handler = ProtocolHandler()
    gateway = MediaGateway(auth_manager, protocol_handler)

    valid_input = MediaInput(b"video_stream_data", "RTMP",
{"auth_token": "valid_token_example"})
    gateway.receiveData(valid_input)

    invalid_input = MediaInput(b"video_stream_data", "UNKNOWN",
{"auth_token": "valid_token_example"})

```

```
gateway.receiveData(invalid_input)

no_auth_input = MediaInput(b"video_stream_data", "RTMP",
{"auth_token": "invalid"})
gateway.receiveData(no_auth_input)
```