МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ ЦИФРОВИХ ДВІЙНИКІВ

КОНСПЕКТ ЛЕКЦІЙ

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського як навчальний посібник для аспірантів, які навчаються за спеціальністю 121 «Інженерія програмного забезпечення», освітньою програмою «Інженерія програмного забезпечення»

Київ
КПІ ім. Ігоря Сікорського
2021

Рецензент: Дичка Іван Андрійович, д-р техн. наук, професор

Відповідальний редактор: Заболотня Тетяна Миколаївна, канд. техн. наук, доцент

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол 7 від 13.05.2021 р.) за поданням Вченої ради факультету прикладної математики (протокол № 10 від 29.03.2021 р.)

Електронне мережне навчальне видання

Сулема Євгенія Станіславівна,
Сулема Ольга Костянтинівна, Шкурат Оксана Сергіївна

ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТЕХНОЛОГІЇ ЦИФРОВИХ ДВІЙНИКІВ

КОНСПЕКТ ЛЕКЦІЙ

Проєктування програмного забезпечення технології цифрових двійників: Конспект лекцій [Електронний ресурс]: навч. посіб. Для аспірантів спеціальності 121 «Інженерія програмного забезпечення», освітньої програми «Інженерія програмного забезпечення» / Є. С. Сулема, О. К. Сулема, О. С. Шкурат ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 4,72 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2021. – 144 с.

Навчальний посібник розроблено для ознайомлення аспірантів з лекційним курсом з дисципліни «Проєктування програмного забезпечення технології цифрових двійників» та призначено для аспірантів, які навчаються за спеціальністю 121 «Інженерія програмного забезпечення», освітньою програмою «Інженерія програмного забезпечення» КПІ ім. Ігоря Сікорського. Посібник містить викладання теоретичних основ та практичних аспектів проєктування програмних систем цифрових двійників, а також оброблення мультимодальних темпоральних даних моделей цифрових двійників.

ЗМІСТ

ВСТУП

Сучасний розвиток інформаційних технологій відбувається в умовах стрімкої цифрової трансформації практично всіх сфер людської діяльності. Обчислювальні системи дедалі глибше інтегруються у виробничі, наукові, освітні та соціальні процеси, виконуючи не лише допоміжні, а й ключові функції, пов’язані з аналізом даних, прогнозуванням, керуванням і підтримкою прийняття рішень. Для втілення цієї трансформації особливого значення набувають технології, що забезпечують тісний зв’язок між реальними об’єктами та їхніми цифровими представленнями. Однією з таких технологій є технологія цифрових двійників, яка поєднує методи програмного моделювання, оброблення даних у реальному часі, аналітики, візуалізації та інтелектуальних алгоритмів. Цифровий двійник дає змогу створити динамічне цифрове представлення фізичного об’єкта, що відображає його стан, поведінку та еволюцію протягом життєвого циклу. На відміну від традиційних моделей і симуляторів, цифровий двійник передбачає постійний або регулярний інформаційний зв’язок із реальним об’єктом, що підвищує ефективність аналізу даних про фізичний об’єкт, прогнозування його стану та оптимізації процесів керування.

Актуальність технології цифрових двійників зумовлена зростанням кількості ситуацій, у яких безпосередня участь людини у керуванні об’єктами або процесами є ускладненою, небезпечною або економічно недоцільною. Техногенні аварії, природні катастрофи, глобальні епідемії та інші кризові явища продемонстрували необхідність розвитку дистанційних форм взаємодії людини з реальними об’єктами. У таких умовах цифрові двійники можуть виступати ефективним інструментом для забезпечення інформованості, ситуаційної обізнаності та підтримки прийняття рішень. Ключовим трендом тут стає перехід від класичних інформаційних систем до складних кіберфізичних комплексів. У цьому контексті програмні системи перестають бути лише інструментом оброблення даних, перебираючи на себе функції керування, інтелектуального аналізу інформації та підтримки прийняття рішень, що раніше належали

виключно до компетенції людини. Водночас еволюція методологій розроблення програмного забезпечення демонструє зміщення фокусу на людино-центрований підхід. Це передбачає проєктування інтерфейсів та систем взаємодії для забезпечення високого рівня зручності використання програмного забезпечення в складних умовах експлуатації, що є критичним для людино-машинних систем нового покоління. Програмна реалізація технології цифрових двійників потребує застосування сучасних архітектурних підходів, методів проєктування складних програмних систем, засобів роботи з потоковими та мультимодальними даними, а також механізмів забезпечення надійності, масштабованості та безпеки програмного забезпечення. Разом із тим відсутність уніфікованих підходів до проєктування цифрових двійників і фрагментарність наявних рішень зумовлюють потребу у проведенні подальших досліджень. Таким чином, дисципліна «Проєктування програмного забезпечення технології цифрових двійників», яка належить до варіативної частини освітньо-наукової програми «Інженерія програмного забезпечення» підготовки докторів філософії, є наукоємною та спонукає здобувачів освіти до інноваційної діяльності.

Навчальний посібник з дисципліни «Проєктування програмного забезпечення технології цифрових двійників» спрямований на формування у здобувачів ступеня доктора філософії цілісного уявлення про цифрові двійники як об’єкт інженерного проєктування та наукових досліджень. До ключових тем, що вивчаються у дисципліні «Проєктування програмного забезпечення технології цифрових двійників», належать теми, присвячені архітектурним підходам до проєктування програмних систем цифрових двійників, обробленню та синхронізації мультимодальних потоків даних, специфікації моделей, стандартизації програмних рішень тощо. Таким чином, посібник охоплює теоретичні основи та практичні аспекти проєктування програмного забезпечення технології цифрових двійників, зокрема, архітектурні підходи, методи моделювання та аналізу, а також проблеми забезпечення життєвого циклу програмних реалізацій цифрових двійників, що створює підґрунтя для виконання самостійних наукових досліджень у цій галузі.

1. ВСТУП ДО ТЕХНОЛОГІЇ ЦИФРОВИХ ДВІЙНИКІВ

У розділі розглянуто концептуальні засади технології цифрових двійників як сучасного підходу до цифрового подання, аналізу та керування фізичними об'єктами протягом їхнього життєвого циклу. Наведено класифікацію цифрових двійників за рівнем інтеграції з фізичним об'єктом, рівнем складності та функціональним призначенням. Описано узагальнений процес створення цифрового двійника як складної програмної системи, інтегрованої з джерелами даних, аналітичними модулями та користувацькими інтерфейсами. Проаналізовано програмні платформи для реалізації цифрових двійників, їх можливостей і обмеження. Розділ формує інформаційну основу для подальшого розгляду шляхів проєктування програмного забезпечення цифрових двійників.

1.1. Концепція цифрового двійника

Ідея створення цифрових моделей фізичних об'єктів не є новою [1]. Протягом десятиліть в інженерії використовувалися моделі різноманітних типів: математичні, імітаційні, CAD-моделі, системи комп'ютерного моделювання фізичних процесів тощо. Однак усі ці підходи мають спільну рису – модель існує відокремлено від реального об'єкта. Вона створюється на етапі проєктування або аналізу та використовується в обмеженому часовому інтервалі. Також недоліками традиційного підходу до моделювання є те, що модель об'єкта не оновлюється автоматично і не відображає поточний стан реального об'єкта.

Розвиток кіберфізичних систем [2, 3] та Інтернету речей (IoT) [4, 5] створив принципово нові умови для цифровізації фізичних об'єктів, оскільки з'явилася можливість безперервного збору даних про них, фізичні об'єкти стали «спостережуваними» і, відповідно, виникла потреба в постійному цифровому супроводі цих об'єктів. Саме на цьому етапі стало очевидно, що статичні цифрові моделі більше не відповідають складності реальних систем, тож виникла потреба у комплексному динамічному моделюванні, яке здійснюється у режимі, близькому до реального часу, дає змогу відображати не лише структуру, а й поведінку фізичного об'єкта, що може бути використане для аналізу, прогнозування та оптимізації процесів, які виникають або стосуються функціонування реальних фізичних об'єктів. Таким новим підходом до створення цифрових моделей є *технологія цифрових двійників* [1, 6–10], яка

реалізує подання у цифровій формі фізичного об’єкта, його минулого, поточного і майбутнього стану, поведінки, характеристик, зовнішнього вигляду тощо.

Відповідно до визначення, сформульованого авторами концепції цифрового двійника, Майклом Грівзом (Michael Grieves) та Джоном Вікерсом (John Vickers), у роботі [10], *цифровий двійник* – це сукупність віртуальних інформаційних конструкцій (virtual information constructs), яка повністю описує фізичний об’єкт – від мікрорівня (рівень окремого елементу) до макрорівня (загальний вигляд, геометричне подання, загальні властивості об’єкта в цілому).

Цифровий двійник являє собою модель фізичного об’єкта – *фізичного двійника*, яка якнайповніше відображає його характеристики у динаміці протягом певного періоду часу. Концепція цифрового двійника передбачає подання, оброблення, маніпулювання усіма даними, що характеризують стан фізичного двійника, та отримання певних компонент цих даних по мірі виникнення потреби у них при вирішенні задач аналізу та прогнозування стану фізичного двійника, оптимізації процедур керування фізичним двійником тощо. Цифровий двійник також має забезпечувати реалістичність відображення фізичного двійника засобами мультимедійних користувацьких інтерфейсів.

Термін «цифровий двійник (digital twin)» вперше був застосований у звіті [11] Національного управління з аеронавтики і дослідження космічного простору США (NASA). Цифровий двійник там був визначений як інтегрована мультифізична (multiphysics), мультимасштабована (multiscale), ймовірнісна симуляція об’єкта дослідження, яка використовує найкращі наявні моделі, що постійно вдосконалюються, дані, що безперервно надходять з давачів реального об’єкта та накопичуються, щоб відобразити життєвий цикл фізичного двійника. Створення цифрових двійників для літальних апаратів NASA мало на меті виявлення аномалій поведінки складових фізичного двійника – реального літального апарата – до виникнення у ньому аварійної ситуації та, таким чином, вчасне запобігання виникненню надзвичайній ситуації [6, 11]. Так, цифровий двійник літака інтегрує дані, що отримуються з давачів вбудованої системи, що знаходиться на борту реального літака, містить дані про історію його

обслуговування, аварійних ситуацій тощо (рис. 1.1). Таким чином, цифровий двійник є *індивідуалізованою* реалістичною віртуальною моделлю певного об'єкта – фізичного двійника.

Рис. 1.1. Концепція цифрового двійника

Цифровий двійник складається з *візуальної моделі* та *поведінкової моделі* фізичного об'єкта, що реалізовані на основі відповідних математичних моделей і моделей подання даних та забезпечують синхронізацію між віртуальною і реальною системою на рівні даних, що надходять із давачів, встановлених для постійного моніторингу досліджуваного об'єкта [9].

Таким чином, цифровий двійник являє собою *складну програмну систему*, інтегровану зі зовнішніми джерелами даних (давачами) та актуаторами, яка включає сховище даних та програмні модулі, що забезпечують отримання, оброблення та візуалізацію даних, і у який комплексна модель фізичного об'єкта є лише одним з компонентів цієї системи.

1.2. Класифікація цифрових двійників

Класифікація цифрових двійників може бути здійснена за такими класифікаційними ознаками, як рівень інтеграції цифрового та фізичного двійників, рівень складності та призначення цифрового двійника.

За *рівнем інтеграції цифрового та фізичного двійників* розрізняють [8] цифрову модель (digital model), цифрову тінь (digital shadow) та, власно, цифровий двійник (рис. 1.2).

Рис. 1.2. Інтеграція фізичного та цифрового об'єктів:
(а) цифрова модель, (б) цифрова тінь, (в) цифровий двійник

Цифрова модель (рис. 1.2а) являє собою цифрове подання наявного або проєктованого фізичного об'єкта, яке не використовує жодної форми автоматизованого обміну даними між фізичним та цифровим об'єктами. Цифрова модель передбачає використання імітаційних моделей, математичних моделей або інших видів моделювання фізичного об'єкта, коли автоматична інтеграція даних відсутня. Цифрові дані існуючих фізичних систем можуть використовуватися для розроблення таких моделей, але обмін даними здійснюється вручну. Зміна стану фізичного об'єкта не має прямого впливу на цифровий об'єкт і навпаки.

Цифрова тінь (рис. 1.2б) – це цифрове подання фізичного об'єкта, при якому передбачено автоматизований односторонній потік даних між фізичним об'єктом та його цифровою реалізацією. Зміна стану фізичного об'єкта призводить до зміни стану цифрового об'єкта, але не навпаки.

Цифровий двійник (рис. 1.2в) є цифровим поданням фізичного двійника, при якому потоки даних між фізичним об'єктом і цифровим об'єктом наявні в обох напрямках, тобто зміна стану фізичного об'єкта безпосередньо призводить до зміни стану цифрового об'єкта, а також навпаки – зміна стану цифрового

об’єкта призводить до зміни стану фізичного об’єкта, що утворює механізм керування фізичним двійником через цифрового двійника.

За *рівнем складності* (рис. 1.3) візуальної моделі цифрового двійника розрізняють [12] цифрові двійники чотирьох рівнів: *підготовчій цифровий двійник* (pre-digital twin), *цифровий двійник*, *адаптивний цифровий двійник* (adaptive digital twin) та *інтелектуальний цифровий двійник* (intelligent digital twin).

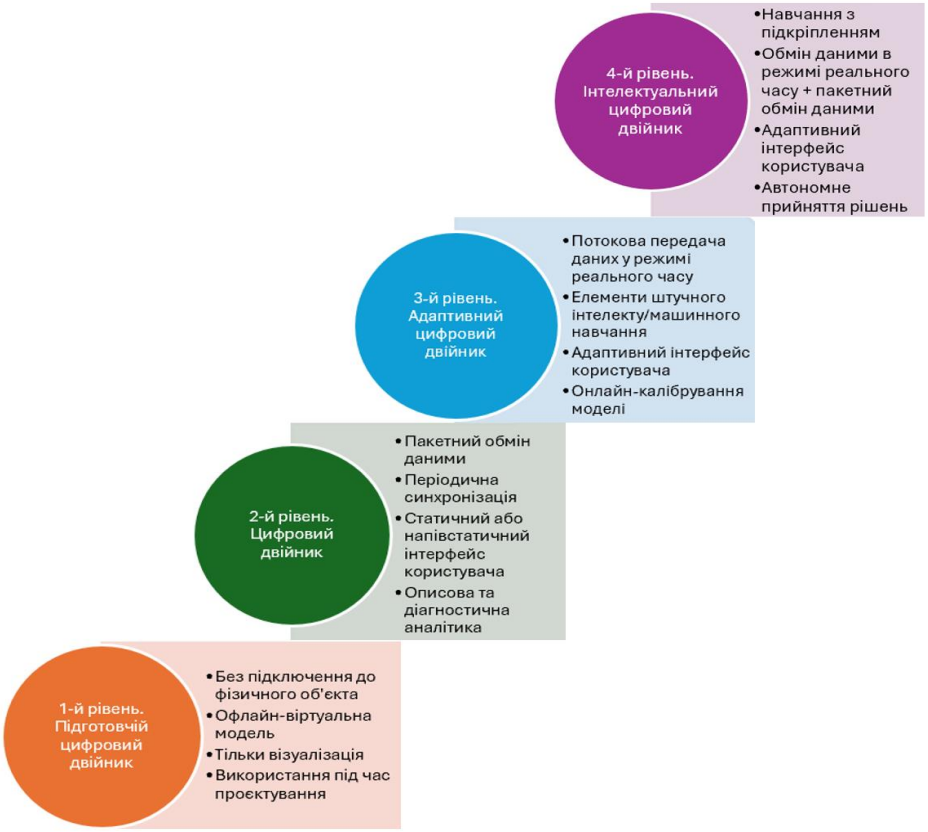

Рис. 1.3. Рівні складності цифрового двійника

Підготовчій цифровий двійник (перший рівень складності) є спрощеною віртуальною моделлю, у якій зв’язок з фізичним двійником відсутній.

Власне *цифровий двійник* (другий рівень складності) являє собою віртуальну модель, яка взаємодіє з фізичним двійником шляхом обміну пакетними даними.

Адаптивний цифровий двійник (третій рівень складності) є віртуальною моделлю фізичного двійника з адаптивним користувацьким інтерфейсом, у якій

дані передаються у реальному часі та застосовуються елементи штучного інтелекту.

Інтелектуальний цифровий двійник (четвертий рівень складності) – це віртуальна модель фізичного двійника з адаптивним користувацьким інтерфейсом та застосуванням алгоритмів навчання з підкріпленням, у якій підтримується як пакетне передавання даних та і обмін даними між цифровим та фізичним об’єктами у реальному часі.

За *призначенням* розрізняють [1] цифрові двійники двох основних типів: цифровий двійник-прототип (digital twin prototype) та цифровий двійник-екземпляр (digital twin instance).

Цифровий двійник-прототип створюється для фізичного об’єкта, який ще не існує та буде створений після дослідження цифрового двійника. Він містить набори даних, необхідні для опису та створення фізичного двійника. Ці набори даних включають вимоги до фізичного двійника, тривимірну модель фізичного об’єкта, специфікацію матеріалів та компонентів, з яких він буде вироблений тощо.

Цифровий двійник-екземпляр створюється для конкретного існуючого фізичного об’єкта, з яким цей цифровий двійник залишається пов’язаним протягом усього життєвого циклу або циклу дослідження фізичного двійника.

Окрім цифрового двійника-прототипу та цифрового двійника-екземпляру, розрізняють *цифровий двійник-агрегат* (digital twin aggregate), який отримується в результаті агрегації деякої сукупності цифрових двійників-екземплярів та використовується для дослідження певного набору (класу) фізичних об’єктів, а не окремого досліджуваного об’єкта.

Середовище цифрового двійника (digital twin environment) – це сукупність програмного та програмно-апаратного забезпечення процесу дослідження одного або декількох фізичних об’єктів. Середовище цифрового двійника має дві основні функції: передбачувальну та інформаційну. Передбачувальна функція реалізується у тому, що цифровий двійник використовується для прогнозування майбутньої поведінки фізичного двійника. У випадку застосування цифрового

двійника-прототипу прогнозування дозволяє проаналізувати різні варіанти реалізації фізичного двійника ще до його фізичного створення. У випадку застосування цифрового двійника-екземпляру прогнозування має на меті передбачення можливих критичних ситуацій для фізичного двійника та моделювання розвитку стану фізичного двійника залежно від варіантів впливу (взаємодії, керування) на цей фізичний об'єкт. Інформаційна функція полягає у тому, що використання цифрового двійника дає змогу отримати актуальну інформацію про фізичний об'єкт в цілому або про його конкретні характеристики. Ця функція є важливою для вирішення широкого кола дослідницьких задач.

1.3. Узагальнений процес створення цифрового двійника

Для реалізації проєкту цифрового двійника [13–15] необхідні три базові складові: фізичний об'єкт, джерела даних про стан цього об'єкта та програмне забезпечення для моделювання й інтеграції цифрового двійника. Процес створення цифрового двійника об'єкта характеризується високим рівнем складності і може бути умовно поділений на дванадцять етапів (рис. 1.4).

Рис. 1.4. Узагальнений процес створення цифрового двійника

На першому етапі здійснюється визначення мети проєкту. Цей етап передбачає ідентифікацію фізичного двійника. Необхідно чітко сформулювати цілі та завдання створення цифрового двійника, а також окреслити бажані функціональні можливості та очікувані результати. Ця стадія планування є фундаментальною для всього проєкту та гарантує, що отриманий цифровий продукт відповідатиме потребам.

Другий етап спрямований на збір та інтеграцію даних. Він передбачає визначення всіх релевантних джерел даних, до яких можуть належати IoT-давачі, архівні записи та інженерні специфікації. Критично важливим є розроблення надійних протоколів збору даних та відповідної інфраструктури, а також інтеграція інформації з різнорідних джерел на єдину уніфіковану платформу. Це забезпечуватиме доступ до вичерпної та точної інформації цифрового двійника про його фізичний аналог.

Третій етап – це 3D-моделювання об’єкта. Цей етап не є обов’язковим, проте створення тривимірної моделі є важливою складовою процесу створення цифрових двійників для багатьох застосувань. Він передбачає розроблення моделі «з нуля» або імпорт наявних 3D-моделей фізичного активу. На цьому етапі увага має бути приділена забезпеченню точності та деталізації віртуального представлення, оскільки точність моделі безпосередньо впливає на ефективність симуляції та аналізу поведінки фізичного об’єкта.

Четвертий етап призначений для інтеграції мереж давачів. Він передбачає встановлення та конфігурацію пристроїв IoT на фізичному активі, а також організацію передавання даних на цифрову платформу в режимі реального часу. Ці давачі мають безперервно забезпечувати програмну систему цифрового двійника актуальною інформацією про стан та продуктивність фізичного об’єкта.

П’ятий етап охоплює процеси впровадження алгоритмів очищення та попереднього оброблення даних для забезпечення їхньої якості, розроблення аналітичних моделей для інтерпретації інформації, а також налаштування алгоритмів машинного навчання для реалізації прогностичних функцій. Саме на

цьому етапі розкривається цінність цифрового двійника завдяки наданню інформації, яку неможливо отримати безпосередньо з «сирих» даних.

Шостий етап – це створення інтерфейсу користувача. Він передбачає розроблення та/або налаштування інтуїтивно зрозумілої інформаційної панелі (dashboard) для візуалізації даних та аналітичних висновків, а також розроблення інтерфейсів керування для взаємодії з цифровим двійником. Мета полягає у поданні складної інформації у чіткому, зрозумілому форматі та забезпеченні користувачів інструментами для ефективного дослідження та керування моделлю.

На сьомому етапі здійснюється моделювання та тестування, що включає розроблення симуляційних можливостей для тестування різноманітних сценаріїв та валідацію роботи цифрового двійника шляхом порівняння з реальними даними. Завдяки тестуванню розробники можуть удосконалювати моделі та алгоритми, забезпечуючи точність прогнозів у широкому діапазоні умов експлуатації.

Восьмий етап присвячений інтеграції ядра створеного цифрового двійника з наявними системами, що передбачає підключення цифрового двійника до відповідних корпоративних систем, таких як платформи ERP (планування ресурсів підприємства) або CRM (управління взаємовідносинами з клієнтами). Забезпечення сумісності та безперервного потоку даних між цими системами є ключовим для інтеграції цифрового двійника в загальну екосистему даних організації.

На дев’ятому етапі відбуваються процедури забезпечення інформаційної безпеки. Цей етап охоплює встановлення комплексних протоколів кібербезпеки, реалізацію механізмів контролю доступу та налаштування заходів аутентифікації. Враховуючи потенційну чутливість даних, цей крок є критично важливим для збереження цілісності та конфіденційності цифрового двійника та пов’язаної з ним інформації.

Десятий етап – це постійне вдосконалення та технічне обслуговування, що включає налагодження механізмів для постійного збору даних та оновлення

моделей, а також затвердження протоколів регулярного технічного обслуговування та калібрування цифрового двійника. Це гарантує збереження точності та актуальності системи з плином часу, дозволяючи адаптуватися до змін фізичного активу або умов його експлуатації.

Одинадцятий етап призначений для навчання персоналу та розгортання системи. Він включає навчання відповідного персоналу методам використання та інтерпретації даних цифрового двійника, а також розгортання системи у контрольованому середовищі перед повномасштабним впровадженням. Належна підготовка та поетапний підхід до впровадження сприяють адаптації користувачів до нової технології та своєчасному виявленню й усуненню можливих проблем.

Врешті, дванадцятий етап – це моніторинг та оптимізація цифрового двійника, що передбачає безперервний моніторинг продуктивності та точності цифрового двійника, а також впровадження покращень та оптимізаційних заходів на основі зворотного зв’язку від користувачів та нових даних. Цей ітеративний процес забезпечує відповідність цифрового двійника змінним потребам організації та дозволяє використовувати переваги новітніх технологічних досягнень.

Дотримання зазначених етапів дозволяє створювати комплексні та ефективні цифрові двійники, що надають цінні аналітичні дані та забезпечують прийняття рішень на основі об’єктивної інформації. Слід ще раз підкреслити, що поділ на етапи є умовним, оскільки деякі з них можуть виконуватися в іншому порядку чи поєднуватися для прискорення та оптимізації процесів розроблення. Специфіка реалізації кожного етапу може варіюватися залежно від складності системи, що моделюється, та цільового призначення цифрового двійника.

1.4. Програмне забезпечення технології цифрових двійників

Розглянемо програмні продукти, які можуть бути застосовані для реалізації технології цифрових двійників.

AWS IoT TwinMaker (рис. 1.5) є спеціалізованою платформою для створення та управління цифровими двійниками реальних систем, яка значно спрощує процес розроблення для промислових, виробничих та комерційних об’єктів [16]. Функціональність сервісу надає можливість створювати віртуальні репрезентації шляхом агрегації наявних 3D-моделей, даних з давачів та корпоративної інформації без необхідності глибоких знань у програмуванні. Високий рівень інтероперабельності досягається завдяки вбудованим конекторам до різноманітних сервісів AWS та сторонніх додатків, що полегшує інтеграцію з наявними джерелами даних. Також підтримується створення веб-додатків для візуалізації цифрових двійників у двовимірному та тривимірному форматах, що забезпечує моніторинг операцій у реальному часі, історичний аналіз та симуляцію процесів. Архітектура AWS IoT TwinMaker, посилена функціями нормалізації даних та вбудованої аналітики, може масштабуватися для підтримки складних систем із мільйонами пристроїв, надаючи організаціям комплексний інструментарій для підвищення ефективності та прогнозування технічного обслуговування.

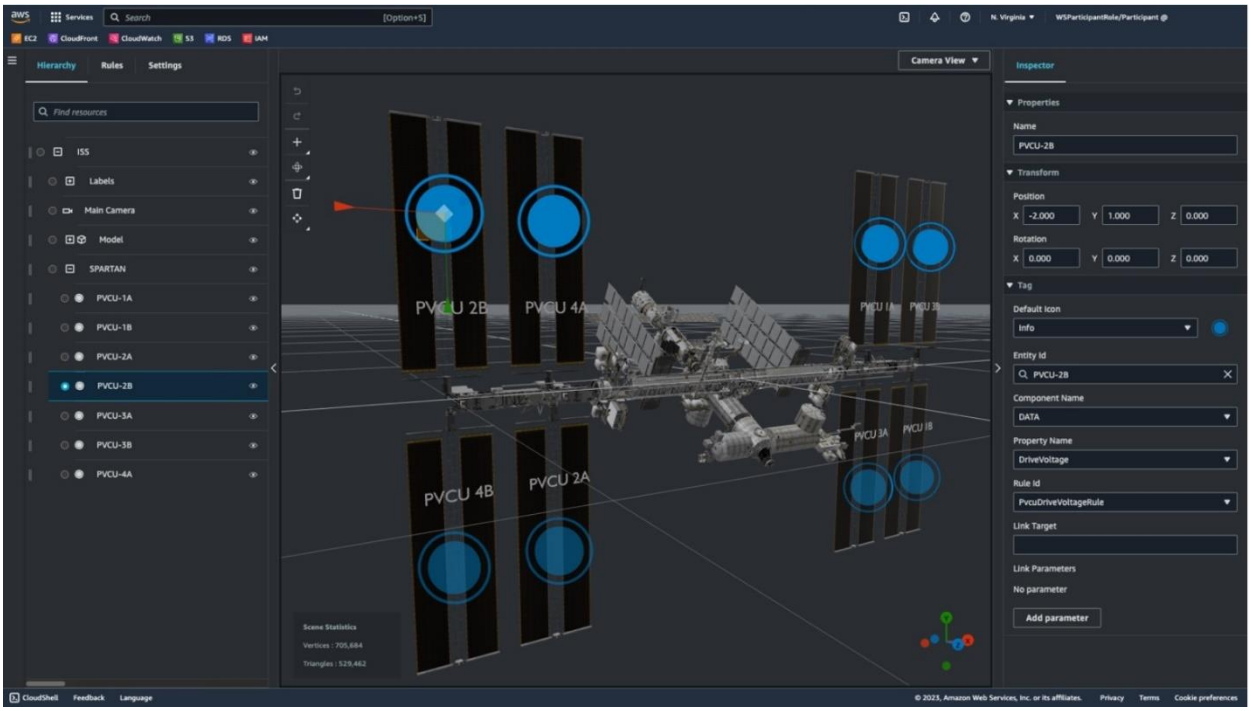

Рис. 1.5. AWS IoT TwinMaker

Серед платформ, що частково реалізують концепцію цифрового двійника, слід відзначити IBM® Watson™ IoT Platform [17]. Це програмне середовище забезпечує збір даних про об’єкт моніторингу від мережі давачів. Платформа підтримує різноманітні формати даних, включно з JSON, текстовими форматами та спеціалізованими форматами для геопросторової інформації.

Архітектура зберігання даних у IBM® Watson™ IoT Platform передбачає три рівні.

1. Оперативні дані: інформація, що надходить від давачів у режимі реального часу, зберігається у базі даних NoSQL [18].
2. Аналітичні дані: масиви, що потребують подальшого оброблення, акумулюються у хмарному сховищі, організованому за технологією Data Lake [19, 20].
3. Архівні дані: інформація, що підлягає тривалому зберіганню, розміщується у хмарному сховищі об’єктів.

Суттєвим обмеженням цієї платформи є відсутність вбудованих компонентів для візуалізації та формування повноцінної моделі цифрового двійника.

ANSYS Twin Builder (рис. 1.6) є програмною платформою, орієнтованою на швидке створення, валідацію та розгортання цифрових двійників з акцентом на фізичне моделювання [21]. Інструментарій платформи поєднує можливості мультифізичної симуляції, системного моделювання та засоби розроблення вбудованого програмного забезпечення. Для досягнення балансу між складністю моделі та обчислювальною ефективністю програмний комплекс підтримує різноманітні підходи, включаючи 0D та 1D системне моделювання, 3D-орієнтовану фізичну симуляцію, а також методи моделювання зниженого порядку (Reduced-Order Modelling). Платформа пропонує широкі можливості підключення зовнішніх модулів, забезпечуючи інтеграцію з IoT-платформами та корпоративними системами для обміну даними в реальному часі. Ключовими особливостями платформи є модельно-орієнтоване проєктування, напівнатурне моделювання (Hardware-in-the-Loop) та функції прогнозного обслуговування.

ANSYS Twin Builder є оптимальним рішенням для аерокосмічної, автомобільної та енергетичної галузей, оскільки завдяки високоточним симуляціям та відповідній функціональності ця платформа уможливлює скорочення обсягу фізичних випробувань та оптимізацію продуктивності фізичних двійників протягом усього їхнього життєвого циклу.

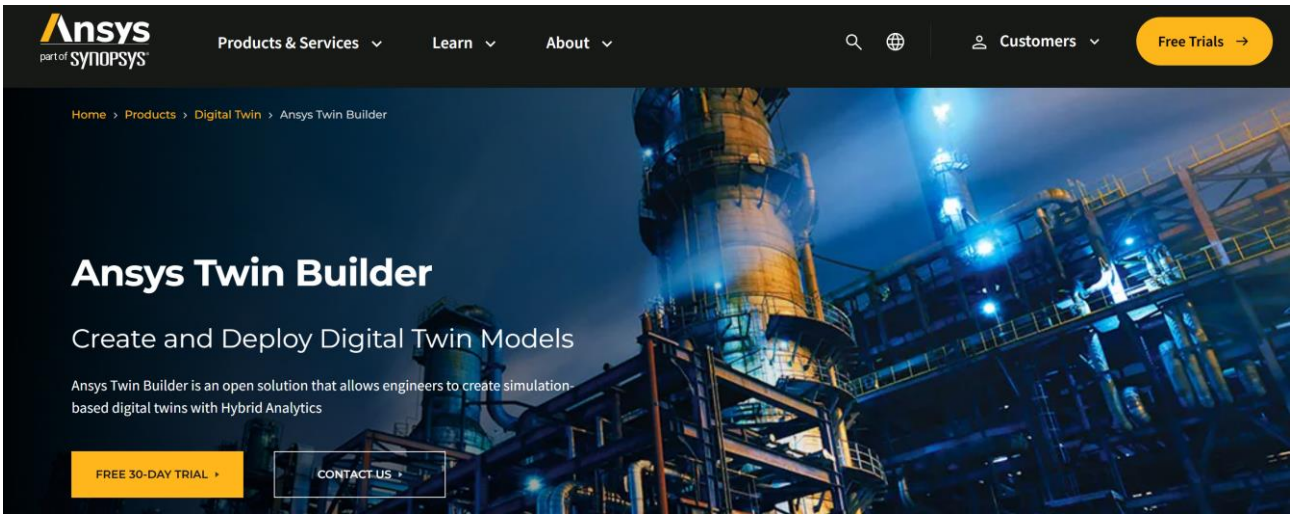

Рис. 1.6. ANSYS Twin Builder

У виробничому секторі широкого застосування набуло програмне забезпечення Seebo Digital Twin [22]. Його основним призначенням є моделювання роботи технологічних ліній, створення графічних прототипів промислових продуктів та симуляція виробничих процесів для виявлення «вузьких місць». Однак, можливості цього програмного забезпечення щодо інтеграції з фізичним двійником у реальному часі є обмеженими, а сфера застосування здебільшого сфокусована на промислове виробництво.

Для створення візуальних симуляційних моделей у промисловості доцільним є використання спеціалізованого інструментарію компанії Simio [23, 24]. Ця система дає змогу формувати візуальні моделі на основі стандартних бібліотек об’єктів. Редактор симуляцій надає можливості для визначення операцій, часових параметрів та логічних зв’язків між віртуальними об’єктами. Для опису властивостей фізичного об’єкта у системі передбачено близько 50 типів даних, які класифікують як елементи (наприклад, матеріал, стан тощо) та об’єкти (наприклад, сутність, ресурс, вузол та інші).

На сьогодні найбільш розвинену функціональність для розроблення цифрових двійників пропонує платформа Microsoft Azure Digital Twins (рис. 1.7) [25–28].

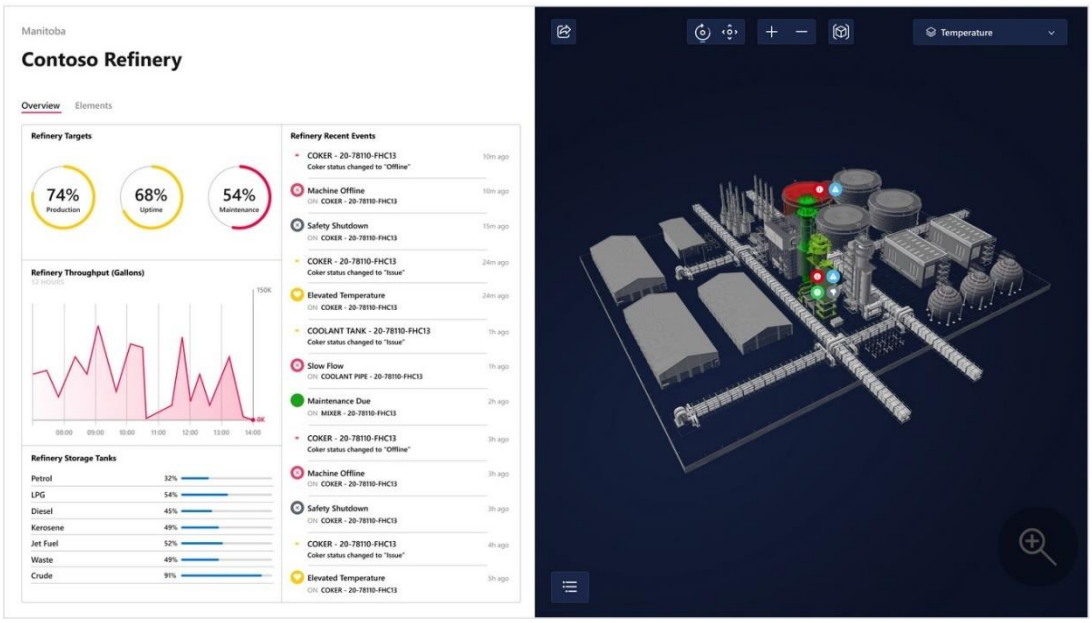

Рис. 1.7. Microsoft Azure Digital Twins

Керування життєвим циклом цифрових двійників (створення, модифікація, вилучення) здійснюється через DigitalTwins API, які поділяються на два рівні: рівень управління (Control Plane) – операції з екземпляром сервісу Azure Digital Twins в цілому, та рівень даних (Data Plane) – управління елементами всередині екземпляра, а також категорії *DigitalTwinsModels* (управління моделями), *DigitalTwins* (операції над двійниками), *Query* (пошукові запити) та *EventRoutes* (маршрутизація подій).

Для моделювання складних систем (наприклад, виробничих ліній) використовується графова модель, що об’єднує цифрові двійники окремих компонентів [29]. Візуалізація та редагування графа здійснюються за допомогою інструменту Azure Digital Twins Explorer [30]. Зберігання даних реалізовано на базі Azure Data Lake [26], а аналітичне оброблення даних забезпечується засобами Azure Synapse Analytics [28]. Основним цільовим сегментом платформи є великі інфраструктурні та промислові об’єкти (енергосистеми, транспортні мережі, «розумні» будівлі тощо).

Серед інших програмних рішень слід зазначити Cosmo Tech Enterprise Digital Twin [31], SAP Digital Manufacturing Cloud [32] та продукти Autodesk для цифрових двійників [33]. Спільною рисою цих систем є їхня орієнтація на виробничі та бізнес-процеси, що обмежує їх використання в інших сферах, таких як, наприклад, медицина чи освіта.

Проведений аналіз наявних програмних засобів оброблення мультимодальних даних цифрових двійників дозволяє виявити низку системних недоліків. Першим з них є галузева обмеженість, яка проявляється у тому, що більшість систем орієнтовані виключно на інженерні та виробничі завдання, ігноруючи специфіку гуманітарних, медичних та освітніх галузей. Другий недолік – недостатній рівень візуалізації. Візуальні моделі часто обмежуються спрощеним двовимірним графічним поданням фізичного об’єкта, не використовуючи повний спектр мультимодальних даних для підвищення реалістичності. І, врешті, третім недоліком є відсутність універсального підходу, оскільки на цей час не сформовано єдиної методології подання та оброблення темпоральних мультимодальних даних, що ускладнює масштабування концепції цифрового двійника на довільні класи об’єктів дослідження. Таким чином, задача проєктування нових програмних систем, як спеціалізованих, так і універсальних, є актуальною науково-дослідною та інженерною задачею.

Контрольні запитання

1. У чому полягає принципова відмінність між традиційною цифровою моделлю та цифровим двійником з точки зору життєвого циклу та актуалізації даних?
2. Які передумови (технологічні та концептуальні) сприяли виникненню технології цифрових двійників?
3. Поясніть сутність двонапрямного обміну даними між фізичним та цифровим двійниками і його значення для керування.
4. Охарактеризуйте різницю між цифровою моделлю, цифровою тінню та цифровим двійником за рівнем інтеграції.

5. Які рівні складності цифрових двійників виділяють і чим вони принципово відрізняються?
6. У чому полягає різниця між цифровим двійником-прототипом і цифровим двійником-екземпляром?
7. Яку роль відіграє середовище цифрового двійника та які його ключові функції?
8. Які етапи створення цифрового двійника є критичними для забезпечення достовірності прогнозування?
9. Чому цифровий двійник розглядається як складна програмна система, а не лише як модель?
10. Які системні обмеження сучасних платформ цифрових двійників виділяються у наукових дослідженнях?

Завдання для самостійної підготовки

1. Проаналізуйте та обґрунтуйте принципову відмінність цифрового двійника від симуляційної моделі з позицій програмної архітектури та потоків даних.
2. Визначте та обґрунтуйте межу між цифровим двійником і системою підтримки рішень; наведіть приклади їхнього функціонального перетину.
3. Проаналізуйте можливості відомих платформ для створення цифрових двійників з точки зору масштабованості, візуалізації та роботи з мультимодальними даними.
4. Проаналізуйте обмеження відомих платформ для створення цифрових двійників з точки зору масштабованості, візуалізації та роботи з мультимодальними даними.
5. Запропонуйте класифікацію цифрового двійника для конкретної предметної галузі та обґрунтуйте вибір класифікаційних ознак.

2. ПРОЄКТУВАННЯ ПРОГРАМНИХ СИСТЕМ ЦИФРОВИХ ДВІЙНИКІВ

У розділі розглянуто проєктування програмних систем цифрових двійників з позицій сучасної інженерії програмного забезпечення. Проаналізовано життєвий цикл цифрового двійника як програмного продукту у тісному зв'язку з життєвим циклом фізичного активу та показано його відмінність від класичних моделей SDLC. Сформульовано та систематизовано функціональні й нефункціональні вимоги до програмного забезпечення цифрових двійників, зокрема вимоги щодо роботи з даними, моделями, аналітикою, візуалізацією та зворотним впливом. Окрему увагу приділено референтним архітектурам платформ цифрових двійників. Розділ формує інженерну основу для подальшого розроблення та аналізу програмних платформ цифрових двійників.

2.1. Життєвий цикл цифрового двійника як програмного продукту

Згідно з міжнародним стандартом ISO/IEC 12207:2017 «Systems and software engineering – Software life cycle processes» [34], *життєвий цикл* – це еволюція системи, продукту, послуги, проєкту або іншого створеного людиною об'єкта від задуму до виведення з експлуатації. Для аналізу життєвого циклу цифрового двійника з позицій розроблення програмного забезпечення спочатку пригадаємо життєвий цикл та методологію розроблення програмного продукту.

Життєвий цикл програмного продукту (Software Development Life Cycle – SDLC) складається з шести фаз [35]. *Першою фазою* є збір та аналіз вимог до програмного забезпечення. Основною діяльністю на цьому етапі є інтерв'ю зі стейкхолдерами (замовниками), аналіз існуючих систем, моделювання варіантів використання. Результатом першого етапу є формування специфікації вимог до програмного забезпечення (Software Requirements Specification – SRS) [36]. На *другій фазі* здійснюється проєктування програмного забезпечення, а саме, архітектурне проєктування (High-Level Design) та детальне проєктування (Low-Level Design) [37]. Архітектурне проєктування передбачає вибір технологічного стеку, архітектурних шаблонів, проєктування бази даних. Детальне проєктування включає опис алгоритмів, інтерфейсів класів, API. *Третя фаза* – це реалізація, тобто безпосереднє написання коду згідно з вимогами та проєктною документацією. Під час *четвертої фази* відбувається верифікація (перевірка відповідності специфікації) та валідація (перевірка відповідності

очікуванням користувача) програмного продукту. *П'ята фаза* – це розгортання, тобто процес постачання готового продукту кінцевим користувачам. У сучасних підходах (DevOps) цей процес часто автоматизований через CI/CD (Continuous Integration / Continuous Deployment) [38–40]. Найдовшою фазою життєвого циклу є *шоста фаза* – супровід програмного продукту, який поділяється на: коригувальний (виправлення помилок, виявлених після релізу), адаптивний (зміни через зовнішні чинники), удосконалювальний (додавання нових функцій за запитом користувачів) та превентивний (рефакторинг коду для запобігання майбутнім проблемам).

Для організації процесів, що відбуваються на кожній фазі життєвого циклу, застосовують моделі [41–43]. Прикладами моделей життєвого циклу можуть слугувати каскадна модель, V-подібна модель, Agile тощо. Так, класичною моделлю життєвого циклу є *каскадна модель*, у якій передбачається, що кожна наступна фаза починається тільки після повного завершення попередньої. Її перевагами є чіткість та документованість. Недоліки каскадної моделі – висока вартість внесення змін на пізніх етапах та неможливість швидкого отримання зворотного зв'язку, тому доцільно застосовувати цю модель для розроблення програмного забезпечення для задач, де вимоги не змінюються, наприклад, для створення систем з критичними вимогами до безпеки (авіація, медицина тощо). Розширенням каскадної моделі є *V-подібна модель* (V-Model), у якій кожному етапу розроблення відповідає певний етап тестування. Модель *Agile* є ітеративною та інкрементальною. Вона базується на «Agile Manifesto» [42]. Розроблення згідно з цією моделлю ведеться короткими циклами (спринтами), в кінці кожного з яких замовник отримує робочу версію продукту (інкремент). Перевагами моделі Agile є гнучкість до змін та швидкий вихід на ринок, а до недоліків слід віднести складність в оцінці фінального бюджету та строків.

Тепер розглянемо *життєвий цикл цифрового двійника*. Він є нерозривно пов'язаним із життєвим циклом фізичного об'єкта (активу). Не можливо розглядати програмну реалізацію цифрового двійника окремо від фізичного

двійника, якого цей цифровий двійник відображає. З огляду на це, життєвий цикл цифрового двійника поділяють на три макро-етапи [44, 45]:

1) початок життєвого циклу (Beginning of Life – BoL);
2) середина життєвого циклу (Middle of Life – MoL);
3) кінець життєвого циклу (End of Life – EoL).

Розглянемо їх докладно.

Eman BoL відповідає етапам проєктування та реалізації в класичному розумінні, але має суттєві відмінності, оскільки цифровий двійник часто з'являється раніше за свій фізичний прототип. У цьому разі спочатку створюється віртуальна модель майбутнього об'єкта – цифровий двійник-прототип (Digital Twin Prototype – DTP) [1]. Для цього використовуються CAD-системи та інструменти симуляції (наприклад, ANSYS Twin Builder), щоб перевірити фізику процесів, як-от, аеродинаміку, теплообмін, тощо. Метою розроблення DTP є верифікація дизайну до витрачання ресурсів на фізичне виробництво. За словами Майкла Грівза (Michael Grieves), автора концепції цифрових двійників, цінність DTP полягає в можливості «помилятися віртуально», щоб уникнути помилок у реальному виробі [1].

Наступним кроком є перехід до етапу виробництва фізичного двійника. Коли фізичний актив (наприклад, турбіна) виготовляється, цифровий двійник трансформується з «прототипу» в «екземпляр» (Digital Twin Instance – DTI) [1]. При цьому унікальний серійний номер фізичного активу пов'язується з його цифровою копією. Далі встановлюються давачі, налаштовуються протоколи передачі даних (MQTT [46], OPC UA [47, 48]) та перевіряється зв'язок між фізичним об'єктом і хмарною платформою (наприклад, Azure Digital Twins).

Eman MoL є основною фазою експлуатації, яка відповідає фазі супроводу в SDLC, але є значно динамічнішою. Саме тут реалізується двосторонній потік даних, що є головною відмінністю цифрового двійника від звичайної 3D-моделі. Цей етап включає три ключові процеси:

1) моніторинг та синхронізація – двійник постійно оновлює свій стан на основі телеметрії з фізичного об'єкта в реальному часі (наприклад, якщо двигун

нагрівається в реальності, він «нагрівається» і на моніторі оператора);

2) предиктивна аналітика (використання машинного навчання для прогнозування відмов) – двійник аналізує історичні дані та попереджає про необхідність ремонту *до* моменту поломки;
3) оптимізація – проведення симуляцій «що, якщо» (What-If Analysis) [49] для перевірки критичних ситуацій на двійнику, не ризикуючи реальним обладнанням (наприклад, «Що станеться з конвеєром, якщо збільшити швидкість на 15%?»).

Eman EoL настає, коли фізичний актив виводиться з експлуатації (списання, переробка). При цьому життєвий цикл цифрового двійника не обов'язково завершується миттєво. Так, двійник може бути переведений у статичний стан і надалі використовуватися як цифровий архів, який зберігає історію експлуатації, дані про поломки та ремонти. Інший варіант – використання цифрового двійника для подальших досліджень. У цьому випадку дані, накопичені за весь період існування двійника, передаються розробникам для покращення проєктування наступних поколінь продуктів, що замикає цикл і фактично починає фазу BoL нових систем. Отже, життєвий цикл цифрового двійника – це безперервний процес зближення фізичного та віртуального об'єктів. На відміну від звичайного програмного забезпечення, де можна випустити оновлення і «забути» про стару версію, цифровий двійник мусить еволюціонувати синхронно зі старінням, зносом та модернізацією свого фізичного двійника.

Таким чином, фундаментальна відмінність між класичним програмним продуктом та програмним забезпеченням цифрового двійника полягає у розумінні поняття «завершеність»: програмний продукт у класичному SDLC може вважатися завершеним після прийняття його замовником, натомість, цифровий двійник ніколи не є завершеним, доки актив функціонує, оскільки він повинен перебувати в стані постійної синхронізації з реальним об'єктом. Якщо фізичний об'єкт змінюється (наприклад, відбулася деградація підшипника), а програмна модель – ні, двійник втрачає свою валідність.

Сучасні платформи цифрових двійників мають реалізовувати комплексний підхід для підтримки повного життєвого циклу цифрового двійника. Такий життєвий цикл охоплює етапи проєктування та побудови віртуальної моделі, інтеграції даних з фізичних об’єктів, виконання аналітики та прогнозування, а також розгортання і подальшої експлуатації цифрового двійника у хмарних, крайових (edge) або автономних середовищах. У межах цих платформ цифровий двійник розглядається не як окрема модель чи сервіс, а як багатокомпонентна програмна система, що поєднує моделювання, дані, алгоритми та інтерфейси взаємодії.

Однією з базових функціональних можливостей таких платформ є підтримка процесів моделювання та побудови цифрових двійників складних систем. Вони надають засоби багатодоменного моделювання, що дає змогу описувати ієрархічні структури, які включають компоненти з різних фізичних доменів, таких як механіка, електроніка, гідравліка або теплові процеси. Для забезпечення можливості виконання обчислень у режимі, наближеному до реального часу, використовуються моделі зниженого порядку, які трансформують детальні тривимірні фізичні моделі у компактні представлення без суттєвої втрати точності. Окремим напрямом розвитку є побудова графів цифрових двійників, що концептуально описують зв’язки між об’єктами, середовищами та суб’єктами взаємодії, наприклад у межах будівель, виробничих майданчиків або міських систем. У таких випадках цифровий двійник набуває форми графової моделі або моделі, що ґрунтується на знаннях, яка може автоматично наповнюватися даними з різних джерел [50, 51] та використовуватися для більш точного відображення реальних умов функціонування системи.

Не менш важливою складовою функціональності платформ цифрових двійників є інтеграція даних та забезпечення зв’язності з фізичними об’єктами. Платформи орієнтовані на тісну взаємодію з промисловими та корпоративними IoT-екосистемами, що дозволяє отримувати потоки даних у реальному часі без необхідності створення спеціалізованої інфраструктури «з нуля». Важливою

особливістю є можливість об’єднання розрізнених джерел даних, таких як давачі, відеокамери, виробничі та управлінські інформаційні системи, без їх фізичного переміщення або дублювання. Це забезпечує формування так званих «живих» цифрових двійників, стан яких постійно синхронізується зі станом фізичного об’єкта за рахунок потоків даних, що надходять через спеціалізовані сервіси зв’язку.

Наступним ключовим блоком функціональності є аналітика та прогнозування. Сучасні платформи цифрових двійників реалізують гібридні підходи до аналізу, поєднуючи фізично обґрунтовані моделі з методами аналізу даних і машинного навчання. Такий підхід дозволяє калібрувати параметри моделей, компенсувати неточності вимірювань і описувати процеси, які складно або неможливо формалізувати суто математично [52]. На практиці це відкриває можливості для реалізації сценаріїв прогнозного обслуговування, коли стан активів аналізується в реальному часі з метою передбачення відмов і зниження витрат на технічне обслуговування [53, 54]. Крім того, цифрові двійники активно використовуються для системної оптимізації шляхом тестування альтернативних сценаріїв експлуатації та аналізу їхнього впливу на продуктивність, надійність і ефективність системи.

Візуалізація та інтерфейс взаємодії є ще одним важливим аспектом сучасних платформ цифрових двійників. Значна увага приділяється створенню імерсійних тривимірних сцен, у яких дані з давачів і результати аналітики відображаються безпосередньо в контексті 3D-моделей обладнання або середовищ. Такий підхід спрощує діагностику, підвищує ситуаційну обізнаність користувачів і зменшує час реагування на аномальні події. Платформи цифрових двійників також мають підтримувати швидке прототипування людино-машинних інтерфейсів, що дає змогу на ранніх етапах проєктування перевіряти вимоги та сценарії взаємодії. Важливою практичною можливістю є експорт веб-застосунків, які дозволяють здійснювати віддалений моніторинг і керування цифровими двійниками через стандартні браузери.

Завершальним етапом життєвого циклу цифрового двійника є його валідація та розгортання. Сучасні платформи підтримують різні підходи до тестування, зокрема інтеграцію вбудованого програмного забезпечення з віртуальними моделями системи, що дозволяє перевіряти роботу керуючих алгоритмів ще до створення фізичних прототипів. Процеси розгортання включають перевірку та верифікацію цифрових двійників перед запуском у хмарних середовищах, на edge-пристроях або в автономному режимі, що є критично важливим для забезпечення надійності та безпеки.

Реалізація зазначених функціональних можливостей висуває низку вимог до програмного забезпечення технології цифрових двійників. На загальносистемному рівні ключовими є універсальність і відкритість архітектури, що дозволяє уникати замкнених екосистем і забезпечувати інтеграцію моделей та даних між різними інструментами. Не менш важливою є легкість розгортання, що передбачає підтримку як офлайн-сценаріїв, так і хмарних рішень із доступом через веб-інтерфейси. Масштабованість і повторне використання компонентів досягаються завдяки використанню API та бібліотек готових моделей, алгоритмів і шаблонів.

2.2. Аналіз вимог до програмного забезпечення технології цифрових двійників

Сформулюємо та проаналізуємо вимоги до програмного забезпечення технології цифрових двійників. Почнемо з функціональних вимог та визначимо функціональні вимоги таких категорій:

1) загальносистемні вимоги;
2) вимоги щодо роботи з даними;
3) вимоги щодо роботи з моделями;
4) вимоги щодо аналітики та прогнозування;
5) вимоги щодо моніторингу та зворотного впливу.

Основні загальносистемні вимоги визначені у табл. 2.1. Розглянемо їх докладно. Вимога FR01/01 є базовою вимогою до платформи цифрових двійників. Вимоги FR01/02 та FR01/03 є взаємодоповнюючими, оскільки режим

обміну даними між фізичним та цифровим об’єктами залежить від підтримуваних платформою типів цифрового двійника: пакетний режим є достатнім для цифрової моделі та цифрової тіні, водночас, обмін даними у реальному часі дає змогу забезпечити режим повноцінного цифрового двійника. Вимога FR01/04 випливає з галузевих стандартів, зокрема, OPC UA (Open Platform Communications Unified Architecture) – стандарту обміну даними в промисловій автоматизації та IoT [47, 48]. Вимога FR01/05 перетворює цифрового двійника на активний компонент та відповідає подієво-орієнтованій архітектурі (Event-Driven Architecture – EDA) [55]. Вимога FR01/06 випливає з моделі BoL–MoL–EoL життєвого циклу цифрового двійника.

Таблиця 2.1 – Загальносистемні функціональні вимоги

Код вимоги	Назва вимоги	Опис вимоги
FR01/01	Представлення фізичного об’єкта	Система повинна забезпечувати цифрове представлення фізичного об’єкта (активу), включно зі станами, параметрами та властивостями
FR01/02	Синхронізація з фізичним об’єктом у пакетному режимі	Система повинна підтримувати синхронізацію даних між фізичним та цифровим об’єктами у пакетному режимі
FR01/03	Синхронізація з фізичним об’єктом у реальному часі	Система повинна підтримувати синхронізацію даних між фізичним та цифровим об’єктами у реальному часі
FR01/04	API та інтероперабельність	Система повинна надавати програмні інтерфейси для інтеграції з іншими системами
FR01/05	Управління подіями та сповіщеннями	Система повинна автоматично виявляти аномалії, генерувати події при перетині порогових значень та ініціювати робочі процеси у зовнішніх системах
FR01/06	Життєвий цикл цифрового двійника	Система повинна підтримувати етапи створення, експлуатації та виведення з експлуатації цифрового двійника

Розглянемо вимоги щодо роботи з даними: табл. 2.2. Вимога FR02/01 є однією з базових вимог, завдяки яким цифрове подання фізичного об’єкта перетворюється з симуляції на цифрового двійника. Вимога FR02/02 є типовою вимогою, виконання якої забезпечує функціонування цифрового двійника, незалежно від типу пристроїв, з яких система отримує дані про фізичного двійника. Вимога FR02/03 є однією з основних вимог, завдяки яким цифрове подання фізичного об’єкта перетворюється з інформаційної панелі (dashboard) на цифрового двійника. Вимога FR02/04 є важливою для аналізу даних та забезпечення етапу EoL життєвого циклу цифрового двійника.

Таблиця 2.2 – Функціональні вимоги щодо роботи з даними

Код вимоги	Назва вимоги	Опис вимоги
FR02/01	Збір даних	Система повинна збирати дані з давачів, IoT-пристроїв та промислових систем
FR02/02	Перетворення даних	Система повинна перетворювати дані з давачів, IoT-пристроїв та промислових систем у внутрішні формати системи
FR02/03	Мультимодаль-на агрегація даних	Система повинна забезпечувати прийом даних з гетерогенних джерел, як-от, потокової телеметрії IoT (Time Series), 3D-моделей (CAD/BIM), статичних атрибутів (метадані), даних з корпоративних систем (ERP, PLM) тощо
FR02/04	Архівування даних	Система повинна забезпечувати збереження історичних даних цифрового двійника

Вимоги щодо роботи з моделями наведені у табл. 2.3. Розглянемо їх докладніше. Вимоги FR03/01 та FR03/02 є основними вимогами, виконання яких забезпечує функціонування цифрового двійника. Вимога FR03/03 узгоджується з вимогою FR04/04 щодо забезпечення гібридної симуляції. Вимога FR03/04 пов’язана з необхідності підтримки стандартів DTDL (Digital Twins Definition Language) [56] та AAS (Asset Administration Shell) [57–59] для уніфікації опису фізичних активів.

Таблиця 2.3 – Функціональні вимоги щодо роботи з моделями

Код вимоги	Назва вимоги	Опис вимоги
FR03/01	Управління новими моделями	Система повинна підтримувати створення та завантаження моделей цифрового двійника
FR03/02	Управління наявними моделями	Система повинна підтримувати оновлення, версіонування та заміну моделей цифрового двійника
FR03/03	Підтримка різних типів моделей	Система повинна підтримувати фізичні, поведінкові та моделі, керовані даними (data-driven)
FR03/04	Семантичне моделювання	Система має надавати інструменти для створення та редагування онтологій, що описують сутності, їхні властивості та взаємозв’язки (топологію) у вигляді графу знань

Вимоги щодо аналітики та прогнозування представлені у табл. 2.4. Вимоги FR04/01, FR04/02, FR04/03 є основними вимогами до платформи цифрового двійника. Виконання вимоги FR04/04 розширює функціональні можливості та діапазон застосування цифрових двійників.

Таблиця 2.4 – Функціональні вимоги щодо аналітики та прогнозування

Код вимоги	Назва вимоги	Опис вимоги
FR04/01	Аналітика	Система повинна виконувати аналіз даних про стан фізичного двійника
FR04/02	Оцінювання	Система повинна виконувати оцінювання поточного стану фізичного двійника
FR04/03	Прогнозування	Система повинна виконувати прогнозування стану фізичного двійника за заданими параметрами
FR04/04	Гібридна симуляція	Система має надавати можливість запуску математичних (фізичних, хімічних) моделей та ML-моделей на основі історичних даних для прогнозування поведінки системи («Що-якщо» аналіз)

Розглянемо вимоги роботи щодо моніторингу та зворотного впливу (табл. 2.5). Вимога FR05/01 є типовою вимогою до платформи цифрового

двійника. Виконання вимоги FR05/02 доповнює вимогу FR05/01 та розширяє можливості платформи цифрового двійника [60]. Вимога FR05/03 також є типовою. Вимоги FR05/04 та FR05/05 є основними вимогами, виконання яких забезпечує функціонування повноцінного цифрового двійника.

Таблиця 2.5 – Функціональні вимоги щодо моніторингу та зворотного впливу

Код вимоги	Назва вимоги	Опис вимоги
FR05/01	Візуалізація стану об’єкта	Система повинна надавати 2D/3D-візуалізацію цифрового двійника та його станів
FR05/02	Візуалізація у AR/VR	Система може мати вебінтерфейс/API для рендерингу просторового контексту об’єкта з накладанням шарів даних телеметрії (наприклад, теплові карти, анімація вузлів тощо)
FR05/03	Інтерфейс взаємодії	Система повинна забезпечувати користувацькі інтерфейси для моніторингу та керування
FR05/04	Синхронізація станів	Система повинна забезпечувати оновлення параметрів віртуальної моделі на основі змін фізичного двійника в реальному часі
FR05/05	Підтримка керуючих дій	Система може (за наявності дозволу) передавати керуючі впливи фізичному двійнику

Нефункціональні вимоги наведені у табл. 2.6. Вимога NFR01 пов’язана з типовою проблемою давачів та пристроїв IoT, а також кіберфізичних систем щодо якості вхідних даних. Вимога NFR02 викликана необхідністю забезпечення того, щоб час між виникненням події у фізичному двійнику та її відображенням у цифровому двійнику не повинен перевищувати критичних значень для конкретного домену. Вимога NFR03 є типовою вимогою до хмарних платформ цифрових двійників. Вимога NFR04 пов’язана з необхідністю забезпечення агностичності до протоколів нижнього рівня (відповідно до ISO 23247) [61]. Вимога NFR05 є типовою вимогою до програмної системи. Вимога NFR06 також є типовою вимогою до платформ цифрових двійників. Вимога NFR07 відповідає стандарту ISO/IEC 25010 [62]. Вимоги NFR08 та NFR09 спрямовані на еволюцію цифрового двійника, що є важливим з огляду на тривалий життєвий цикл двійника (десятиліття), тому система має еволюціонувати разом з фізичним об’єктом. Вимога NFR10 має виконуватися в разі, якщо цифровий двійник

планується використовувати у тому числі для наукових досліджень. Вимога NFR11 є важливою зокрема для забезпечення MoL-фази життєвого циклу цифрового двійника. Вимога NFR12 є типовою вимогою до програмних систем такого класу. Вимога NFR13 є вимогою стандартів та регуляторів [63–66].

Таблиця 2.6 – Нефункціональні вимоги

Код вимоги	Назва вимоги	Опис вимоги
NFR01	Точність та надійність	Система повинна мати механізми валідації вхідних даних, фільтрації шумів та заповнення пропусків, щоб гарантувати достовірність моделі
NFR02	Продуктивність	Система повинна забезпечувати оброблення даних з допустимими затримками
NFR03	Масштабова-ність	Система повинна підтримувати зростання кількості об’єктів, даних та користувачів
NFR04	Інтероперабель-ність	Система повинна взаємодіяти з іншими системами через стандартні протоколи та формати
NFR05	Безпека	Система повинна забезпечувати захист даних, автентифікацію та контроль доступу
NFR06	Цілісність даних	Система повинна забезпечувати узгодженість і достовірність даних
NFR07	Підтримуваність	Система повинна бути придатною до супроводу та оновлення
NFR08	Модульність	Архітектура системи повинна бути модульною
NFR09	Розширюваність	Можливість додавання нових типів активів, моделей симуляції або аналітичних модулів без необхідності зупинки або перекомпіляції ядра системи
NFR10	Відтворюваність	Результати аналізу та симуляцій повинні бути відтворюваними
NFR11	Сумісність із життєвим циклом	Система повинна підтримувати еволюцію моделей і програмного забезпечення без повної зупинки
NFR12	Доступність	Система повинна забезпечувати заданий рівень доступності
NFR13	Аудит та трасування	Система повинна забезпечувати журналювання дій і змін

Подальший аналіз сформульованих вимог має здійснюватися з огляду на галузь застосування, конкретний варіант використання, поставлені перед розробниками задачі та їхню пріоритетність.

2.3. Референтні архітектури платформ цифрового двійника

Архітектуру платформи цифрового двійника слід проєктувати як багатошарову та комбіновану, що включає фізичний шар (актив і давачі), канал зв'язку (IoT, промислові шини), шар збирання та оброблення даних, шар моделей (фізичні/гібридні/data-driven), аналітичні сервіси і інтерфейси «людина-система». У ISO 23247 цифровий двійник описується як інтеграційна сутність із ролями (observable item, digital twin service тощо), які взаємодіють у межах референтної архітектури. Ключовими принципами проєктування архітектури цієї платформи є наступні:

- чітке розділення відповідальностей (data ingestion, model management, analytics, visualization, control);
- слабке зв'язування (loose coupling) між підсистемами через стандартизовані інтерфейси;
- підтримка ієрархії телеконекторів/агрегаторів (edge → fog → cloud) для мінімізації латентності й витрат на мережевий трафік;
- трасування походження даних для верифікації результатів моделювання та аналітики.

На функціональному рівні програмне забезпечення цифрових двійників повинно охоплювати повний набір ключових модулів, що забезпечують управління моделями, оброблення даних, розроблення алгоритмів, IoT-зв'язок, симуляцію та візуалізацію. Ці модулі мають працювати з гетерогенними форматами даних, підтримувати генерацію віртуальних даних у разі їх дефіциту, забезпечувати інкапсуляцію алгоритмів і синхронізацію станів фізичних та цифрових об'єктів у реальному часі.

Важливою особливістю сучасних платформ цифрових двійників є їх орієнтація на різні категорії користувачів. Для дослідників критичною є

наявність інструментів, що дозволяють перевіряти теоретичні гіпотези та досліджувати синергію між різними моделями (геометричні, фізичні, поведінкові тощо) та правилами. Для розробників суттєве значення має підтримка підходів, які прискорюють створення прикладних рішень за рахунок конфігурації готових модулів. Для підприємств, своєю чергою, важливою є можливість гнучкого налаштування та міграції цифрових двійників між різними сценаріями використання, наприклад від простого моніторингу до складної діагностики та оптимізації, що забезпечує довгострокову цінність таких систем.

Розглянемо декілька референтних архітектур програмного забезпечення цифрових двійників, представлених на різних рівнях деталізації.

При розробленні платформ цифрових двійників доцільно спиратися на стандартизовані референтні архітектури. Найбільш авторитетним стандартом на сьогодні є ISO 23247 «Automation systems and integration – Digital twin framework for manufacturing» [61]. Цей стандарт пропонує 4-рівневу архітектуру, яка чітко розмежовує відповідальність компонентів програмної системи:

1) рівень спостережуваних виробничих елементів (Observable Manufacturing Elements), який визначає джерела даних;
2) рівень комунікації пристроїв (Device Communication Entity), який відповідає за збір та нормалізацію даних;
3) рівень цифрового двійника (Digital Twin Entity), який визначає ядро системи, що включає управління даними (збір та зберігання поточного стану) та моделювання (виконання обчислювальних моделей);
4) рівень користувача (User Entity), тобто людино-машинні інтерфейси, AR/VR та зовнішні API.

Розглянемо діаграму пакетів, що відображає цю структуру: рис. 2.1.

На діаграмі зображено такі потоки інформації:

- висхідний потік телеметрії від сенсорів через шлюзи до бази даних двійника для оновлення стану;
- низхідний потік керуючих впливів;

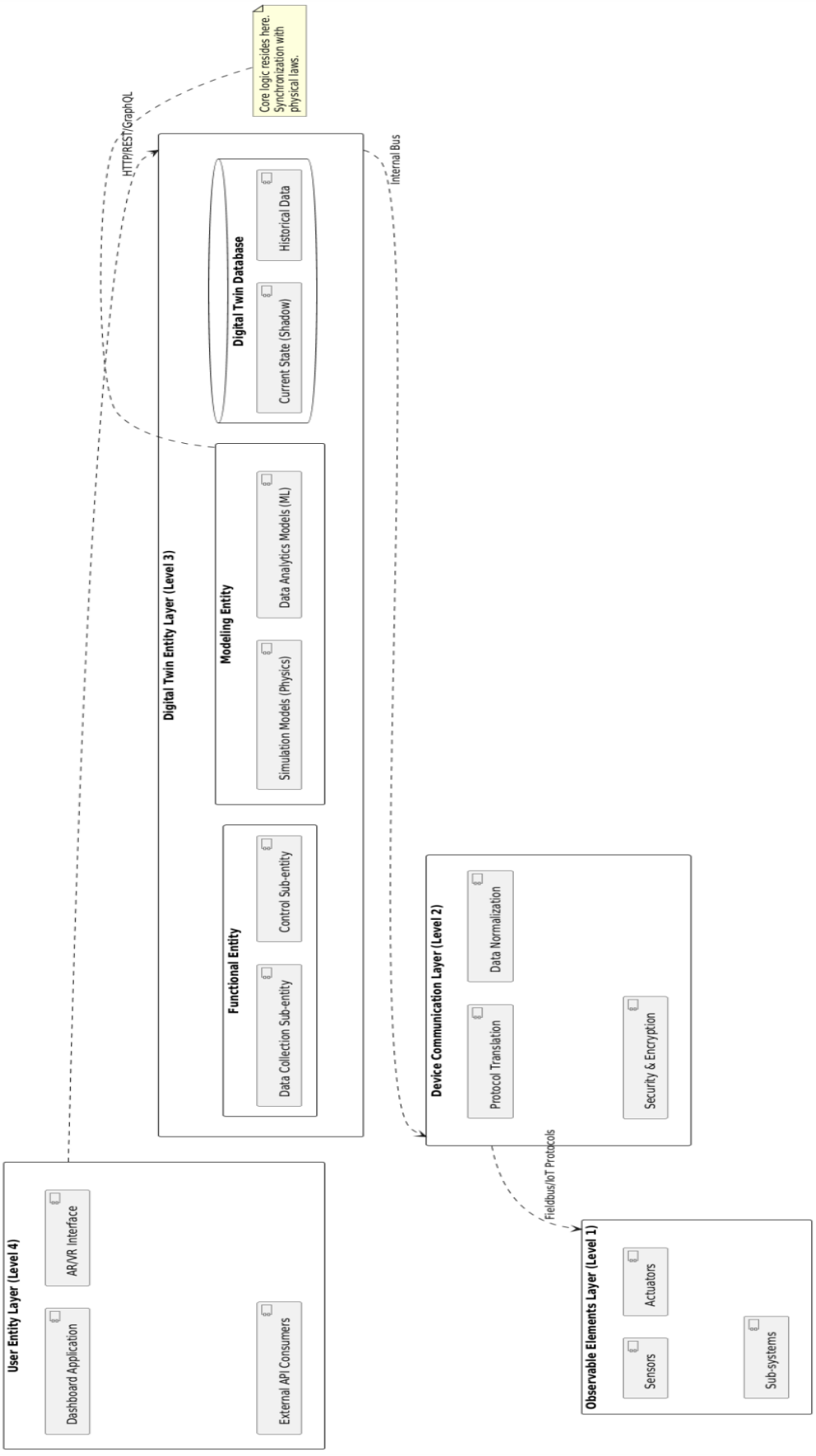

Рис. 2.1. Діаграма пакетів референтної архітектури цифрового двійника на основі стандарту ISO 23247

- зв’язки всередині рівнів для обміну даними між моделями та базами даних всередині ядра двійника.

Розглянемо докладно кожний з рівнів, представлених на діаграмі.

Рівень спостережуваних виробничих елементів (Level 1: Observable Manufacturing Elements – OME) відображає рівень фізичної реальності, до якого належать верстати, маніпулятори, технологічні лінії, давачі, приводи, а також персонал та матеріали. З точки зору програмного забезпечення цей рівень є джерелом подій та даних. На цьому рівні відбувається генерація первинних даних (телеметрії) та виконання фізичних дій (актуація).

Рівень комунікації пристроїв (Level 2: Device Communication Entity – DCE) – це рівень абстракції обладнання, який зменшує складність та оптимізує різноманітність фізичних інтерфейсів від наступних рівнів системи. На цьому рівні здійснюється: опитування пристроїв через промислові протоколи (Modbus, Profinet, EtherCAT); перетворення різнорідних даних в єдиний уніфікований формат (наприклад, всі значення температури приводяться до градусів Цельсія, а структура пакетів до JSON [67] або XML [68]); управління пристроями.

Рівень сутності цифрового двійника (Level 3: Digital Twin Entity – DTE) являє собою ядро архітектури, де розташована основна бізнес-логіка та моделі. Цей рівень поділяється на дві ключові функціональні зони:

- управління даними (Data Collection & Management), що відповідає за підтримку актуального стану двійника, зберігає цифрову тінь, веде архів (Time Series Data) для можливості ретроспективного аналізу;
- моделювання та аналітика (Modeling & Analytics), де виконуються обчислювальні моделі, здійснюється розрахунок параметрів, які неможливо виміряти давачами, та застосовуються алгоритми машинного навчання для прогнозування відмов або оптимізації режимів роботи.

Рівень сутності користувача (Level 4: User Entity – UE) є рівнем забезпечення взаємодії та надавання інформації. Цей рівень звертається до Рівня 3 через API (наприклад, REST або GraphQL) для візуалізації стану фізичного двійника через різноманітні інтерфейси: інформаційні панелі, AR/VR,

тощо. Отримувачами інформації можуть бути як користувачі (оператори, інженери, менеджери), так і інші програмні системи (ERP, MES, Supply Chain Management).

Розглянемо іншу референтну архітектуру платформи цифрового двійника. Ключовими складовими цієї високорівневої архітектури є наступні:

- система фізичного активу (Physical Asset System), яка включає вбудоване програмне забезпечення (Embedded Systems), давачі та контролери (PLC);
- система зв'язку (Connectivity System), що складається з IoT-шлюзів та мережевої інфраструктури (Edge/Fog Computing);
- система управління даними (Data Management System), зокрема, хмарні сховища (Data Lakes), бази даних часових рядів тощо;
- система моделювання (Modeling System), тобто середовища виконання фізичних симуляцій (FEM, CFD) та ML-моделей;
- бізнес-системи (Enterprise Systems), як-от, ERP, PLM, MES системи, які надають контекст.

На рис. 2.2 наведено діаграму розгортання платформи цифрового двійника, який включає зазначені високорівневі компоненти. Ця діаграма демонструє принцип слабкого зв'язування та спеціалізації. По-перше, критичні реакції залишаються на рівні PLC/Edge (наприклад, щоб миттєво зупинити фізичного двійника, як-от, станок, на якому у наступний момент часу може статися збій). По-друге, аналітика даних та зберігання історії винесені в хмарне сховище, де ресурси практично необмежені. І врешті, бізнес-дані залишаються в корпоративних системах (ERP/PLM), а цифровий двійник лише обмінюється з ними інформацією через API. Така архітектура дає змогу створювати масштабовані рішення, де вихід з ладу одного компонента (наприклад, втрата зв'язку з хмарним сховищем) не зупиняє роботу фізичного виробництва (PLC продовжує працювати автономно) [7, 9]. Проаналізуємо рівні системи, представлені на діаграмі розгортання цієї платформи цифрового двійника.

Інфраструктура поділена на чотири ключові зони (вузли).

На рівні фізичного середовища (Physical Environment / Shop Floor) маємо два компоненти: Physical Asset (CNC Machine) та PLC / SCADA. Компонент Physical Asset – це реальний об’єкт (наприклад, станок з ЧПУ), який генерує фізичні сигнали (вібрація, температура, струм). Компонент PLC / SCADA – це програмовані логічні контролери, які керують обладнанням у реальному часі. Для забезпечення зв’язку на цьому рівні звичайно використовуються дротові з’єднання (Analog/Digital I/O), а дані часто є неструктурованими та у деяких випадках аналоговими.

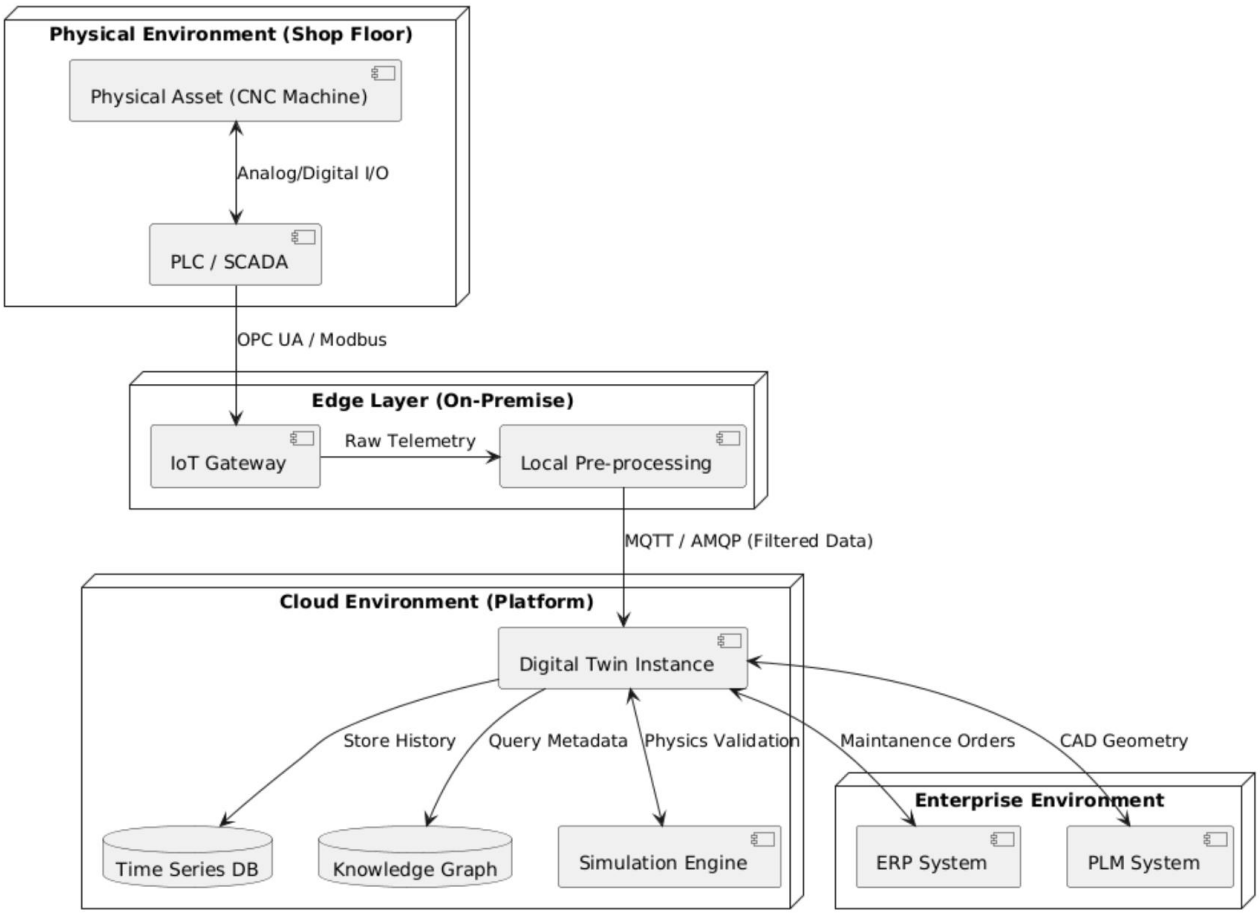

Рис. 2.2. Діаграма розгортання платформи цифрового двійника

Рівень Edge Layer (On-Premise) знаходиться безпосередньо на виробництві, поруч з обладнанням. Він включає компоненти IoT Gateway та Local Pre-processing. Компонент IoT Gateway (шлюз) є апаратно-програмним або програмним компонентом, який підключається до PLC через промислові протоколи (Modbus, Profinet, OPC UA). Він конвертує ці специфічні протоколи у

формат JSON або аналогічний формат. Компонент Local Pre-processing реалізує первинне оброблення даних. Так, цей компонент може оптимізувати кількість даних, які передаються у хмарний ресурс: наприклад, передавати кожну мілісекунду вібрації в хмару є неоптимальним, тому edge-пристрій може обчислювати середнє значення або виявляти піки і передавати у хмарне сховище лише значущі події. У такий спосіб може відбуватися фільтрація та первинне оброблення даних, що надходять від давачів та IoT-пристроїв у реальному часі.

Хмарне середовище (Cloud Environment / Platform) містить реалізацію основної логіки цифрового двійника. Компонент Digital Twin Instance є центральним програмним компонентом цієї реалізації. Він отримує дані від Edge-рівня (через протоколи MQTT або AMQP) і оновлює свій стан. Прикладом може слугувати віртуальна репрезентація конкретного станка. Компонент Time Series DB є спеціалізованою базою даних для зберігання історії показників (часових рядів). Компонент Knowledge Graph (граф знань) зберігає метадані та зв’язки, наприклад, інформацію про те, що певний двигун є частиною саме цього станка і розташований у конкретному цеху. Компонент Simulation Engine – це рушій симуляції, який відтворює поведінку цифрового двійника на основі математичних розрахунків, які неможливо виконати на edge-пристрої на рівні Edge Layer.

Корпоративне середовище (Enterprise Environment) є рівнем реалізації бізнес-логіки. Цей рівень інтегрує цифровий двійник у процеси компанії. Компонент ERP (Enterprise Resource Planning) забезпечує зв’язок платформи цифрового двійника з системою планування ресурсів. Так, наприклад, якщо симуляція, яка виконується у хмарному середовищі, виявить потенційну або передбачувану поломку, то компонент ERP згенерує запит до ERP-системи підприємства на створення наряд-замовлення для ремонтної бригади. Компонент PLM (Product Lifecycle Management) забезпечує зв’язок платформи цифрового двійника з системою управління життєвим циклом виробу, яке є джерелом інженерних 3D-моделей та специфікацій для цифрового двійника.

Розглянемо ще одну референтну архітектуру. На рис. 2.3 наведено діаграму типових компонентів платформи цифрового двійника. На цій діаграмі архітектура цифрового двійника представлена як ланцюг взаємодіючих програмних компонентів, що забезпечують повний цикл оброблення даних – від фізичного об’єкта до аналітики, візуалізації та інтеграції із зовнішніми системами.

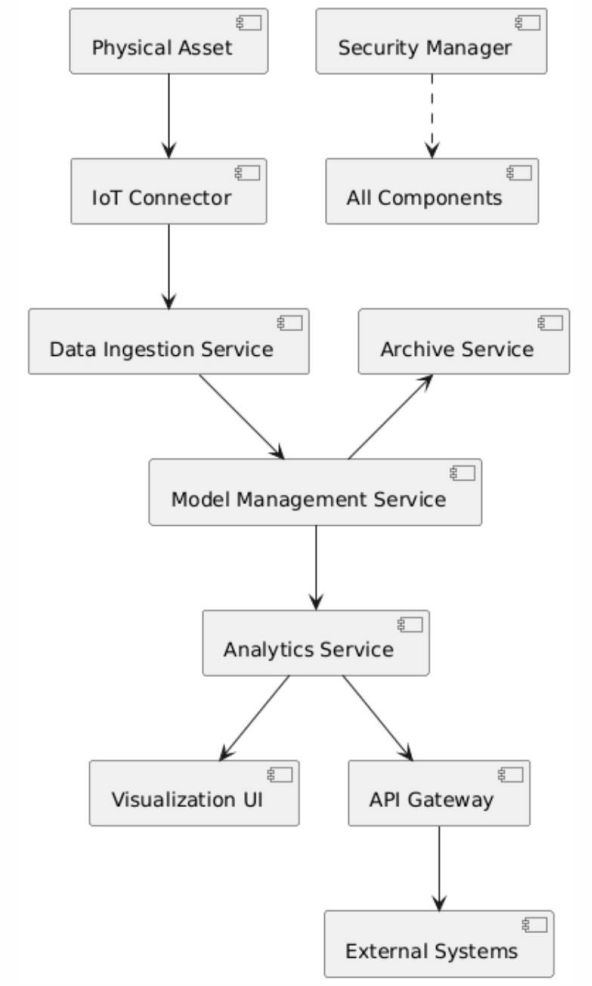

Рис. 2.3. Діаграма компонентів платформи цифрового двійника

Компонент Physical Asset репрезентує фізичний об’єкт, з якого надходять первинні дані. Він не є програмним компонентом у прямому сенсі, проте включений до діаграми як джерело подій і станів, що ініціюють роботу цифрового двійника.

Компонент IoT Connector виконує функцію програмного адаптера між фізичним об’єктом і цифровим середовищем. Його основне призначення полягає

у зборі телеметричних даних та подій через стандартизовані протоколи промислового та IoT-зв'язку (зокрема OPC UA та MQTT). Архітектурно цей компонент ізолює специфіку апаратних інтерфейсів від решти системи, що підвищує портативність і масштабованість платформи.

Компонент Data Ingestion Service відповідає за первинне оброблення потоків даних, які надходять від компонента IoT Connector. У межах цього компонента реалізуються функції фільтрації, нормалізації, агрегування та перетворення даних у внутрішні уніфіковані формати. Важливо, що цей сервіс може працювати як у пакетному, так і в потоковому режимах, що дає змогу підтримувати різні типи цифрових двійників – від цифрової тіні до повноцінного цифрового двійника, який забезпечує синхронізацію з фізичним двійником у режимі реального часу.

Компонент Model Management Service є центральним елементом архітектури платформи цифрового двійника. Він відповідає за керування моделями, що описують поведінку фізичного об'єкта: фізичними, математичними, імітаційними або data-driven моделями. До його функцій належать реєстрація моделей, керування версіями, оновлення параметрів, а також зв'язок між моделями та поточним станом об'єкта. Саме через цей компонент реалізується еволюція цифрового двійника протягом життєвого циклу.

Компонент Analytics Service призначений для виконання обчислювальних і аналітичних задач на основі даних та моделей. У цьому компоненті можуть бути реалізовані алгоритми машинного навчання, прогнозування, діагностики, оптимізації, а також симуляційні сценарії типу «Що якщо?» (what-if). Архітектурне відокремлення аналітики дає змогу незалежно масштабувати обчислювальні ресурси та впроваджувати MLOps-практики без впливу на інші частини системи.

Результати аналітики передаються до компонента Visualization UI, який забезпечує взаємодію з користувачем. Цей компонент відповідає за 2D/3D-візуалізацію станів цифрового двійника, побудову графіків, панелей моніторингу

та, за потреби, імерсійних інтерфейсів. Таким чином, компонент Visualization UI реалізує людино-орієнтований рівень цифрового двійника.

Паралельно компонент Analytics Service взаємодіє з API Gateway, який виконує роль єдиної точки доступу для зовнішніх систем. Через API Gateway цифровий двійник може інтегруватися з корпоративними інформаційними системами (MES, ERP, PLM), іншими цифровими двійниками або зовнішніми сервісами. Такий підхід відповідає принципам сервіс-орієнтованої та мікросервісної архітектури.

Компонент Security Manager логічно пов’язаний з усіма іншими компонентами. Він забезпечує автентифікацію, авторизацію, керування правами доступу та захист каналів обміну даними.

Компонент Archive Service відповідає за довготривале зберігання моделей і даних, зокрема історичних версій та результатів симуляцій. Архітектурно цей компонент пов’язаний з компонентом Model Management Service, оскільки саме моделі та їхня еволюція є ключовими артефактами фази EoL життєвого циклу цифрового двійника. Наявність компонента Archive Service забезпечує відтворюваність досліджень, аудит та формування так званої «цифрової спадщини» об’єкта.

Основним етапом життєвого циклу цифрового двійника є MoL. На рис. 2.4 представлено діаграму послідовності експлуатації цифрового двійника, яка відповідає етапу MoL. Діаграма відображає безпосереднє виконання функціональних вимог FR01/02 (синхронізація з фізичним об’єктом у пакетному режимі), FR01/03 (синхронізація з фізичним об’єктом у реальному часі), FR04/01 (аналітика), FR04/02 (оцінювання), а також нефункціональної вимоги NFR02 (продуктивність).

Сценарій починається з ініціації взаємодії з фізичним об’єктом. Компонент PhysicalAsset надсилає повідомлення sensorData до компонента IoTConnector. Це повідомлення містить поточні значення давачів або інші телеметричні дані, що відображають реальний стан фізичного двійника. Цей етап може реалізовуватися

як періодичне (пакетне) передавання даних або як безперервний потік у режимі реального часу.

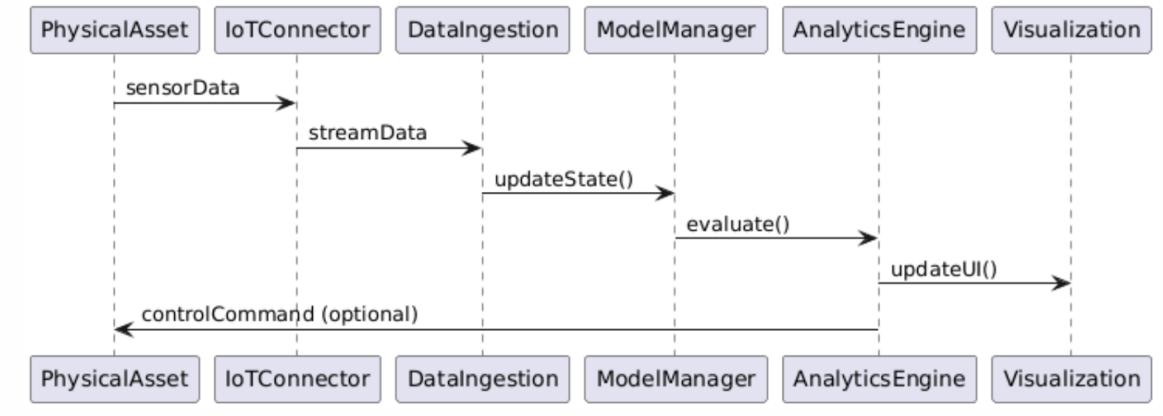

Рис. 2.4. Діаграма послідовності експлуатації цифрового двійника

Після отримання даних компонент IoTConnector передає їх у компонент DataIngestion за допомогою повідомлення streamData. На цьому етапі відбувається відокремлення специфіки протоколів і форматів, пов’язаних із фізичним рівнем, від внутрішнього подання даних у цифровому двійнику. Таким чином реалізується архітектурний принцип слабкого зв’язування між фізичним та цифровим шарами.

Компонент DataIngestion, отримавши потік даних, ініціює оновлення цифрового стану, викликаючи операцію updateState() у компоненті ModelManager. Цей крок є ключовим для підтримання узгодженості між фізичним об’єктом і його цифровим двійником. Компонент ModelManager застосовує отримані дані до відповідних моделей, оновлює параметри та фіксує новий стан цифрового двійника. Після оновлення стану компонент ModelManager ініціює аналітичний етап, передаючи керування до компоненту AnalyticsEngine через повідомлення evaluate(). На цьому етапі виконуються обчислювальні процедури, які можуть включати діагностичний аналіз, прогнозування майбутніх станів, оцінювання ризиків або симуляцію альтернативних сценаріїв. Саме тут реалізується інтелектуальна складова цифрового двійника.

Результати аналітики надсилаються до компонента Visualization у вигляді повідомлення updateUI(). Компонент Visualization інтерпретує отримані дані та оновлює користувацький інтерфейс: наприклад, 3D-модель, графіки часових рядів або панелі моніторингу. Цей етап забезпечує прозорість та інтерпретованість роботи цифрового двійника для користувача платформи.

За потреби компонент AnalyticsEngine може ініціювати повідомлення controlCommand у напрямку компонента PhysicalAsset. Такий зворотний вплив можливий лише за умови, що архітектура цифрового двійника підтримує активне керування фізичним об’єктом, а також дотримані вимоги безпеки та авторизації. У цьому випадку цифровий двійник переходить від ролі пасивного спостерігача до ролі активного учасника керування кіберфізичною системою.

Окремо розглянемо внутрішню організацію узагальненого компонента цифрового двійника (рис. 2.5). Цей компонент є ядром будь-якої програмної системи цифрових двійників, у тому числі програмних платформ, референтні архітектури яких розглянуті вище. Архітектура компонента Digital Twin має відповідати стандарту AAS та повинна підтримувати модульність, оскільки двійник може еволюціонувати (наприклад, шляхом додавання нових моделей симуляції).

Основні програмні модулі цього компонента мають наступне призначення. Рушій синхронізації (Synchronization Engine) порівнює стан фізичного об’єкта (telemetry) зі станом віртуальної моделі. Він відповідає за вирішення конфліктів даних та оновлення цифрової тіні. Оркестратор моделей (Model Orchestrator) керує запуском різних типів моделей (FMU/FMI, Python-скрипти, MATLAB-моделі). Він вирішує, яку модель застосувати для конкретного запиту. Семантичний менеджер (Semantic Manager) забезпечує доступ до онтології (наприклад, DTDL) та валідує відповідність даних схемі. Адаптер інтерфейсу (Interface Adapter) надає стандартизовані API (REST, OPC UA) для зовнішніх модулів та систем.

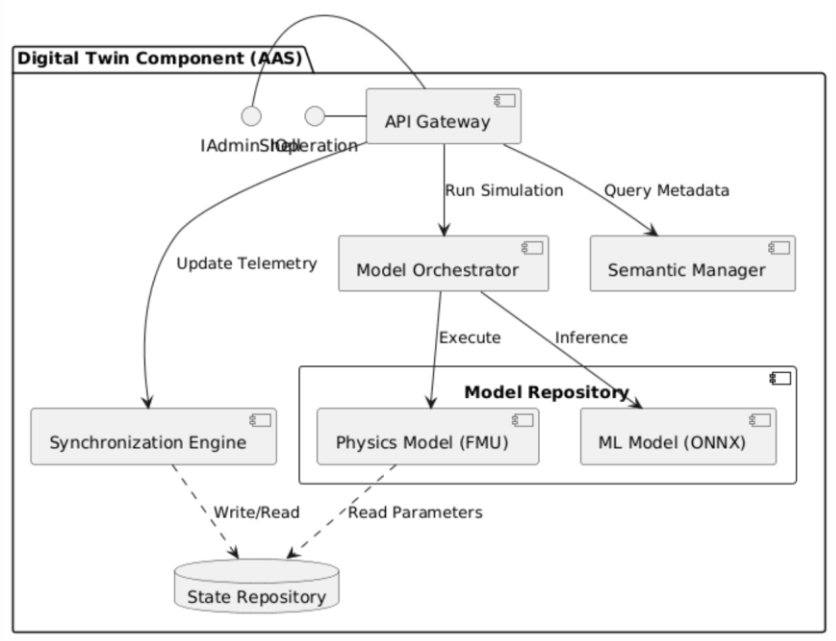

Рис. 2.5. Компонентна діаграма внутрішньої структури цифрового двійника на основі концепції AAS

Завершуючи розгляд референтних архітектур, слід звернути увагу, що при проєктуванні платформ цифрових двійників доцільно керуватися принципами подієво-орієнтованої архітектури у поєднанні з мікросервісами. У першу чергу це стосується організації асинхронної комунікації, коли замість HTTP-запитів використовується шаблон Publish/Subscribe через брокери повідомлень (Apache Kafka, RabbitMQ). Це дозволяє розв’язати проблему «Backpressure», коли потік даних від давачів перевищує швидкість їх оброблення. Також доцільно дотримуватися принципами CQRS (Command Query Responsibility Segregation) і розділяти канали запису (отримання телеметрії) та читання (аналітика, інформаційні панелі). Це критично важливо, оскільки навантаження на запис у двійниках звичайно на порядок вище. Крім того, доцільно застосовувати безсерверні обчислення (AWS Lambda, Azure Functions) для запуску короткоживучих процесів оброблення подій, як-от, наприклад, перевірка порогового значення температури об’єкта.

Загалом, вибір архітектурного стилю для програмного забезпечення цифрових двійників залежить від вимог до латентності, масштабу та складності моделей. Для систем реального часу з жорсткими обмеженнями доцільно використовувати edge-орієнтовані архітектури з мінімізацією хмарних викликів.

Для складних аналітичних систем доцільна хмарна мікросервісна архітектура (Cloud-Native) з використанням EDA [55, 69]. Основою сумісності має бути дотримання стандартів ISO 23247 та IEC 63278 (AAS).

Контрольні запитання

1. У чому полягає принципова відмінність життєвого циклу цифрового двійника від класичного життєвого циклу програмного продукту (SDLC)?
2. Як співвідносяться етапи BoL, MoL та EoL цифрового двійника з фазами розроблення та експлуатації фізичного активу?
3. Чому цифровий двійник не може вважатися «завершеним» програмним продуктом у класичному розумінні?
4. Які ключові функціональні вимоги визначають перехід від цифрової тіні до повноцінного цифрового двійника?
5. У чому полягає роль мультимодальної агрегації даних у програмних системах цифрових двійників?
6. Поясніть відмінності між фізичними, поведінковими та data-driven моделями у складі цифрового двійника.
7. Які завдання вирішують аналітичні та симуляційні модулі цифрового двійника на етапі MoL?
8. Які нефункціональні вимоги є критичними для систем цифрових двійників реального часу і чому?
9. У чому полягає призначення референтних архітектур ISO 23247 та AAS при проєктуванні платформ цифрових двійників?
10. Чому для програмного забезпечення цифрових двійників доцільно застосовувати подієво-орієнтовану та мікросервісну архітектури?

Завдання для самостійної підготовки

1. Підготуйте перелік функціональних вимог до програмного забезпечення цифрового двійника для заданої предметної галузі (виробництво, енергетика, медицина, освіта).

2. Розробіть сценарій MoL-етапу цифрового двійника з описом потоків даних, аналітики та можливого зворотного впливу.
3. Запропонуйте архітектурні рішення для забезпечення масштабованості цифрового двійника при зростанні кількості активів.
4. Проаналізуйте ризики втрати валідності цифрового двійника та запропонуйте механізми їх виявлення й мінімізації.
5. Сформулюйте дослідницьку задачу, пов’язану з еволюцією програмної архітектури цифрового двійника протягом тривалого життєвого циклу фізичного активу.

3. ФОРМАЛЬНІ МЕТОДИ ПРОЄКТУВАННЯ ЦИФРОВИХ ДВІЙНИКІВ

У розділі представлено алгебраїчну систему агрегатів – математичний апарат, який може бути застосований для формальної специфікації моделей цифрових двійників. Розглянуто базові поняття алгебраїчної системи агрегатів, операції над агрегатами, відношення агрегатів. Представлено метод створення мультиобразу для формалізації моделі цифрового двійника та метод синхронізації мультиобразів і шаблони синхронізації для часового узгодження мультимодільних темпоральних даних цифрового двійника.

3.1. Математичний підхід до специфікації моделей цифрових двійників

Для формальної специфікації цифрових двійників фізичних об’єктів потрібен теоретичний апарат [9, 70–72], який забезпечить логіку подання та оброблення темпоральних мультимодальних даних на рівні математичного, алгоритмічного та програмного забезпечення з урахуванням таких властивостей цих даних.

1. *Мультимодальність* – цифровий двійник визначається сукупністю даних різної природи; логіка подання та оброблення даних цифрового двійника залежить від якісного та кількісного складу цієї сукупності даних.
2. *Темпоральність* – елементи сукупності даних цифрового двійника є впорядкованими за часом їх отримання (ґенерації, реєстрації); порядок слідування окремих елементів послідовності даних впливає на результат оброблення всієї сукупності даних цифрового двійника.

Для опису цифрового двійника та ефективного оброблення темпоральних мультимодальних даних потрібні специфічні математичні абстракції та механізми оперування ними. Сформулюємо основні вимоги до математичного апарату для опису та оброблення даних цифрового двійника.

1. Можливість подання сукупності даних як структури взаємопов’язаних елементів.
2. Можливість врахування порядку слідування елементів сукупності даних при виконанні логічних операції над ними.
3. Можливість перевпорядковування елементів сукупності даних.

Цим вимогам задовольняє *алгебраїчна система агрегатів* [9, 70–72]. Розглянемо її основні положення та можливості щодо подання та оброблення темпоральних мультимодальних даних цифрових двійників.

Агрегат A – це впорядкована скінчена сукупність елементів, яка визначається кортежем множин $\{A\}$ та кортежем кортежів елементів $\langle A\rangle$, причому елементи a_i^j кожного кортежу елементів $\langle a_i^j\rangle_{i=1}^{n_j} \in \langle A\rangle$ належать до відповідної множини $M_j \in \{A\}$, $j = [1 \dots N]$, що задає взаємо-однозначний зв'язок між порядком слідування множин та порядком слідування кортежів елементів:

$$A = \left[\!\left[M_j | \langle a_i^j\rangle_{i=1}^{n_j}\right]\!\right]_{j=1}^{N} = [\![\{A\}|\langle A\rangle]\!],$$

де $\{A\}$ – кортеж множин M_j; $\langle A\rangle$ – кортеж кортежів елементів $\langle a_i^j\rangle_{i=1}^{n_j}$; a_i^j – окремий елемент (значення) або складений елемент (кортеж однорідних значень або кортеж різнорідних значень), $a_i^j \in M_j$.

Визначальними рисами агрегату, які відрізняють цю математичну абстракцію від інших, є наступні:

- агрегат є складеним математичним об'єктом, всі компоненти якого є впорядкованими;
- елементами кортежів, з яких складається агрегат, можуть бути окремі значення або кортежі значень, причому кортежі значень можуть складатись як зі значень одного типу, так і зі значень різних типів; при цьому кожен кортеж містить елементи, що належать одній множині.

Згідно з наведеним вище математичним визначенням агрегату елементи першого кортежу належать до першої множини, елементи другого кортежу належать до другої множини тощо. Порядок слідування множин у агрегаті визначає, як будуть виконуватися операції над агрегатом. Множини у кортежі множин можуть повторюватись; це означає, що в агрегат входять декілька кортежів, що складаються з елементів одного типу.

Окремий елемент кортежу може бути чітким значенням a_i, нечітким значенням $\widetilde{a_i}$ або може бути невизначеним _. Відмінність невизначеного елементу від визначених (чіткого та нечіткого значень) полягає у тому, що невизначений елемент не має значення або воно невідомо до певного моменту, який визначається семантично. Кортеж, який містить лише невизначені елементи є *невизначеним* та позначається ⟨_⟩. Кортеж, який не містить жодного є *пустим* та позначається ⟨∅⟩. Відповідно, агрегат, який складається лише з пустих кортежів, називається *пустим*: $A = [\![M_j|\langle\emptyset\rangle]\!]_{j=1}^{N}$.

Агрегат $A_\emptyset$, який не містить жодного компоненту, називається *нуль-агрегатом*: $A_\emptyset = [\![\emptyset|\langle\emptyset\rangle]\!]$. Нуль-агрегат відіграє у АСА роль *нейтрального елементу*.

Агрегат, який складається лише з невизначених кортежів називається *невизначеним*: $A = [\![M_j|\langle\_\rangle]\!]_{j=1}^{N}$. Невизначений агрегат відрізняється від пустого агрегата тим, що його елементи отримують значення при виконанні певної умови. У практичному сенсі це може означати, що дані будуть отримані з наперед визначеного давача (тобто тип даних є відомим), коли цей давач буде увімкнено або з ним буде встановлено зв'язок.

Порядок слідування множин і відповідних їм кортежів в агрегаті є важливим – саме він визначає, як будуть виконуватись операції над агрегатами. Визначимо це через поняття *сумісності* агрегатів.

Агрегати A_1 та A_2 називаються *сумісними* ($A_1 \doteq A_2$), якщо вони мають однакову довжину, а тип та порядок слідування множин в них збігаються, тобто виконуються умови: $\begin{cases} |A_1| = |A_2| \\ \{A_1\} \equiv \{A_2\} \end{cases}$. Наприклад, наступні агрегати є сумісними:

$$A_1 = \left[\!\!\left[M_1, M_2, M_3 | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1^1}, \langle a_{i_2}^1 \rangle_{i_2=1}^{n_2^1}, \langle a_{i_3}^1 \rangle_{i_3=1}^{n_3^1} \right]\!\!\right],$$
$$A_2 = \left[\!\!\left[M_1, M_2, M_3 | \langle a_{i_1}^2 \rangle_{i_1=1}^{n_1^2}, \langle a_{i_2}^2 \rangle_{i_2=1}^{n_2^2}, \langle a_{i_3}^2 \rangle_{i_3=1}^{n_3^2} \right]\!\!\right].$$

Агрегати A_1 та A_2 називаються *квазісумісними* ($A_1 \doteq A_2$) якщо тип та порядок слідування множин в них збігаються частково, при цьому немає вимоги

щодо рівності довжин цих агрегатів, тобто виконуються умови: $\begin{cases} \{A_1\} \not\equiv \{A_2\} \\ \{A_1\} \cap \{A_2\} \neq \emptyset \end{cases}$.

Наприклад, наступні агрегати є квазісумісними:

$$A_1 = \left[\!\left[M_1, M_2, M_3^1 | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1^1}, \langle a_{i_2}^1 \rangle_{i_2=1}^{n_2^1}, \langle a_{i_3}^1 \rangle_{i_3=1}^{n_3^1} \right]\!\right],$$
$$A_2 = \left[\!\left[M_1, M_2, M_3^2 | \langle a_{i_1}^2 \rangle_{i_1=1}^{n_1^2}, \langle a_{i_2}^2 \rangle_{i_2=1}^{n_2^2}, \langle a_{i_3}^2 \rangle_{i_3=1}^{n_3^2} \right]\!\right].$$

Агрегати A_1 та A_2 називаються *несумісними* ($A_1 \triangleq A_2$), якщо тип та порядок слідування множин в них не збігаються, тобто виконується умова $\{A_1\} \cap \{A_2\} = \emptyset$. Наприклад, наступні агрегати є несумісними:

$$A_1 = \left[\!\left[M_1^1, M_2^1, M_3^1 | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1^1}, \langle a_{i_2}^1 \rangle_{i_2=1}^{n_2^1}, \langle a_{i_3}^1 \rangle_{i_3=1}^{n_3^1} \right]\!\right],$$
$$A_2 = \left[\!\left[M_1^2, M_2^2, M_3^2 | \langle a_{i_1}^2 \rangle_{i_1=1}^{n_1^2}, \langle a_{i_2}^2 \rangle_{i_2=1}^{n_2^2}, \langle a_{i_3}^2 \rangle_{i_3=1}^{n_3^2} \right]\!\right].$$

Особливим випадком несумісності є *прихована сумісність*. Агрегати A_1 та A_2 називаються *приховано сумісними*, A_1 $(\div)$ A_2, якщо обидва агрегати мають однаковий набір множин, але їх порядок слідування відрізняється, тобто виконуються умови $\begin{cases} \{A_1\} \not\equiv \{A_2\} \\ |A_1| = |A_2| = N, \text{ де } j = [1, \dots, N],\ k = [1, 2]. \\ \forall M_j \subset \{A_k\}, \end{cases}$ Наприклад, наступні агрегати є приховано сумісними:

$$A_1 = \left[\!\left[M_1, M_2, M_3 | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1^1}, \langle a_{i_2}^1 \rangle_{i_2=1}^{n_2^1}, \langle a_{i_3}^1 \rangle_{i_3=1}^{n_3^1} \right]\!\right],$$
$$A_2 = \left[\!\left[M_2, M_3, M_1 | \langle a_{i_2}^2 \rangle_{i_2=1}^{n_2^2}, \langle a_{i_3}^2 \rangle_{i_3=1}^{n_3^2}, \langle a_{i_1}^2 \rangle_{i_1=1}^{n_1^2} \right]\!\right].$$

Приховано сумісні агрегати можуть бути перетворені на сумісні шляхом застосування до них певних операцій.

В алгебраїчній системі агрегатів визначені такі операції над агрегатами: *логічні операції*, *операції впорядкування* та *арифметичні операції*.

Логічними операціями [9, 70, 72] є наступні: об'єднання (∪), переріз (∩), різниця (\), симетрична різниця (Δ) та виключний переріз (¬).

Об'єднанням двох агрегатів A_1 та A_2 є агрегат B, який містить елементи кортежів, які належать обом агрегатам та впорядковані за таким правилом:

1. Якщо $A_1 \div A_2$ та

$A_1 = [\![M_1, M_2, \dots, M_N | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1^1, b_2^1, \dots, b_m^1 \rangle, \dots, \langle w_1^1, w_2^1, \dots, w_n^1 \rangle]\!]$,

$$A_2 = [\![M_1, M_2, \dots, M_N | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle b_1^2, b_2^2, \dots, b_q^2 \rangle, \dots, \langle w_1^2, w_2^2, \dots, w_p^2 \rangle]\!],$$

то елементи *i*-того кортежу агрегата A_2 додаються у кінець *i*-го кортежу агрегата A_1:

$$B = A_1 \cup A_2 = [\![M_1, M_2, \dots, M_N | \langle a_1^1, a_2^1, \dots, a_l^1, a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle b_1^1, b_2^1, \dots, b_m^1, b_1^2, b_2^2, \dots, b_q^2 \rangle, \dots, \langle w_1^1, w_2^1, \dots, w_n^1, w_1^2, w_2^2, \dots, w_p^2 \rangle]\!].$$

2. Якщо $A_1 \triangleq A_2$ та

$$A_1 = [\![M_1^1, M_2^1, \dots, M_N^1 | \langle a_1, a_2, \dots, a_l \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle]\!],$$

$$A_2 = [\![M_1^2, M_2^2, \dots, M_K^2 | \langle c_1, c_2, \dots, c_r \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!],$$

то кортеж кортежів агрегата A_2 додається у кінець кортежу кортежів агрегата A_1 та кортеж множин агрегата A_2 додається у кінець кортежу множин агрегата A_1:

$$B = A_1 \cup A_2 = [\![M_1^1, M_2^1, \dots, M_N^1, M_1^2, M_2^2, \dots, M_K^2 | \langle a_1, a_2, \dots, a_l \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle, \langle c_1, c_2, \dots, c_r \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!].$$

3. Якщо $A_1 \doteq A_2$ та

$$A_1 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1 | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle f_1^1, f_2^1, \dots, f_t^1 \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle]\!],$$

$$A_2 = [\![M_1, M_2^2, \dots, M_x, \dots, M_K^2 | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle f_1^2, f_2^2, \dots, f_v^2 \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!],$$

то:

а) елементи *i*-того кортежу агрегата A_2 додаються у кінець *i*-го кортежу агрегата A_1, якщо елементи цих кортежів належать до тієї ж самої *i*-тої множини;

б) для всіх *i*-тих кортежів, елементи яких належать до різних множин, кортеж кортежів агрегата A_2 додається у кінець кортежу кортежів агрегата A_1 та кортеж множин агрегата A_2 додається у кінець кортежу множин агрегата A_1, за винятком кортежів, які підпадають під дію правила а) та виключаються з кортежу кортежів:

$$B = A_1 \cup A_2 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1, M_2^2, \dots, M_K^2 | \langle a_1^1, a_2^1, \dots, a_l^1, a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle f_1^1, f_2^1, \dots, f_t^1, f_1^2, f_2^2, \dots, f_v^2 \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!].$$

Перерізом двох агрегатів A_1 та A_2 є агрегат B, який містить компоненти, які є спільними для цих агрегатів, впорядковані за таким правилом:

1. Якщо $A_1 \doteqdot A_2$ та

$A_1 = [\![M_1, M_2, \dots, M_N | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1^1, b_2^1, \dots, b_m^1 \rangle, \dots, \langle w_1^1, w_2^1, \dots, w_n^1 \rangle]\!]$,

$A_2 = [\![M_1, M_2, \dots, M_N | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle b_1^2, b_2^2, \dots, b_q^2 \rangle, \dots, \langle w_1^2, w_2^2, \dots, w_p^2 \rangle]\!]$,

то *i*-тий кортеж агрегата *B* містить спільні елементи *i*-тих кортежів обох агрегатів зі збереженням порядку слідування цих елементів:

$$B = A_1 \cap A_2 = [\![M_1, M_2, \dots, M_N | \langle a_{l_1}^1, \dots, a_{l_\alpha}^1, a_{r_1}^2, \dots, a_{r_\beta}^2 \rangle,$$
$$\langle b_{m_1}^1, \dots, b_{m_\gamma}^1, b_{q_1}^2, \dots, b_{q_\delta}^2 \rangle, \dots, \langle w_{n_1}^1, \dots, w_{n_\lambda}^1, w_{p_1}^2, \dots, w_{p_\mu}^2 \rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \in \overline{a^2}, i \in \langle 1, \dots, \alpha \rangle$; $a_{r_j}^2 \in \overline{a^1}, a_{r_j}^2 \in \overline{a^2}, j \in \langle 1, \dots, \beta \rangle$;

$b_{m_k}^1 \in \overline{b^1}, b_{m_k}^1 \in \overline{b^2}, k \in \langle 1, \dots, \gamma \rangle$; $b_{q_s}^2 \in \overline{b^1}, b_{q_s}^2 \in \overline{b^2}, s \in \langle 1, \dots, \delta \rangle$;

$w_{n_u}^1 \in \overline{w^1}, w_{n_u}^1 \in \overline{w^2}, u \in \langle 1, \dots, \lambda \rangle$; $w_{p_y}^2 \in \overline{w^1}, w_{p_y}^2 \in \overline{w^2}, y \in \langle 1, \dots, \mu \rangle$.

Це означає наступне. Будемо проходити по елементах кортежу $\overline{a^1} = \langle a_1^1, a_2^1, \dots, a_l^1 \rangle$ від його першого елементу a_1^1 до останнього елементу a_l^1 та порівнювати кожен наступний елемент $a_{l_i}^1$ ($1 \le l_i \le l$, де $l = |\overline{a^1}|$) з елементами кортежу $\overline{a^2} = \langle a_1^2, a_2^2, \dots, a_r^2 \rangle$. Тоді $a_{l_1}^1$ – це перший елемент кортежу $\overline{a^1}$, який також присутній у кортежу $\overline{a^2}$, та $a_{l_\alpha}^1$ – це останній елемент кортежу $\overline{a^1}$, який також присутній у кортежу $\overline{a^2}$. Аналогічно, будемо проходити по елементах кортежу $\overline{a^2}$ від його першого елементу a_1^2 до останнього елементуa_r^2 та порівнювати кожен наступний елемент $a_{r_j}^2$ ($1 \le r_j \le r$, де $r = |\overline{a^2}|$) з елементами кортежу$\overline{a^1}$. Тоді $a_{r_1}^2$ – це перший елемент кортежу $\overline{a^2}$, який також присутній у кортежу $\overline{a^1}$, та $a_{r_\beta}^2$ – це останній елемент кортежу $\overline{a^2}$, який також присутній у кортежу $\overline{a^1}$. Отже, результатом перерізу кортежів $\overline{a^1}$ та $\overline{a^2}$ є кортеж $\langle a_{l_1}^1 \dots a_{l_\alpha}^1, a_{r_1}^2 \dots a_{r_\beta}^2 \rangle$.

Наприклад, якщо $\overline{a^1} = \langle 9,5, \mathbf{2}, \mathbf{7}, 9, \mathbf{3}, 6 \rangle$ та $\overline{a^2} = \langle 1, \mathbf{3}, \mathbf{2}, 4, \mathbf{7}, \mathbf{3}, 8 \rangle$, то $a_{l_1}^1 = \mathbf{2}$ та $a_{l_\alpha}^1 = \mathbf{3}$. Водночас, $a_{r_1}^2 = \mathbf{3}$ та $a_{r_\beta}^2 = \mathbf{3}$. Отже, $\overline{a^1} \cap \overline{a^2} = \langle \mathbf{2}, \mathbf{7}, \mathbf{3}, \mathbf{3}, \mathbf{2}, \mathbf{7}, \mathbf{3} \rangle$.

2. Якщо $A_1 \triangleq A_2$, то $B = A_1 \cap A_2 = [\![\emptyset|\langle\emptyset\rangle]\!] = A_\emptyset$.

3. Якщо $A_1 \doteq A_2$ та

$$A_1 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1|\langle a_1^1, a_2^1, \dots, a_l^1\rangle, \langle b_1, b_2, \dots, b_m\rangle, \dots, \langle f_1^1, f_2^1, \dots, f_t^1\rangle, \dots, \langle w_1, w_2, \dots, w_n\rangle]\!],$$

$$A_2 = [\![M_1, M_2^2, \dots, M_x, \dots, M_K^2|\langle a_1^2, a_2^2, \dots, a_r^2\rangle, \langle d_1, d_2, \dots, d_q\rangle, \dots, \langle f_1^2, f_2^2, \dots, f_v^2\rangle, \dots, \langle z_1, z_2, \dots, z_p\rangle]\!],$$

то агрегат B містить спільні елементи обох агрегатів лише для кортежів, які є сумісними; несумісні кортежі відкидаються, отже, кількість кортежів скорочується:

$$B = A_1 \cap A_2 = [\![M_1, \dots, M_x|\langle a_{l_1}^1, \dots, a_{l_\alpha}^1, a_{r_1}^2, \dots, a_{r_\beta}^2\rangle, \dots, \langle f_{t_1}^1, \dots, f_{t_\rho}^1, f_{v_1}^2, \dots, f_{v_\omega}^2\rangle]\!]$$

Різницею агрегатів A_1 та A_2 є агрегат B, який містить компоненти першого агрегата, яких немає у другому агрегаті, впорядковані за таким правилом:

1. Якщо $A_1 \fallingdotseq A_2$ та

$A_1 = [\![M_1, M_2, \dots, M_N|\langle a_1^1, a_2^1, \dots, a_l^1\rangle, \langle b_1^1, b_2^1, \dots, b_m^1\rangle, \dots, \langle w_1^1, w_2^1, \dots, w_n^1\rangle]\!]$,

$A_2 = [\![M_1, M_2, \dots, M_N|\langle a_1^2, a_2^2, \dots, a_r^2\rangle, \langle b_1^2, b_2^2, \dots, b_q^2\rangle, \dots, \langle w_1^2, w_2^2, \dots, w_p^2\rangle]\!]$,

то i-тий кортеж агрегата B містить елементи i-того кортежу агрегата A_1, які відсутні у i-тому кортежі агрегата A_2:

$$B = A_1 \backslash A_2 = [\![M_1, M_2, \dots, M_N|\langle a_{l_1}^1, \dots, a_{l_\alpha}^1\rangle, \langle b_{m_1}^1, \dots, b_{m_\gamma}^1\rangle, \dots, \langle w_{n_1}^1, \dots, w_{n_\lambda}^1\rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \notin \overline{a^2}, i \in \langle 1, \dots, \alpha\rangle$; $b_{m_k}^1 \in \overline{b^1}, b_{m_k}^1 \notin \overline{b^2}, k \in \langle 1, \dots, \gamma\rangle$;

$w_{n_u}^1 \in \overline{w^1}, w_{n_u}^1 \notin \overline{w^2}, u \in \langle 1, \dots, \lambda\rangle$.

2. Якщо $A_1 \triangleq A_2$, то $B = A_1 \backslash A_2 = A_1$.

3. Якщо $A_1 \doteq A_2$ та

$$A_1 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1|\langle a_1^1, a_2^1, \dots, a_l^1\rangle, \langle b_1, b_2, \dots, b_m\rangle, \dots, \langle f_1^1, f_2^1, \dots, f_t^1\rangle, \dots, \langle w_1, w_2, \dots, w_n\rangle]\!],$$

$$A_2 = [\![M_1, M_2^2, \dots, M_x, \dots, M_K^2 | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle f_1^2, f_2^2, \dots, f_v^2 \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!],$$

то *i*-тий кортеж агрегата B містить лише елементи *i*-того кортежу агрегата A_1, які відсутні у *i*-тому кортежі агрегата A_2, якщо цей *i*-тий кортеж є сумісним для обох агрегатів, та містить всі елементи *i*-того кортежу агрегата A_1, якщо *i*-тий кортеж є несумісним:

$$B = A_1 \backslash A_2 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1 | \langle a_{i_1}^1, \dots, a_{i_\alpha}^1 \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle f_{t_1}^1, \dots, f_{t_\rho}^1 \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \notin \overline{a^2}, i \in \langle 1, \dots, \alpha \rangle; f_{t_e}^1 \in \overline{f^1}, f_{t_e}^1 \notin \overline{f^2}, e \in \langle 1, \dots, \rho \rangle.$

Симетричною різницею агрегатів A_1 та A_2 є агрегат B, який містить компоненти першого агрегата, яких немає у другому агрегаті, та компоненти другого агрегата, яких немає у першому агрегаті, впорядковані за таким правилом:

1. Якщо $A_1 \doteqdot A_2$ та

$A_1 = [\![M_1, M_2, \dots, M_N | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1^1, b_2^1, \dots, b_m^1 \rangle, \dots, \langle w_1^1, w_2^1, \dots, w_n^1 \rangle]\!],$

$A_2 = [\![M_1, M_2, \dots, M_N | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle b_1^2, b_2^2, \dots, b_q^2 \rangle, \dots, \langle w_1^2, w_2^2, \dots, w_p^2 \rangle]\!],$

то *i*-тий кортеж агрегата B містить елементи *i*-того кортежу агрегата A_1, які відсутні у *i*-тому кортежі агрегата A_2, та елементи *i*-того кортежу агрегата A_2, які відсутні у *i*-тому кортежі агрегата A_1:

$$B = A_1 \Delta A_2 = [\![M_1, M_2, \dots, M_N | \langle a_{l_1}^1, \dots, a_{l_\alpha}^1, a_{r_1}^2, \dots, a_{r_\beta}^2 \rangle, \langle b_{m_1}^1, \dots, b_{m_\gamma}^1, b_{q_1}^2, \dots, b_{q_\delta}^2 \rangle, \dots, \langle w_{n_1}^1, \dots, w_{n_\lambda}^1, w_{p_1}^2, \dots, w_{p_\mu}^2 \rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \notin \overline{a^2}, i \in \langle 1, \dots, \alpha \rangle; a_{r_j}^2 \notin \overline{a^1}, a_{r_j}^2 \in \overline{a^2}, j \in \langle 1, \dots, \beta \rangle;$

$b_{m_k}^1 \in \overline{b^1}, b_{m_k}^1 \notin \overline{b^2}, k \in \langle 1, \dots, \gamma \rangle; b_{q_s}^2 \notin \overline{b^1}, b_{q_s}^2 \in \overline{b^2}, s \in \langle 1, \dots, \delta \rangle;$

$w_{n_u}^1 \in \overline{w^1}, w_{n_u}^1 \notin \overline{w^2}, u \in \langle 1, \dots, \lambda \rangle; w_{p_y}^2 \notin \overline{w^1}, w_{p_y}^2 \in \overline{w^2}, y \in \langle 1, \dots, \mu \rangle.$

2. Якщо $A_1 \triangleq A_2$ та

$A_1 = [\![M_1^1, M_2^1, \dots, M_N^1 | \langle a_1, a_2, \dots, a_l \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle]\!],$

$A_2 = [\![M_1^2, M_2^2, \dots, M_K^2 | \langle c_1, c_2, \dots, c_r \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!]$,

то агрегат B дорівнює результату об'єднання агрегатів A_1 та A_2:

$$B = A_1 \Delta A_2 = [\![M_1^1, M_2^1, \dots, M_N^1, M_1^2, M_2^2, \dots, M_K^2 | \langle a_1, a_2, \dots, a_l \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle, \langle c_1, c_2, \dots, c_r \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!] = A_1 \cup A_2.$$

3. Якщо $A_1 \doteq A_2$ та

$$A_1 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1 | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle f_1^1, f_2^1, \dots, f_t^1 \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle]\!],$$

$$A_2 = [\![M_1, M_2^2, \dots, M_x, \dots, M_K^2 | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle f_1^2, f_2^2, \dots, f_v^2 \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!],$$

то i-тий кортеж агрегата B містить лише елементи i-того кортежу агрегата A_1, які відсутні у i-тому кортежі агрегата A_2, та елементи i-того кортежу агрегата A_2, які відсутні у i-тому кортежі агрегата A_1, якщо цей i-тий кортеж є сумісним для обох агрегатів, та містить всі елементи i-того кортежу агрегата A_1 та всі елементи i-того кортежу агрегата A_2, якщо i-тий кортеж є несумісним:

$$B = A_1 \Delta A_2 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1, M_2^2, \dots, M_K^2 |, \langle a_{l_1}^1, \dots, a_{l_\alpha}^1, a_{r_1}^2, \dots, a_{r_\beta}^2 \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle f_{t_1}^1, \dots, f_{t_\rho}^1, f_{v_1}^2, \dots, f_{v_\omega}^2 \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle, \langle d_1, d_2, \dots, d_q \rangle, \dots, \langle z_1, z_2, \dots, z_p \rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \notin \overline{a^2}, i \in \langle 1, \dots, \alpha \rangle; a_{r_j}^2 \notin \overline{a^1}, a_{r_j}^2 \in \overline{a^2}, j \in \langle 1, \dots, \beta \rangle;$

$f_{t_e}^1 \in \overline{f^1}, f_{t_e}^1 \notin \overline{f^2}, e \in \langle 1, \dots, \rho \rangle; f_{v_h}^2 \notin \overline{f^1}, f_{v_h}^2 \in \overline{f^2}, h \in \langle 1, \dots, \omega \rangle.$

Виключним перерізом двох агрегатів A_1 та A_2 є агрегат B, який містить лише компоненти агрегата A_1, які є спільними для цих агрегатів, впорядковані за таким правилом:

1. Якщо $A_1 \doteqdot A_2$ та

$A_1 = [\![M_1, M_2, \dots, M_N | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1^1, b_2^1, \dots, b_m^1 \rangle, \dots, \langle w_1^1, w_2^1, \dots, w_n^1 \rangle]\!],$

$A_2 = [\![M_1, M_2, \dots, M_N | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \langle b_1^2, b_2^2, \dots, b_q^2 \rangle, \dots, \langle w_1^2, w_2^2, \dots, w_p^2 \rangle]\!],$

то i-тий кортеж агрегата B містить елементи i-того кортежу агрегата A_1, які також містяться у i-тому кортежі агрегата A_2:

$$B = A_1 \neg A_2 = [\![M_1, M_2, \dots, M_N | \langle a_{l_1}^1, \dots, a_{l_\alpha}^1 \rangle, \left\langle b_{m_1}^1, \dots, b_{m_\gamma}^1 \right\rangle, \dots, \langle w_{n_1}^1, \dots, w_{n_\lambda}^1 \rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \in \overline{a^2}, i \in \langle 1, \dots, \alpha \rangle; b_{m_k}^1 \in \overline{b^1}, b_{m_k}^1 \in \overline{b^2}, k \in \langle 1, \dots, \gamma \rangle;$

$w_{n_u}^1 \in \overline{w^1}, w_{n_u}^1 \in \overline{w^2}, u \in \langle 1, \dots, \lambda \rangle.$

2. Якщо $A_1 \triangleq A_2$, то $B = A_1 \neg A_2 = [\![\emptyset | \langle \emptyset \rangle]\!] = A_\emptyset$.

3. Якщо $A_1 \doteq A_2$ та

$$A_1 = [\![M_1, M_2^1, \dots, M_x, \dots, M_N^1 | \langle a_1^1, a_2^1, \dots, a_l^1 \rangle, \langle b_1, b_2, \dots, b_m \rangle, \dots, \langle f_1^1, f_2^1, \dots, f_t^1 \rangle, \dots, \langle w_1, w_2, \dots, w_n \rangle]\!],$$

$$A_2 = [\![M_1, M_2^2, \dots, M_x, \dots, M_K^2 | \langle a_1^2, a_2^2, \dots, a_r^2 \rangle, \left\langle d_1, d_2, \dots, d_q \right\rangle, \dots, \langle f_1^2, f_2^2, \dots, f_v^2 \rangle, \dots, \left\langle z_1, z_2, \dots, z_p \right\rangle]\!],$$

то *i*-тий кортеж агрегата B містить елементи *i*-того кортежу агрегата A_1, які також містяться у *i*-тому кортежі агрегата A_2, якщо цей *i*-тий кортеж є сумісним для обох агрегатів, отже, кількість кортежів скорочується:

$$B = A_1 \neg A_2 = [\![M_1, \dots, M_x | \langle a_{l_1}^1, \dots, a_{l_\alpha}^1 \rangle, \dots, \left\langle f_{t_1}^1, \dots, f_{t_\rho}^1 \right\rangle]\!],$$

де $a_{l_i}^1 \in \overline{a^1}, a_{l_i}^1 \in \overline{a^2}, i \in \langle 1, \dots, \alpha \rangle; f_{t_e}^1 \in \overline{f^1}, f_{t_e}^1 \in \overline{f^2}, e \in \langle 1, \dots, \rho \rangle.$

Відмінність між операціями *виключного перерізу* та *перерізу* полягає у тому, що результатом *перерізу* агрегатів A_1 та A_2 є агрегат, який включає спільні компоненти обох агрегатів. Водночас, результатом *виключного перерізу* є агрегат, який містить лише компоненти агрегата A_1, які присутні у агрегаті A_2, проте він не включає компоненти агрегата A_2. Наприклад, якщо агрегати A_1 та A_2 є сумісними та мають вигляд:

$$A_1 = [\![M_t, M_{hr} | \langle 36.4,36.1,36.3,36.2,36.5,36.3 \rangle, \langle 75,76,74,73,75,75 \rangle]\!],$$

$$A_2 = [\![M_t, M_{hr} | \langle 36.5,36.5,36.8,36.6,36.3,36.4,37.0,36.5 \rangle, \langle 74,81,76,93,97, 97,96 \rangle]\!].$$

Тоді, результатом їхнього *перерізу* є агрегат:

$$A_1 \cap A_2 = [\![M_t, M_{hr} | \langle 36.4,36.3,36.5,36.3,36.5,36.5,36.3,36.4,36.5 \rangle, \langle 76,74,[\![74,76 \rangle]\!].$$

Водночас, результатом *виключного перерізу* є агрегат:

$$A_1 \neg A_2 = [\![M_t, M_{hr} | \langle 36.4,36.3,36.5,36.3 \rangle, \langle 76,74 \rangle]\!].$$

Розглянемо агрегат A_3, який є квазісумісним з агрегатом A_1 та має вигляд:

$$A_3 = [\![M_t, M_{sp}|\langle 36.5,36.5,36.8,36.6,36.3,36.4,37.0,36.5\rangle, \langle 177,159,174,155,167, 150,177,135\rangle]\!].$$

Тоді, результатом *перерізу* агрегатів A_1 та A_3 є агрегат:

$$A_1 \cap A_3 = [\![M_t|\langle 36.4,36.3,36.5,36.3,36.5,36.5,36.3,36.4,36.5\rangle]\!],$$

а результатом *виключного перерізу* цих агрегатів є агрегат:

$$A_1 \neg A_3 = [\![M_t|\langle 36.4,36.3,36.5,36.3\rangle]\!].$$

Логічні операції в АСА є некомутативними, оскільки порядок слідування компонентів є важливим. Це відрізняє логічні операції над агрегатами від логічних операцій над множинами.

Операціями впорядкування є розміщення, сортування, проріджування, видалення та вставлення [9, 70, 71].

Операція *розміщення* перевпорядковує кортеж множин та кортеж кортежів агрегата A відповідно до порядку слідування кортежу множин та кортежу кортежів шаблонного агрегата A_{tem}. За шаблонний агрегат може бути взятий невизначений, пустий або будь-який довільний агрегат.

Нехай агрегат A визначений наступним чином:

$$A = \left[\!\!\left[M_3, M_1, M_2, \dots, M_N|\left\langle a_{i_3}^3\right\rangle_{i_3=1}^{n_3}, \left\langle a_{i_1}^1\right\rangle_{i_1=1}^{n_1}, \left\langle a_{i_2}^2\right\rangle_{i_2=1}^{n_2}, \dots, \left\langle a_{i_N}^N\right\rangle_{i_N=1}^{n_N}\right]\!\!\right]$$

та нехай шаблонний агрегат A_{tem} має вигляд $A_{tem} = [\![M_1, M_2, M_3, \dots, M_N|\langle \_\rangle]\!]$.

Тоді результатом операції *розміщення* над агрегатом A за зразком агрегата A_{tem} є агрегат B виду:

$$B = A \vDash A_{tem} = [\![\{A_{tem}\}|\langle A\rangle]\!] = \left[\!\!\left[M_1, M_2, M_3, \dots, M_N|\left\langle a_{i_1}^1\right\rangle_{i_1=1}^{n_1}, \left\langle a_{i_2}^2\right\rangle_{i_2=1}^{n_2}, \left\langle a_{i_3}^3\right\rangle_{i_3=1}^{n_3}, \dots, \left\langle a_{i_N}^N\right\rangle_{i_N=1}^{n_N}\right]\!\!\right]$$

Операція *розміщення* може бути застосована до двох довільних агрегатів A_1 та A_2, де агрегат A_2 використовується як шаблон. Результат операції *розміщення* залежить від сумісності агрегатів.

Якщо $A_1 \doteqdot A_2$, то результатом операції *розміщення* є агрегат B, що дорівнює агрегату A_1 $B = A_1 \vDash A_2 = [\![\{A_2\}|\langle A_1\rangle]\!] = [\![\{A_1\}|\langle A_1\rangle]\!] = A_1$.

Якщо $A_1 \doteq A_2$ та немає прихованої сумісності агрегатів A_1 та A_2, то результатом застосування операції *розміщення* є пустий агрегат $B = A_1 \vDash A_2 = [\![\{A_2\}|\langle\emptyset\rangle]\!]$.

Якщо $A_1 \triangleq A_2$ та немає прихованої сумісності агрегатів A_1 та A_2, то результатом застосування операції *розміщення* є агрегат B виду:

$B = A_1 \vDash A_2 = \left[\!\!\left[M_j^2|\left\langle a_i^j\right\rangle_{i=1}^{n_j}\right]\!\!\right]_{j=1}^{N}$, де $\left\langle a_i^j\right\rangle_{i=1}^{n_j} \in M_j^2$, $M_j^2 \subset \{A_2\}$, $\left\langle a_i^j\right\rangle_{i=1}^{n_j} \subset \langle A_1\rangle$, $\langle B\rangle \not\equiv \langle A_1\rangle$.

Якщо $A_1(\div)\, A_2$, то результатом застосування операції *розміщення* є агрегат B виду: $B = A_1 \vDash A_2 = \left[\!\!\left[\{A_2\}|\left\langle a_i^j\right\rangle_{i=1}^{n_j}\right]\!\!\right]_{j=1}^{N}$, де $\left\langle a_i^j\right\rangle_{i=1}^{n_j} \in \langle A_1\rangle$, $\langle B\rangle \equiv \langle A_1\rangle$.

Практичне значення застосування операції *розміщення* над деякими приховано сумісними агрегатами A_1 та A_2, полягає у тому, що компоненти одного агрегата (A_1) може бути перевпорядкований згідно з порядком слідування компонентів другого агрегата (A_2), отже, в результаті застосування операції *розміщення* приховано сумісні агрегати стають сумісними.

Розглянемо агрегати A_1, A_2 та A_3, такі що $A_1 \triangleq A_2$, $A_2(\div)\, A_3$ та:

$A_1 = [\![M_t, M_{hr}, M_{sp}|\langle 36.4,36.1,36.3,36.2,36.5,36.3\rangle, \langle 75,76,74,73,75,75\rangle, \langle 163,161,164,165,168\rangle]\!]$,

$A_2 = [\![M_t, M_{sp}|\langle 36.5,36.5,36.8,36.6,36.3,36.4,37.0,36.5\rangle, \langle 171,183,175,183, 167,167,176\rangle]\!]$

$A_3 = [\![M_{sp}, M_t|\langle 177,159,174,155,167,150,177,135\rangle, \langle 37.5,37.2,36.8, 37.0,36.6\rangle]\!]$.

Тоді в результаті застосування операції *розміщення* отримуємо:

$A_1 \vDash A_2 = [\![M_t, M_{sp} \mid \langle 36.4,36.1,36.3,36.2,36.5,36.3\rangle, \langle 163,161,164, 165,168\rangle]\!]$,

$A_3 \vDash A_2 = [\![M_t, M_{sp} \mid \langle 37.5,37.2,36.8,37.0,36.6\rangle, \langle 177,159,174,155,167, 150,177,135\rangle]\!]$.

Операція *сортування* (*сортування за зростанням* та *сортування за спаданням*) дозволяє отримати новий – відсортований – порядок слідування елементів (за зростанням та за спаданням відповідно) певного кортежу, який

називається *первісним кортежем*. Результатом застосування операції *сортування* до агрегата є агрегат, у всіх кортежах якого елементи перевпорядковані відповідно до нового порядку слідування індексів елементів відсортованого первісного кортежу.

Нехай $A = \left[\left[M_j | \left\langle a_i^j \right\rangle_{i=1}^{n_j}\right]\right]_{j=1}^{N}$ та $\exists k$ таке, що $1 < k < N, k \neq 2$ та $n_1 > n_k > n_N, n_2 = n_k$. Тоді результатом застосування операції *сортування за зростанням* агрегата A відносно елементів кортежу $\bar{a}^k$ є агрегат B такий, що:

$$B = A \uparrow \bar{a}^k = [\![M_1, M_2, \dots M_k, \dots M_N | , \left\langle a_\alpha^1, a_\beta^1, \dots a_\nu^1, \dots a_\omega^1, a_{n_k+1}^1, \dots, a_{n_1}^1 \right\rangle, \left\langle a_\alpha^2, a_\beta^2, \dots a_\nu^2, \dots a_\omega^2 \right\rangle, \dots \left\langle a_\alpha^k, a_\beta^k, \dots a_\nu^k, \dots a_\omega^k \right\rangle, \dots, \left\langle a_\alpha^N, a_\beta^N, \dots, a_\nu^N \right\rangle]\!],$$

де $a_\alpha^k \leq a_\beta^k \leq \dots \leq a_\nu^k \leq \dots \leq a_\omega^k$, $a_m^j \in \left\langle a_i^j \right\rangle_{i=1}^{n_j}$, $j = 1 \dots N$, $m \in [\alpha, \beta, \dots, \nu, \dots, \omega]$, $1 \leq m \leq n$ та $n = n_k$, якщо $n_j \geq n_k$, або $n = n_j$, якщо $n_j < n_k$.

Результатом застосування операції *сортування за спаданням* агрегата A відносно елементів кортежу $\bar{a}^k$ є агрегат B такий, що:

$$B = A \downarrow \bar{a}^k = [\![M_1, M_2, \dots M_k, \dots M_N | \left\langle a_\omega^1, \dots a_\nu^1, \dots a_\beta^1, a_\alpha^1, a_{n_k+1}^1, \dots a_{n_1}^1 \right\rangle, \left\langle a_\omega^2, \dots, a_\nu^2, \dots, a_\beta^2, a_\alpha^2 \right\rangle, \dots, \left\langle a_\omega^k, \dots, a_\nu^k, \dots, a_\beta^k, a_\alpha^k \right\rangle, \dots, \left\langle a_\nu^N, \dots, a_\beta^N, a_\alpha^N \right\rangle]\!].$$

де $a_\alpha^k \geq a_\beta^k \geq \dots \geq a_\nu^k \geq \dots \geq a_\omega^k$, $a_m^j \in \left\langle a_i^j \right\rangle_{i=1}^{n_j}$, $j = 1 \dots N$, $m \in [\alpha, \beta, \dots, \nu, \dots, \omega]$, $1 \leq m \leq n$ та $n = n_k$, якщо $n_j \geq n_k$, або $n = n_j$, якщо $n_j < n_k$.

Якщо $k = 1, k = 2$ або $k = N$, то операція сортування виконується аналогічно.

Якщо $n_1 = n_2 = \dots = n_k = \dots = n_N$, то результатом *сортування за зростанням* є агрегат виду: $B = A \uparrow \bar{a}^k = \left[\left[M_j | \left\langle a_\alpha^j, a_\beta^j, \dots, a_\nu^j, \dots, a_\omega^j \right\rangle\right]\right]_{j=1}^{N}$, а результатом *сортування за спаданням* є агрегат вигляду: $B = A \downarrow \bar{a}^k = \left[\left[M_j | \left\langle a_\omega^j, \dots, a_\nu^j, \dots, a_\beta^j, a_\alpha^j \right\rangle\right]\right]_{j=1}^{N}$.

Різновидом операцій *сортування* є операції *сортування з доповненням* (*сортування за зростанням з доповненням* та *сортування за спаданням з*

доповненням), які дозволяють збільшити довжину коротших кортежів відповідно до довжини первісного кортежу шляхом додавання значення x ($x \in [\emptyset, \_, q], q \in M_j, 1 \le j \le N$) або у кінець, або у початок коротших кортежів.

Результатом *сортування за зростанням з доповненням* у кінець коротших кортежів агрегата A є агрегат B:

$$B = A \uparrow (\bar{a}^k, x) = [\![M_1, M_2, \dots, M_k, \dots, M_N |, \langle a_\alpha^1, a_\beta^1, \dots, a_\nu^1, \dots, a_\omega^1, a_{n_k+1}^1, \dots, a_{n_1}^1 \rangle, \langle a_\alpha^2, a_\beta^2, \dots, a_\nu^2, \dots, a_\omega^2 \rangle, \dots, \langle a_\alpha^k, a_\beta^k, \dots, a_\nu^k, \dots, a_\omega^k \rangle, \dots, \langle a_\alpha^N, a_\beta^N, \dots, a_\nu^N, x, \dots, x \rangle]\!],$$

де $|\langle a_\alpha^N, a_\beta^N, \dots, a_\nu^N, x, \dots, x \rangle| = n_k$.

Результатом *сортування за зростанням з доповненням* напочатку коротших кортежів агрегата A є агрегат B:

$$B = A \uparrow (x, \bar{a}^k) = [\![M_1, M_2, \dots, M_k, \dots, M_N |, \langle a_\alpha^1, a_\beta^1, \dots, a_\nu^1, \dots, a_\omega^1, a_{n_k+1}^1, \dots, a_{n_1}^1 \rangle, \langle a_\alpha^2, a_\beta^2, \dots, a_\nu^2, \dots, a_\omega^2 \rangle, \dots, \langle a_\alpha^k, a_\beta^k, \dots, a_\nu^k, \dots, a_\omega^k \rangle, \dots, \langle x, \dots, x, a_\alpha^N, a_\beta^N, \dots, a_\nu^N \rangle]\!],$$

де $|\langle x, \dots, x, a_\alpha^N, a_\beta^N, \dots, a_\nu^N \rangle| = n_k$.

Результатом *сортування за спаданням з доповненням* у кінець коротших кортежів агрегата A є агрегат B:

$$B = A \downarrow (\bar{a}^k, x) = [\![M_1, M_2, \dots, M_k, \dots, M_N |, \langle a_\omega^1, \dots a_\nu^1, \dots a_\beta^1, a_\alpha^1, a_{n_k+1}^1, \dots, a_{n_1}^1 \rangle, \langle a_\omega^2, \dots, a_\nu^2, \dots, a_\beta^2, a_\alpha^2 \rangle, \dots, \langle a_\omega^k, \dots, a_\nu^k, \dots, a_\beta^k, a_\alpha^k \rangle, \dots, \langle a_\nu^N, \dots, a_\beta^N, a_\alpha^N, x, \dots, x \rangle]\!]^{\cdot}$$

Результатом *сортування за спаданням з доповненням* напочатку коротших кортежів агрегата A є агрегат B:

$$B = A \downarrow (x, \bar{a}^k) = [\![M_1, M_2, \dots, M_k, \dots, M_N |, \langle a_\omega^1, \dots a_\nu^1, \dots a_\beta^1, a_\alpha^1, a_{n_k+1}^1, \dots, a_{n_1}^1 \rangle, \langle a_\omega^2, \dots, a_\nu^2, \dots, a_\beta^2, a_\alpha^2 \rangle, \dots, \langle a_\omega^k, \dots, a_\nu^k, \dots, a_\beta^k, a_\alpha^k \rangle, \dots, \langle x, \dots, x, a_\nu^N, \dots, a_\beta^N, a_\alpha^N \rangle]\!]^{\cdot}$$

Операція *проріджування* дозволяє вилучити з первісного кортежу агрегата однакові значення, що розташовані поруч; одночасно вилучаються елементи з тими самими індексами у всіх інших кортежах. Якщо агрегат A визначений як:

$$A = [\![M_1, \dots, M_k, \dots, M_N | \langle a_1^1, \dots, a_m^1, \dots, a_{m+p}^1, a_{m+p+1}^1, \dots, a_{n_1}^1 \rangle, \dots,$$
$$\langle a_1^k, \dots, a_m^k, \dots, a_{m+p}^k, a_{m+p+1}^k, \dots, a_{n_k}^k \rangle, \dots,$$
$$\langle a_1^N, \dots, a_m^N, \dots, a_{m+p}^N, a_{m+p+1}^N, \dots, a_{n_N}^N \rangle]\!],$$

де $1 \le k \le N$, та $\exists m_l$, $\forall l$, таке, що $a_m^k = a_{m+1}^k = \dots = a_{m+p}^k$, $1 \le m \le (n_k - p)$, $1 \le p \le n_k$, то результатом операції проріджування агрегата A відносно кортежу $\bar{a}^k$ є агрегат B виду:

$$B = A \parallel \bar{a}^k = [\![M_1, \dots, M_k, \dots, M_N | \langle a_1^1, \dots, a_m^1, a_{m+p+1}^1, \dots, a_{n_1}^1 \rangle, \dots,$$
$$\langle a_1^k, \dots, a_m^k, a_{m+p+1}^k, \dots, a_{n_k}^k \rangle, \dots, \langle a_1^N, \dots, a_m^N, a_{m+p+1}^N, \dots, a_{n_N}^N \rangle]\!].$$

Операція *видалення* дозволяє вилучити певний елемент з агрегата. Нехай агрегат A має вигляд:

$$A = [\![M_1, \dots, M_k, \dots, M_N | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1}, \dots, \langle a_1^k, \dots, a_{m-1}^k, a_m^k,$$
$$a_{m+1}^k, \dots, a_{n_k}^k \rangle, \dots, \langle a_{i_N}^N \rangle_{i_N=1}^{n_N}]\!],$$

тоді результатом операції *видалення* елементу a_m^k з агрегата A є агрегат B такий, що:

$$B = A \ltimes a_m^k = [\![M_1, \dots, M_k, \dots, M_N | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1}, \dots, \langle a_1^k, \dots, a_{m-1}^k,$$
$$a_{m+1}^k, \dots, a_{n_k}^k \rangle \dots, \langle a_{i_N}^N \rangle_{i_N=1}^{n_N}]\!]$$

Різновидом операції видалення є операція *умовного видалення*, яка передбачає видалення елементу в разі виконання певної умови, як-то $a_m^k = d$, $a_m^k < d$ або $a_m^k > d$, $\forall d \in M_k$. Наприклад: $B = A \ltimes a_m^k\big|_{a_m^k = d}$.

Операція *вставлення* дозволяє додати певний елемент у агрегат. Нехай агрегат A має вигляд:

$$A = [\![M_1, \dots, M_k, \dots, M_N | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1}, \dots, \langle a_1^k, \dots, a_{m-1}^k, a_m^k,$$
$$a_{m+1}^k, \dots, a_{n_k}^k \rangle, \dots, \langle a_{i_N}^N \rangle_{i_N=1}^{n_N}]\!],$$

де $1 \le m \le N$ та $\exists d$, таке що $d \in M_k, \quad 1 \le k \le N$, то вставлення елементу d в одне і те саме місце у агрегаті A може бути виконане двома еквівалентними способами. Перший спосіб: $B = A \rtimes (d \prec a_m^k) = [\![M_1, \dots, M_k, \dots, M_N | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1}, \dots, \langle a_1^k, \dots, a_{m-1}^k, d, a_m^k, a_{m+1}^k, \dots, a_{n_k}^k \rangle, \dots, \langle a_{i_N}^N \rangle_{i_N=1}^{n_N}]\!]$.

Другій спосіб: $B = A \rtimes (d \succ a_{m-1}^k) = [\![M_1, \dots, M_k, \dots, M_N | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1}, \dots, \langle a_1^k, \dots, a_{m-1}^k, d, a_m^k, a_{m+1}^k, \dots, a_{n_k}^k \rangle, \dots, \langle a_{i_N}^N \rangle_{i_N=1}^{n_N}]\!]$.

Різновидом операції видалення є операція *умовного вставлення*, яка передбачає додавання елементу за виконання певної умови: $d = a_m^k, \quad d < a_m^k$ або $d > a_m^k, \quad \forall d \in M_k$. Наприклад, $B = A \rtimes (d \prec a_m^k)\big|_{d=a_m^k, \quad m=m_1 \dots m_2}$.

Таким чином, операції впорядкування, визначені у АСА, дозволяють перевпорядковувати елементи у кортежах та кортежі у агрегатах.

Арифметичні операції в АСА [9] можуть виконуватись лише над елементами, які належать до числових множин. Арифметичними операціями над *елементами* є додавання (+), віднімання (–), множення (), ділення (/). Вони виконуються над окремими значеннями за звичайними правилами виконання цих математичних операцій. Арифметичними операціями над *кортежами* є поелементне додавання (+.), поелементне віднімання (–.), поелементне множення (.), поелементне ділення (/.). Ці операції виконуються за такими правилами.

1. Відповідність між елементами різних кортежів встановлюється зліва направо (від першого елементу до останнього).
2. Якщо кортежі, над якими виконується поелементна операція, мають різну довжину, то операція виконується лише для k перших елементів кожного кортежу-операнду, де k – довжина найкоротшого кортежу-операнду; інші елементи всіх довших кортежів-операндів відкидаються.

Наприклад, результатом додавання двох кортежів дійсних чисел: $\langle 2.5, 3.1, 7.4 \rangle$ та $\langle 0.3, 12.8, 1.4, 6.9, 3.2 \rangle$ є кортеж $\langle 2.8, 15.9, 8.8 \rangle$.

Також в алгебраїчній системі агрегатів визначені *відношення*. Відношення в АСА [9, 72, 73] включають відношення між елементами, відношення між кортежами та відношення між агрегатами.

Відношеннями між *елементами* є відношення *більше* (>), *менше* (<), *дорівнює* (=), *передує* ($\prec$), *слідує за* ($\succ$). Перші три відношення (<, > та =) ґрунтуються на порівняні значень елементів, в той час, як два останніх відношення ($\prec$ та $\succ$) ґрунтуються на порівняні позиції елементів у кортежі.

Розглянемо елементи такого кортежу:

$$\bar{a} = \langle a_1, a_2, a_3, a_4 \rangle = \langle 11, 9, 11, 18 \rangle.$$

Тоді між елементами цього кортежу можна встановити наступні відношення: $a_1 > a_2;\quad a_3 < a_4;\quad a_1 = a_3;\quad a_1 \prec a_2;\quad a_3 \succ a_2.$

Арифметичні відношення можуть бути застосовані до двох кортежів $\overline{a^1}$ та $\overline{a^2}$, де $\overline{a^1} = \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1}$ та $\overline{a^2} = \langle a_{i_2}^2 \rangle_{i_2=1}^{n_2}$, якщо $a_{i_1}^1 \in M$ та $a_{i_2}^2 \in M$. Арифметичні відношення виконуються поелементно та включають наступні відношення: *строго більше* (>), *мажоритарно більше* ($\gg$), *строго менше* (<), *мажоритарно менше* ($\ll$), *строго дорівнює* (=) та *мажоритарно дорівнює* (<>).

Відношення *строго більше* між кортежами $\overline{a^1}$ та $\overline{a^2}$ визначається як: $\overline{a^1} > \overline{a^2}$ якщо $a_i^1 > a_i^2, \forall i, i = [1..n_1], n_1 = n_2$.

Відношення *мажоритарно більше* між кортежами $\overline{a^1}$ та $\overline{a^2}$ може бути визначене наступним чином. Нехай $N = \langle 1,2,\dots,n \rangle$, де $n = \begin{cases} n_1, if\ n_1 \le n_2 \\ n_2, if\ n_1 > n_2 \end{cases}$, та $\exists N_p \neq \emptyset$, $\exists N_q \neq \emptyset$ такі, що $N_p \cup N_q = N$, $N_p \cap N_q = \emptyset$ та $|N_p| > |N_q|$. Тоді $\forall p$, $\forall q,\ p \in N_p, q \in N_q$ відношення *мажоритарно більше*: $\overline{a^1} \gg \overline{a^2}$, якщо $a_p^1 > a_p^2$, $a_q^1 \le a_q^2$.

Відношення *строго менше* між кортежами $\overline{a^1}$ та $\overline{a^2}$ визначається як: $\overline{a^1} < \overline{a^2}$, якщо $a_i^1 < a_i^2, \forall i, i = [1..n_1], n_1 = n_2$.

Відношення *мажоритарно менше* між кортежами $\overline{a^1}$ та $\overline{a^2}$ може бути визначене наступним чином. Нехай $N = \langle 1,2,\dots,n \rangle$, де $n = \begin{cases} n_1, if\ n_1 \le n_2 \\ n_2, if\ n_1 > n_2 \end{cases}$ та

$\exists N_p \neq \emptyset, \exists N_q \neq \emptyset$ такі, що $N_p \cup N_q = N,\ N_p \cap N_q = \emptyset$ та $|N_p| < |N_q|$. Тоді $\forall p, \forall q,\ p \in N_p, q \in N_q$, відношення *мажоритарно менше*: $\overline{a^1} \ll \overline{a^2}$, якщо $a_p^1 \geq a_p^2$, $a_q^1 < a_q^2$.

Відношення *строго дорівнює* між кортежами $\overline{a^1}$ та $\overline{a^2}$ визначається як: $\overline{a^1} = \overline{a^2}$, якщо $a_i^1 = a_i^2, \forall i, i = [1..n_1], n_1 = n_2$.

Відношення *мажоритарно дорівнює* між кортежами $\overline{a^1}$ та $\overline{a^2}$ може бути визначене наступним чином.

Нехай $N = \langle 1,2,\dots,n\rangle$, де $n = \begin{cases} n_1, if\ n_1 \leq n_2 \\ n_2, if\ n_1 > n_2 \end{cases}$ та $\exists N_e \neq \emptyset$, $\exists N_p \neq \emptyset$, $\exists N_q \neq \emptyset$ такі, що $N_p \cup N_q \cup N_e = N$, $N_p \cap N_q \cap N_e = \emptyset$ та $|N_e| > |N_p|$, $|N_p| = |N_q|$. Тоді $\forall p \in N_p$, $\forall q \in N_q, \forall e \in N_e$ відношення *мажоритарно дорівнює*: $\overline{a^1} <> \overline{a^2}$, якщо $a_p^1 > a_p^2,\ a_q^1 < a_q^2,\ a_e^1 = a_e^2$

Розглянемо наступні кортежі:

$\overline{a^1} = \langle a_1^1, a_2^1, a_3^1, a_4^1\rangle = \langle 11, 9, 11, 18\rangle$;

$\overline{a^2} = \langle a_1^2, a_2^2, a_3^2, a_4^2\rangle = \langle 2, 7, 4, 10\rangle$;

$\overline{a^3} = \langle a_1^3, a_2^3, a_3^3, a_4^3, a_5^3\rangle = \langle 7, 19, 4, 10, 8\rangle$;

$\overline{a^4} = \langle a_1^4, a_2^4, a_3^4, a_4^4\rangle = \langle 11, 9, 11, 18\rangle$;

$\overline{a^5} = \langle a_1^5, a_2^5, a_3^5, a_4^5, a_5^5\rangle = \langle 14, 9, 11, 10, 8\rangle$.

Тоді між цими кортежами можна встановити наступні відношення:

$$\overline{a^1} > \overline{a^2}; \quad \overline{a^1} \gg \overline{a^3}; \quad \overline{a^2} < \overline{a^4};$$
$$\overline{a^3} \ll \overline{a^4}; \quad \overline{a^1} = \overline{a^4}; \quad \overline{a^1} <> \overline{a^5}.$$

Частотні відношення можуть бути застосовані до двох кортежів $\overline{a^1}$ та $\overline{a^2}$, де $\overline{a^1} = \langle a_{i_1}^1\rangle_{i_1=1}^{n_1}$ та $\overline{a^2} = \langle a_{i_2}^2\rangle_{i_2=1}^{n_2}$, якщо $a_{i_1}^1 \in M$ та $a_{i_2}^2 \in M$. Частотні відношення включають наступні відношення: *частіше* ($\rhd$), *рідше* ($\lhd$) та *однаково часто* ($\sim$).

Відношення *частіше* між кортежами $\overline{a^1}$ та $\overline{a^2}$ визначається як: $\overline{a^1} \rhd \overline{a^2}$, якщо $|\overline{a^1}| > |\overline{a^2}|$.

Відношення *рідше* між кортежами $\overline{a^1}$ та $\overline{a^2}$ визначається як: $\overline{a^1} \lhd \overline{a^2}$, якщо $|\overline{a^1}| < |\overline{a^2}|$.

Відношення *однаково часто* між кортежами $\overline{a^1}$ та $\overline{a^2}$ визначається як: $\overline{a^1} \sim \overline{a^2}$, якщо $\left|\overline{a^1}\right| = \left|\overline{a^2}\right|$.

Розглянемо наступні кортежі:

$\overline{a^1} = \langle a_1^1, a_2^1, a_3^1, a_4^1, a_5^1\rangle = \langle 14, 9, 15, 18{,}6\rangle$;

$\overline{a^2} = \langle a_1^2, a_2^2, a_3^2, a_4^2\rangle = \langle 2, 7, 4, 10\rangle$;

$\overline{a^3} = \langle a_1^3, a_2^3, a_3^3, a_4^3, a_5^3\rangle = \langle 7, 19, 4, 10, 8\rangle$;

Тоді, між ними можна встановити такі відношення: $\overline{a^1} \rhd \overline{a^2}$; $\overline{a^2} \lhd \overline{a^3}$; $\overline{a^1} \sim \overline{a^3}$.

Для того, щоб визначити *наскільки частішим* або *наскільки рідшим* є певний кортеж по відношенню до іншого, введемо *міру частоти* $\eta = \frac{\left|\overline{a^1}\right|}{\left|\overline{a^2}\right|}$. Наприклад, для кортежів $\overline{a^1}$, $\overline{a^2}$ та $\overline{a^3}$, визначених вище, ця міра має значення: $\eta_{12} = 1.25, \eta_{23} = 0.8, \eta_{13} = 1$.

Інтервальні відношення включають наступні відношення: *передує*, *настає після*, *збігається з*, *стикається з початком*, *стикається з кінцем*, *перекриває*, *перекривається*, *відбувається під час*, *містить*, *починає*, *починається з*, *закінчує*, *закінчується* та *відбувається між*.

Інтервальні відношення можуть бути застосовані лише до кортежів, які є дискретними інтервалами. Розглянемо поняття дискретного інтервалу докладно.

Інтервальні відношення ґрунтуються на відношеннях інтервальної алгебри Аллена [74]. Суттєва відмінність інтервальних відношень в АСА від інтервальної алгебри Аллена полягає у тому, що, на відміну від інтервальної алгебри Аллена, яка оперує неперервними інтервалами, інтервальні відношення в АСА застосовуються до дискретних інтервалів або дискретних інтервалів без повторень.

Дискретним інтервалом є кортеж $\bar{a} = \langle a_i\rangle_{i=1}^{n}$, елементами якого можуть бути будь-які значення a_i, впорядковані за зростанням ($a_i \leq a_{i+1}, \forall i \in [1 \ldots n-1],\ a_i \in M$) або за спаданням ($a_i \geq a_{i+1},\ \forall i \in [1 \ldots n-1],\ a_i \in M$), тобто дискретний інтервал є зростаючою або спадною послідовністю, що визначена на

деякій множині M.

Розглянемо відношення між дискретними інтервалами. Для узгодженості з інтервальною алгеброю Аллена будемо позначати ці відношення індексом d, наприклад, e_d. Тоді відношеннями між дискретними інтервалами $\overline{a^1}$ та $\overline{a^2}$ можна визначити наступним чином.

Відношення $b_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *передує* $\overline{a^2}$, якщо $a^1_{n_1} < a^2_1$.

Відношення $bi_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *настає після* $\overline{a^2}$, якщо $a^1_1 > a^2_{n_2}$.

Відношення $e_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *збігається з* $\overline{a^2}$, якщо $a^1_1 = a^2_1$, $a^1_{n_1} = a^2_{n_2}$.

Відношення $m_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *стикається з початком* $\overline{a^2}$, якщо $a^1_{n_1} = a^2_1$.

Відношення $mi_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *стикається з кінцем* $\overline{a^2}$, якщо $a^1_1 = a^2_{n_2}$.

Відношення $o_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *перекриває* $\overline{a^2}$, якщо $a^1_1 < a^2_1$, $a^1_{n_1} < a^2_{n_2}$ та $a^2_1 < a^1_{n_1}$.

Відношення $oi_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *перекривається* $\overline{a^2}$, якщо $a^2_1 < a^1_1$, $a^2_{n_2} < a^1_{n_1}$ та $a^1_1 < a^2_{n_2}$.

Відношення $d_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *відбувається під час* $\overline{a^2}$, якщо $a^1_1 > a^2_1$ та $a^1_{n_1} < a^2_{n_2}$.

Відношення $di_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *містить* $\overline{a^2}$, якщо $a^1_1 < a^2_1$ та $a^1_{n_1} > a^2_{n_2}$.

Відношення $s_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *починає* $\overline{a^2}$, якщо $a^1_1 = a^2_1$ та $a^1_{n_1} < a^2_{n_2}$.

Відношення $si_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *починається з* $\overline{a^2}$, якщо $a^1_1 = a^2_1$ та $a^1_{n_1} > a^2_{n_2}$.

Відношення $f_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *закінчує* $\overline{a^2}$, якщо $a^1_1 > a^2_1$ та $a^1_{n_1} = a^2_{n_2}$.

Відношення $fi_d(\overline{a^1}, \overline{a^2})$: $\overline{a^1}$ *закінчується* $\overline{a^2}$, якщо $a^1_1 < a^2_1$ та $a^1_{n_1} = a^2_{n_2}$.

Відношення *відбувається між* $w_d^{(\alpha,\beta)}(\overline{a^1}, \dots, \overline{a^n})$ не має аналогу в інтервальній алгебрі Аллена. Це відношення може бути застосоване до будь-якої кількості дискретних інтервалів та означає, що $\overline{a^j}$ ($j \in [1 \dots n]$, $n \geq 1$) *відбувається між* дискретними значеннями α та β. Отже, відношення$w_d^{(\alpha,\beta)}(\overline{a^1}, \overline{a^2})$ визначається так: $\overline{a^1}$ *та* $\overline{a^2}$ *відбуваються між* α *та* β, якщо $\alpha \leq a^1_1, \alpha \leq a^2_1, a^1_{n_1} \leq \beta, a^2_{n_2} \leq \beta$, де α та β – задані дискретні значення такі, що $\alpha, \beta, a^j_i \in M$, $j \in [1 \dots n]$, $i \in [1 \dots n_j]$, $n, n_j \in \mathbb{N}$; M – деяка множина.

Розглянемо відношення між дискретними інтервалами на прикладі таких кортежів:

$\overline{a^1} = \langle a_1^1, a_2^1, a_3^1 \rangle = \langle 1, 3, 4 \rangle$;

$\overline{a^2} = \langle a_1^2, a_2^2, a_3^2, a_4^2, a_5^2 \rangle = \langle 8, 9, 12, 13, 16 \rangle$;

$\overline{a^3} = \langle a_1^3, a_2^3, a_3^3, a_4^3 \rangle = \langle 1, 3, 5, 6 \rangle$;

$\overline{a^4} = \langle a_1^4, a_2^4, a_3^4, a_4^4 \rangle = \langle 8, 10, 14, 16 \rangle$;

$\overline{a^5} = \langle a_1^5, a_2^5, a_3^5, a_4^5, a_5^5, a_6^5 \rangle = \langle 2, 4, 5, 6, 7, 8 \rangle$;

$\overline{a^6} = \langle a_1^6, a_2^6, a_3^6 \rangle = \langle 6, 8, 10 \rangle$;

$\overline{a^7} = \langle a_1^7, a_2^7, a_3^7, a_4^7, a_5^7 \rangle = \langle 5, 7, 10, 15, 18 \rangle$;

$\overline{a^8} = \langle a_1^8, a_2^8, a_3^8, a_4^8 \rangle = \langle 8, 10,11, 12 \rangle$;

$\overline{a^9} = \langle a_1^9, a_2^9, a_3^9, a_4^9, a_5^9 \rangle = \langle 5, 7,12,15,16 \rangle$;

$\overline{a^{10}} = \langle a_1^{10}, a_2^{10}, a_3^{10} \rangle = \langle 4, 7, 8 \rangle$.

Тоді між ними можна встановити наступні відношення: $b_d(\overline{a^1}, \overline{a^2})$, $bi_d(\overline{a^2}, \overline{a^3})$, $e_d(\overline{a^2}, \overline{a^4})$, $m_d(\overline{a^5}, \overline{a^2})$, $mi_d(\overline{a^4}, \overline{a^5})$, $o_d(\overline{a^3}, \overline{a^5})$, $oi_d(\overline{a^5}, \overline{a^6})$, $d_d(\overline{a^6}, \overline{a^7})$, $di_d(\overline{a^7}, \overline{a^2})$, $s_d(\overline{a^1}, \overline{a^3})$, $si_d(\overline{a^2}, \overline{a^8})$, $f_d(\overline{a^4}, \overline{a^9})$, $fi_d(\overline{a^5}, \overline{a^{10}})$, $w_d^{(1,20)}(\overline{a^1}, \overline{a^2})$, які проілюстровані на рис. 3.1 – рис. 3.14 [9].

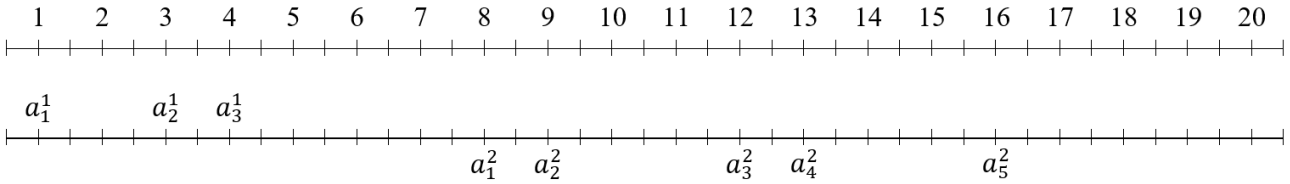

Рис. 3.1. Відношення *передує*

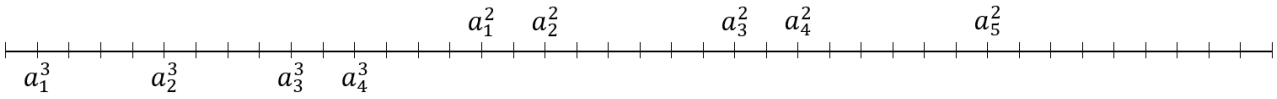

Рис. 3.2. Відношення *настає після*

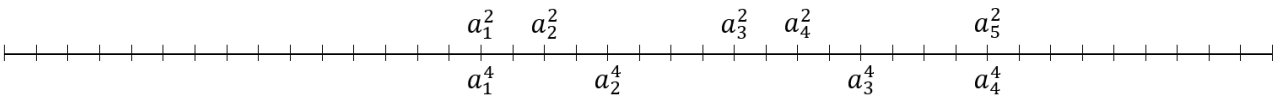

Рис. 3.3. Відношення *збігається з*

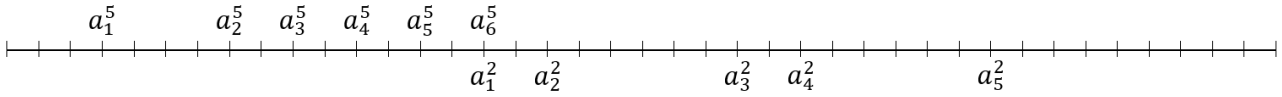

Рис. 3.4. Відношення *стикається з початком*

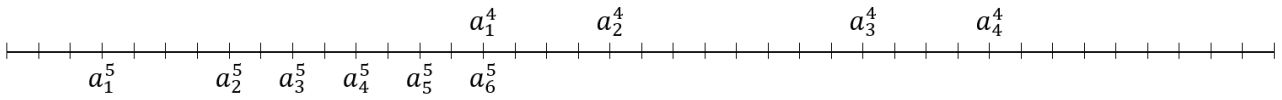

Рис. 3.5. Відношення *стикається з кінцем*

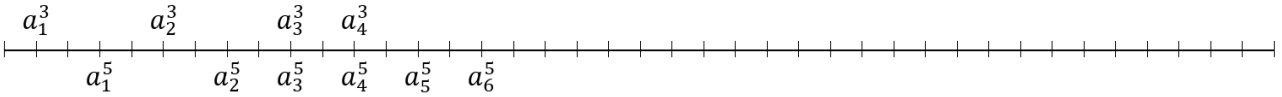

Рис. 3.6. Відношення *перекриває*

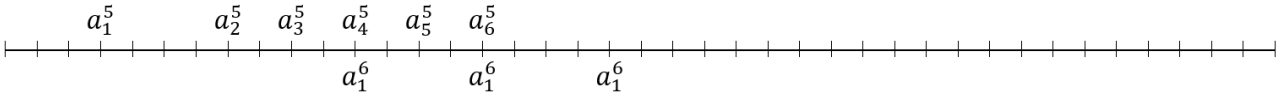

Рис. 3.7. Відношення *перекривається*

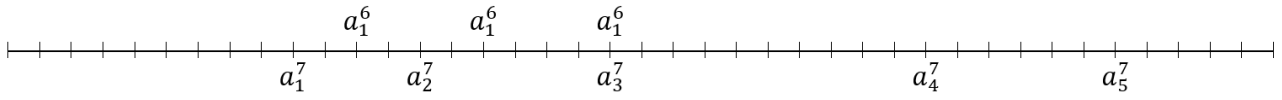

Рис. 3.8. Відношення *відбувається під час*

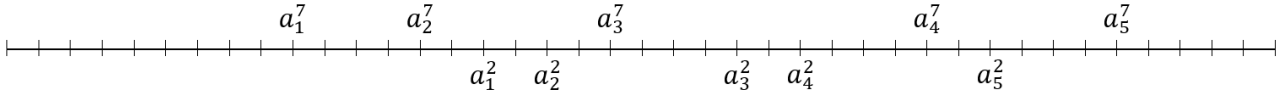

Рис. 3.9. Відношення *містить*

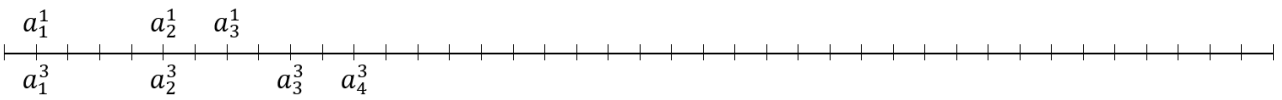

Рис. 3.10. Відношення *починає*

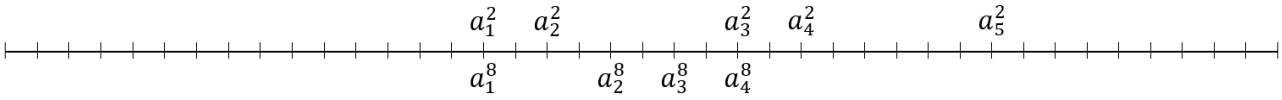

Рис. 3.11. Відношення *починається*

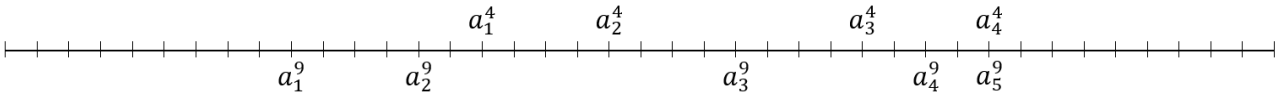

Рис. 3.12. Відношення *закінчує*

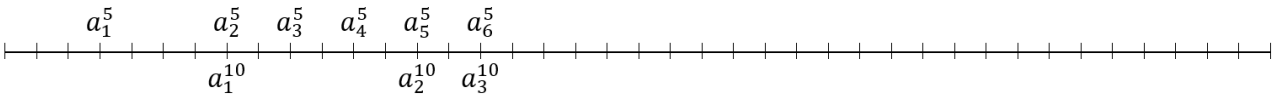

Рис. 3.13. Відношення *закінчується*

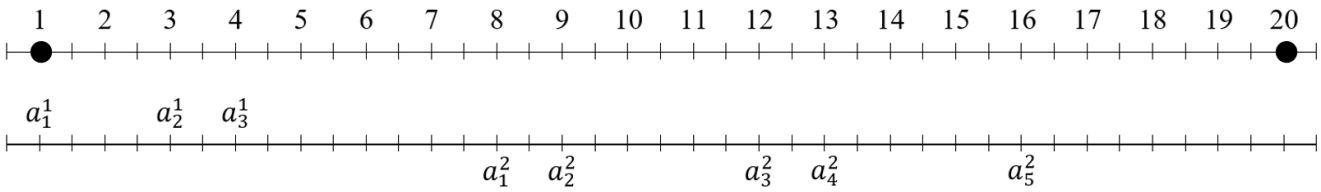

Рис. 3.14. Відношення *відбувається між*

Відношення між дискретними інтервалами призначені для виконання синхронізації темпоральних мультимодальних даних цифрових двійників.

Відношення між агрегатами включають відношення між множинами агрегатів та відношення між кортежами агрегатів [9, 72].

Відношеннями між множинами агрегатів є відношення: *еквівалентне* (≡), *включає* (⊃), *є включеним до* (⊂). Результат цих відношень залежить від сумісності агрегатів. Розглянемо агрегати A_1, A_2, A_3, A_4, A_5, and A_6:

$$A_1 = \left[\!\left[M_1, M_2, \dots, M_N | \langle a_{i_1}^1 \rangle_{i_1=1}^{n_1^1}, \dots, \langle a_{i_N}^1 \rangle_{i_N=1}^{n_N^1}\right]\!\right]$$
$$A_2 = \left[\!\left[M_1, M_2, \dots, M_N | \langle a_{i_1}^2 \rangle_{i_1=1}^{n_1^2}, \dots, \langle a_{i_N}^2 \rangle_{i_N=1}^{n_N^2}\right]\!\right]$$
$$A_3 = \left[\!\left[M_1, M_2^3, \dots, M_S^3 | \langle a_{i_1}^3 \rangle_{i_1=1}^{n_1^3}, \dots, \langle a_{i_S}^3 \rangle_{i_S=1}^{n_S^3}\right]\!\right]$$
$$A_4 = \left[\!\left[M_1^4, \dots, M_W^4 | \langle a_{i_1}^4 \rangle_{i_1=1}^{n_1^4}, \dots \langle a_{i_W}^4 \rangle_{i_W=1}^{n_W^4}\right]\!\right]$$
$$A_5 = \left[\!\left[M_1, M_2 | \langle a_{i_1}^5 \rangle_{i_1=1}^{n_1^5}, \langle a_{i_2}^5 \rangle_{i_2=1}^{n_2^5}\right]\!\right]$$
$$A_6 = \left[\!\left[M_2, M_1 | \langle a_{i_2}^6 \rangle_{i_2=1}^{n_2^6}, \langle a_{i_1}^6 \rangle_{i_1=1}^{n_1^6}\right]\!\right]$$

Отже, сумісність цих агрегатів є наступною:

$$A_1 \div A_2; \quad A_1 \doteq A_3; \quad A_1 \doteq A_5;$$
$$A_2 \doteq A_5; \quad A_1 \triangleq A_4; \quad A_5 \,(\div)\, A_6.$$

Тоді можуть бути встановлені наступні відношення між цими агрегатами:

$$\{A_1\} \equiv \{A_2\}; \quad \{A_1\} \supset \{A_5\}; \quad \{A_5\} \subset \{A_2\}.$$

Решта відношень може бути визначена за допомогою заперечення: $\{A_1\} \not\equiv \{A_3\}$, $\{A_4\} \not\supset \{A_5\}$, $\{A_3\} \not\subset \{A_6\}$. Слід зазначити, що незважаючи на те, що $\{A_1\} \not\supset \{A_3\}$, ці агрегати мають спільну множину M_1. Для встановлення цього факту слід застосувати логічну операцію *перерізу*: $\{A_1\} \cap \{A_3\} = M_1$ або у загальному випадку – $\{A_p\} \cap \{A_q\} \neq \emptyset; p, q \in \mathbb{N}$.

Визначимо відношення між множинами будь-яких двох агрегатів.

Відношення *еквівалентне* між двома агрегатами A_1 та A_2 визначається як: $\{A_1\} \equiv \{A_2\}$, якщо $A_1 \doteqdot A_2$.

Відношення *включає* між двома агрегатами A_1 та A_2 визначається як: $\{A_1\} \supset \{A_2\}$, якщо $A_1 \doteqdot A_2$, $|A_1| > |A_2|$, $\{A_2\} = \langle M_1, \dots, M_K \rangle$ та $\langle M_1, \dots, M_K \rangle \in \{A_1\}$.

Відношення *є включеним до* між двома агрегатами A_1 та A_2 визначається як: $\{A_1\} \subset \{A_2\}$, якщо $A_1 \doteqdot A_2$, $|A_1| < |A_2|$, $\{A_1\} = \langle M_1, \dots, M_K \rangle$ та $\langle M_1, \dots, M_K \rangle \in \{A_2\}$.

Відношення між кортежами агрегатів є ідентичними відношенням між окремими кортежами та включають відношення трьох категорій: арифметичні відношення, частотні відношення та інтервальні відношення. Проте, можливість їх застосування залежить від сумісності агрегатів: відношення між кортежами агрегатів можуть бути встановлені лише у випадку, коли ці агрегати є сумісними або квазісумісними. Приховано сумісні агрегати спочатку мають бути трансформовані у сумісні шляхом застосування до них операції *розміщення*,. Крім того, інтервальні відношення можуть бути застосовані до дискретних інтервалів. Якщо кортежі, що мають перевірятись на будь-яке інтервальне відношення, не є дискретними інтервалами, вони мають бути перетворені на дискретні інтервали шляхом застосування операції *сортування*.

В разі необхідності встановлення відношень між окремими кортежами агрегатів це зазначається явно: наприклад, $A_1(\overline{a^2}) \rhd A_2(\overline{a^2})$, якщо певне відношення правдиве для одного з кортежів кожного агрегата, або

$s_d\left(A_1(\overline{a^1},\overline{a^3}),A_2(\overline{a^1},\overline{a^3})\right)$ та $A_1(\overline{a^1},\overline{a^2},\overline{a^5}) \ll A_2(\overline{a^1},\overline{a^2},\overline{a^5})$ якщо відношення правдиве для двох чи більше кортежів одночасно.

Розглянемо декілька прикладів. Нехай дані сумісні агрегати A_1 та A_2:

$A_1 = [\![M_1, M_2, M_3 | \langle 3,4,8,9\rangle, \langle 3,1{,}16,12\rangle, \langle 48{,}13\rangle]\!]$;

$A_2 = [\![M_1, M_2, M_3 | \langle 1,5,7,8\rangle, \langle 8,10{,}11,12\rangle, \langle 12{,}15\rangle]\!]$;

Тоді між цими агрегатами та їхніми компонентами можна встановити такі відношення: $\{A_1\} \equiv \{A_2\}$; $\langle A_1\rangle \sim \langle A_2\rangle$; $A_1(\overline{a^1}) \gg A_2(\overline{a^1})$.

Нехай також дані квазісумісні агрегати A_3, A_4:

$A_3 = [\![M_1, M_2 | \langle 8,10,11,12\rangle, \langle 17,31\rangle]\!]$;

$A_4 = [\![M_1, M_2, M_4 | \langle 2,4,8\rangle, \langle 5,7,2,6,1\rangle, \langle 1,5,7,8\rangle]\!]$.

Ці агрегати пов’язані такими відношеннями:

$$\{A_3\} \subset \{A_4\}; \quad mi_d\left(A_3(\overline{a^1}), A_4(\overline{a^1})\right).$$

Всі значення у розглянутих прикладах є цілими числами. Проте, операції та відношення у АСА можуть також застосовуватись над будь-якими іншими типами даних, включаючи нечіткі значення.

Розглянутий математичний апарат АСА може бути застосований для формальної специфікації цифрового двійника фізичного об’єкта.

3.2. Формальна специфікація цифрового двійника на основі АСА

Розглянемо об’єкт спостереження S, який виявляє свою сутність через набір властивостей $F_1, F_2, \dots, F_N$, які можуть бути виміряні. В багатьох випадках ці властивості є взаємопов’язаними, оскільки вони представляють стан одного і того ж об’єкта та, фактично, є різними проявами його поведінки, що змінюється з плином часу. Наприклад, згідно з дослідженнями фізіології людини, при підвищені температури тіла пацієнта, його пульс також зростає. Комплексне подання даних, що відображають характеристики досліджуваного об’єкта, сприяє кращому розумінню загальної картини поведінки цього об’єкта [9, 72].

Представимо виміряні значення властивостей досліджуваного об’єкта S як агрегат A_S, компонентами якого є кортежі значень $\langle f_i^j \rangle_{i=1}^{n_j}$ властивостей F_j $(j = 1 \ldots N)$ об’єкта S:

$$A_S = [\![M_1, M_2, \ldots, M_N | \langle f_1^1, f_2^1, \ldots, f_{n_1}^1 \rangle, \langle f_1^2, f_2^2, \ldots, f_{n_2}^2 \rangle, \ldots \langle f_1^N, f_2^N, \ldots, f_{n_N}^N \rangle]\!].$$

Якщо вимірювання значень різних властивостей є одночасним, то всі кортежі агрегата A_S мають однакову довжину. В цьому випадку, агрегат A_S може бути розглянутий як послідовність мультикомпонентних значень (рис. 3.15).

Ця послідовність мультикомпонентних значень може бути розглянута як функція багатьох змінних: $y = x(t, F_1, F_2, \ldots, F_N)$. Такий погляд на агреговані дані, дозволяє застосовувати до них апарат числення багатьох змінних у задачах, в яких є доречними подання та оброблення даних за допомогою операцій та відношень, визначених в АСА, тобто застосування апарату АСА не заперечує використання інших математичних концепцій.

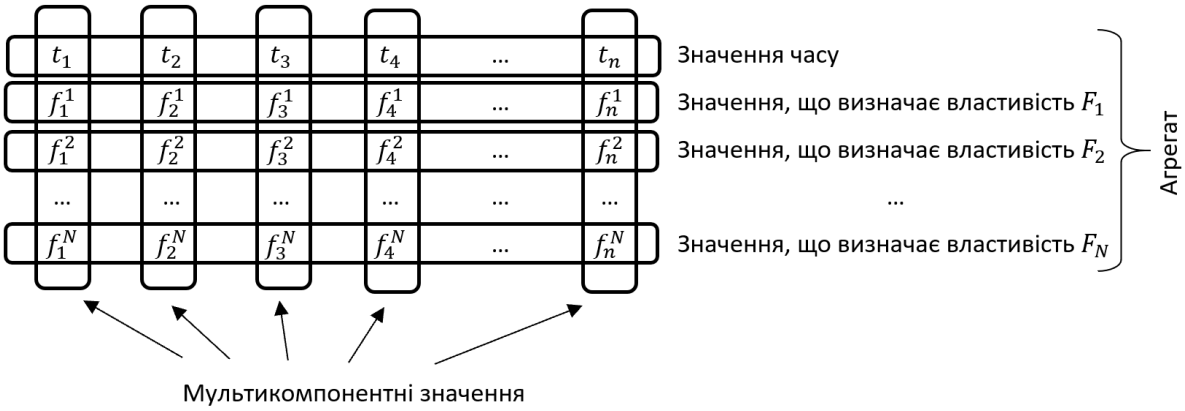

Рис. 3.15. Способи подання агрегованих даних [9]

Якщо дані про об’єкт спостереження отримані з урахуванням часу спостереження або вимірювання, то у агрегаті має бути присутнім кортеж відповідних темпоральних значень, які відповідають моментам часу, коли значення кортежів мультимодальних даних були отримані. Будемо називати такий агрегат *мультиобразом* цифрового двійника досліджуваного об’єкта [9]. Мультиобраз надає формальний опис послідовностей мультимодальних даних про досліджуваний об’єкт, отриманих з урахуванням часу в результаті

вимірювання, ґенерації та інших операцій з отримання даних. Фактично, мультиобраз є абстрактною специфікацією цифрового двійника, на основі якої можуть бути побудовані різноманітні моделі цифрового двійника. Визначимо поняття мультиобразу з позицій алгебраїчної системи агрегатів.

Мультиобраз – це непустий агрегат виду:

$$I = [\![T, M_1, \ldots, M_N | \langle t_1, \ldots, t_\tau \rangle, \langle a_1^1, \ldots, a_{n_1}^1 \rangle, \ldots, \langle a_1^N, \ldots, a_{n_N}^N \rangle]\!],$$

де T – це множина значень часу; $\tau \geq n_j, j \in [1, \ldots, N]$.

Отже, у математичному сенсі, *мультиобразом* є агрегат, першим кортежем якого є непустий кортеж значень часу. Ці значення можуть бути натуральними числами або будь-якими іншими значеннями, які забезпечують очевидність та однозначність подання інформації про моменти часу, в які було отримано елементи інших кортежів мультиобразу.

Сформулюємо *метод створення мультиобразу* досліджуваного об'єкта для формальної специфікації його цифрового двійника [9]. Вхідними даними для метода створення мультиобразу є набори темпоральних даних та тип (модальність) даних кожного набору. Метод складається з сімох етапів. Результатом методу є мультиобраз досліджуваного об'єкта, поданий у вигляді впорядкованої сукупності темпоральних мультимодальних даних.

Першим етапом методу є формування структури даних мультиобразу досліджуваного об'єкта, яка виконується, виходячи із задачі, що вирішується. Мультиобраз може визначатись математично або будь-яким іншим способом, який дозволяє однозначно задати порядок слідування наборів даних та їхню модальність.

На другому етапі методу відбувається декомпозиція специфікації мультиобразу на набір специфікацій *часткових* мультиобразів виду:

$$I_j = [\![T, M_j | \langle t_1, \ldots, t_{\tau_j} \rangle, \langle a_1, \ldots, a_{n_j} \rangle]\!],$$

де T – це множина значень часу; M_j – множина даних j-ї модальності; $\tau_j, n_j \in \mathbb{N}$; $\tau_j \geq n_j$; $j \in [1, \ldots, N]$; N – кількість наборів даних.

На третьому етапі відбувається отримання даних. Процедура отримання даних передбачає визначення способу (протоколу, формату) передавання даних, визначення періоду часу для передавання даних, встановлення зв'язку з джерелом даних та, власне, отримання даних визначеним способом та у визначеному форматі.

На четвертому етапі відбувається підготовка даних кожної модальності з метою формування відповідного часткового мультиобразу. Частковий мультиобраз являє собою агрегат, який включає два кортежі елементів: кортеж часових значень та кортеж елементів певної модальності. Елементами є окремі значення або впорядковані сукупності значень (однорідні або різнорідні). Підготовка даних, яка виконується на цьому етапі, полягає у визначенні (виявленні) значень часу у сукупності даних, отриманих на попередньому етапі. Ця процедура може бути як тривіальною, коли формат даних передбачає явне подання значень часу для компонента темпоральних даних, так і складною, коли значення часу подані у неявній формі, або є нечіткими. Спосіб виявлення прихованих або нечітко визначених значень часу має розроблятись для кожного конкретного формату подання даних.

На п'ятому етапі відбувається об'єднання часткових мультиобразів у єдиний мультиобраз. Процедура об'єднання мультиобразів включає наступні кроки.

1. Нормалізація часткових мультиобразів (*нормалізованим* мультиобразом (*нормалізованим* кортежем) називатимемо мультиобраз (кортеж) до елементів якого додані фіктивні елементи; *фіктивним* елементом називатимемо деяке унікальне значення, яке відсутнє у кортежі до виконання нормалізації; як фіктивний може використовуватись пустий елемент ∅, невизначений елемент _ або будь-яке інше унікальне значення; на інших етапах роботи з мультиобразом (аналіз, відтворення тощо) фіктивні елементи ігноруються):

$$\hat{I}_j = I_j \rtimes \left(\langle \emptyset \rangle_{k=1}^{E_i^j} \succ a_{n_j}^j \right) = \left[\!\left[T, M_j | \langle t_i \rangle_{i=1}^{\tau_j}, \langle \langle a_{i_j}^j \rangle_{i_j=1}^{n_j}, \langle \emptyset \rangle_{l=1}^{E_i^j} \rangle \right]\!\right] =$$

$$= \left[\!\left[T, M_j | \langle t_i \rangle_{i=1}^{\tau_j}, \langle \hat{a}_{i_j}^j \rangle_{i_j=1}^{n_j+E_i^j} \right]\!\right],$$

де $\rtimes$ – операція вставлення;

E_i^j – кількість фіктивних елементів; $E_i^j = \left(\sum_{j=1}^{N} \tau_j\right) - n_j$;

$\hat{a}_{i_j}^j$ – елемент нормалізованого кортежу; $j \in [1, \ldots, N]$.

2. Об'єднання нормалізованих часткових мультиобразів:

$$I = \bigcup_{j=1}^{N} \hat{I}_j = \left[\!\left[T, M_1, \ldots M_N \middle| \langle \langle t_i \rangle_{i=1}^{\tau_j} \rangle_{j=1}^{N}, \langle \hat{a}_{i_1} \rangle_{i_1=1}^{n_1+E_i^1}, \ldots \langle \hat{a}_{i_N} \rangle_{i_N=1}^{n_N+E_i^N}\right]\!\right].$$

На шостому етапі відбувається сортування мультиобразу, отриманого на попередньому етапі, за кортежем часових значень:

$$I_S = I \uparrow \bar{t} = \left[\!\left[T, M_1, \ldots M_N \middle| \langle \langle t_\sigma \rangle_{\sigma=1}^{\tau_j} \rangle_{j=1}^{N}, \langle \hat{a}_{\sigma_1} \rangle_{\sigma_1=1}^{n_1+E_i^1}, \ldots \langle \hat{a}_{\sigma_N} \rangle_{\sigma_N=1}^{n_N+E_i^N}\right]\!\right],$$

де σ та σ_j – індекси, що задають порядок слідування елементів у відсортованому кортежі; $j \in [1, \ldots, N]$.

На сьомому етапі відбувається проріджування відсортованого мультиобразу, отриманого на попередньому етапі, за кортежем часових значень:

$$I_S = I \parallel \bar{t} = \left[\!\left[T, M_1, \ldots M_N \middle| \langle t_\sigma \rangle_{\sigma=1}^{\left(\sum_{j=1}^{N} \tau_j\right)-\delta}, \langle \hat{a}_{\sigma_1} \rangle_{\sigma_1=1}^{n_1+E_i^1-\delta}, \ldots, \langle \hat{a}_{\sigma_N} \rangle_{\sigma_N=1}^{n_N+E_i^N-\delta}\right]\!\right],$$

де δ – кількість відкинутих дублікатів значень часу.

Розглянемо приклад, який демонструє виконання основних етапів методу. Припустимо, що пацієнт, який перебуває вдома під дистанційним медичним наглядом, за призначенням лікаря має тричі на день вимірювати температуру тіла та частоту пульсу: вранці, вдень та ввечері. В результаті проведеного самообстеження пацієнт надав лікарю запитувані дані вимірювань, супроводжуючи їх нечіткими значеннями часу, які можна представити у вигляді двох часткових мультиобразів – часткового мультиобразу A, що містить значення температури, та часткового мультиобразу B, що містить значення пульсу:

$$A = [\![\tilde{T}, M_t \mid \langle (8{:}30, 9{:}00, 9{:}30), (13{:}30, 14{:}00, 14{:}30), (18{:}45, 19{:}00, 19{:}15)\rangle, \langle 36.6, 36.9, 37.1\rangle]\!]$$

$$B = [\![\tilde{T}, M_p \mid \langle (8{:}45, 9{:}00, 9{:}15), (13{:}45, 14{:}00, 14{:}15), (18{:}50, 19{:}00, 19{:}10)\rangle, \langle 75, 78, 80\rangle]\!].$$

Для отримання мультиобразу, який відображає дані цього дослідження, потрібно об’єднати часткові мультиобрази A та B. Безпосереднє застосування операції об’єднання до цих часткових мультиобразів призвело би до того, що кортеж часових значень об'єднаного агрегата G був би вдвічі довший, ніж кортеж значень температури та кортеж значень пульсу, що зруйнувало би відповідність між значеннями часу та вимірюваними значеннями. Щоб уникнути цієї суперечності, спочатку потрібно виконати нормалізацію цих часткових мультиобразів, а саме, застосувати операцію вставлення і додати фіктивні елементи у кінець кортежу значень температури та у кінець кортежу значень пульсу:

$$A_1 = [\![\tilde{T}, M_t \mid \langle(8{:}30, 9{:}00, 9{:}30), (13{:}30, 14{:}00, 14{:}30), (18{:}45, 19{:}00, 19{:}15)\rangle, \langle 36.6, 36.9, 37.1, \emptyset, \emptyset, \emptyset\rangle]\!]$$

$$B_1 = [\![\tilde{T}, M_p \mid \langle(8{:}45, 9{:}00, 9{:}15), (13{:}45, 14{:}00, 14{:}15), (18{:}50, 19{:}00, 19{:}10)\rangle, \langle 75, 78, 80, \emptyset, \emptyset, \emptyset\rangle]\!].$$

Після нормалізації застосуємо до цих агрегатів операцію об’єднання:

$$G_1 = A_1 \cup B_1 = [\![\tilde{T}, M_t, M_p \mid \langle(8{:}30, 9{:}00, 9{:}30), (13{:}30, 14{:}00, 14{:}30), (18{:}45, 19{:}00, 19{:}15), (8{:}45, 9{:}00, 9{:}15), (13{:}45, 14{:}00, 14{:}15), (18{:}50, 19{:}00, 19{:}10)\rangle, \langle 36.6, 36.9, 37.1, \emptyset, \emptyset, \emptyset\rangle, \langle 75, 78, 80, \emptyset, \emptyset, \emptyset\rangle]\!]$$

Далі потрібно відсортувати отриманий мультиобраз G_1 за часовим кортежем за допомогою відповідної операції АСА:

$$G_2 = G_1 \uparrow \langle\tilde{t}\rangle_{i=1}^{3} = [\![\tilde{T}, M_t, M_p \mid \langle(8{:}30, 9{:}00, 9{:}30), (8{:}45, 9{:}00, 9{:}15), (13{:}30, 14{:}00, 14{:}30), (13{:}45, 14{:}00, 14{:}15), (18{:}45, 19{:}00, 19{:}15), (18{:}50, 19{:}00, 19{:}10)\rangle, \langle 36.6, \emptyset, 36.9, \emptyset, 37.1, \emptyset\rangle, \langle 75, \emptyset, 78, \emptyset, 80, \emptyset\rangle]\!]$$

Тепер кортежі впорядковані, проте вони включають деякі зайві елементи, такі, як дублікати нечітких значень часу та фіктивні елементи. Щоб видалити їх, застосуємо операцію проріджування:

$$G_3 = G_2 \parallel \langle\tilde{t}\rangle_{i=1}^{3} = [\![\tilde{T}, M_t, M_p \mid \langle(8{:}30, 9{:}00, 9{:}30), (13{:}30, 14{:}00, 14{:}30), (18{:}45, 19{:}00, 19{:}15), \langle 36.6, 36.9, 37.1\rangle, \langle 75, 78, 80\rangle]\!]$$

В результаті отримуємо мультиобраз, який містить дані різних обстежень пацієнта та може бути використаний для аналізу цих даних.

3.3. Синхронізація мультимодальних темпоральних наборів даних

При вирішенні деяких задач аналізу темпоральних мультимодальних даних може виникати потреба у синхронізації декількох мультиобразів. Розглянемо метод, що дозволяє виконати таку синхронізацію, у тому числі у випадку, коли значення часу є нечітко визначеними.

Метод синхронізації мультиобразів [9, 72] ґрунтується на застосуванні інтервальних відношень та виконанні правил синхронізації. Правила синхронізації поділяються на:

- універсальне правило синхронізації (УП),
- базові правила синхронізації (БП),
- правила нечіткої синхронізації (НП).

Універсальне правило синхронізації дозволяє виконати точну синхронізацію темпоральних мультимодальних даних та може бути застосоване до довільних кортежів значень часу, проте воно потребує виконання операцій, кількість яких не може бути зменшена.

Базові правила синхронізації [9, 72] призначені для виконання точної синхронізації, причому якщо вхідні кортежі значень часу є періодичними послідовностями, то кількість операцій над даними може бути зменшена за рахунок застосування відповідного *шаблону синхронізації* – одного чи декількох правил синхронізації, які відповідають виду періодичності у часових кортежах, що синхронізуються. Таким чином, застосування шаблону синхронізації, що створений шляхом комбінування базових правил синхронізації, дозволяє підвищити ефективність оброблення даних.

Правила нечіткої синхронізації [9, 72] дозволяють виконати синхронізацію з деякою припустимою похибкою визначення елементів кортежів значень часу у мультиобразах, що синхронізуються; вони також можуть бути застосовані для створення шаблону синхронізації з метою підвищення ефективності оброблення даних мультиобразів.

Сформулюємо правила всіх типів [9, 72] для синхронізації двох мультиобразів I_1 та I_2, заданих наступним чином:

$$I_1 = [\![T, M_1 |\, \langle t_{i_1}^1 \rangle, \langle a_{i_1}^1 \rangle]\!]_{i_1=1}^{n_1}$$
$$I_2 = [\![T, M_2 |\, \langle t_{i_2}^2 \rangle, \langle a_{i_2}^2 \rangle]\!]_{i_2=1}^{n_2}.$$

У результаті застосування деякого правила синхронізації (шаблону синхронізації) отримаємо два наступні мультиобрази:

$$I_1^S = [\![T, M_1 |\, \langle t_i \rangle, \langle d_i^1 \rangle]\!]_{i=1}^{n}$$
$$I_2^S = [\![T, M_2 |\, \langle t_i \rangle, \langle d_i^2 \rangle]\!]_{i=1}^{n},$$

де d_i^1 та d_i^2 – значення, що належать нормалізованим кортежам елементів мультиобразів I_1 та I_2 відповідно; $n = n_1 + n_2 - \delta$, δ – кількість дублікатів значень часу у об'єднаному кортежі $\langle t_i \rangle_{i=1}^{n}$ часових значень.

Універсальне правило синхронізації УП1 мультиобразів I_1 та I_2 задається наступним чином:

$$\langle t_i \rangle_{i=1}^{n_1+n_2-\delta} = \left(\left(\left(\langle t_{i_1}^1 \rangle_{i_1=1}^{n_1} \cup \langle t_{i_2}^2 \rangle_{i_2=1}^{n_2} \right) \uparrow \right) \| \right)$$
$$d_i^1 = \begin{cases} a_{i_1}^1 \text{ якщо } t_i = t_{i_1}^1, & \forall i, \forall i_1 \\ \emptyset \text{ в інших випадках} \end{cases}$$
$$d_i^2 = \begin{cases} a_{i_2}^2 \text{ якщо } t_i = t_{i_2}^2, & \forall i, \forall i_2 \\ \emptyset \text{ в інших випадках,} \end{cases}$$

де $\cup$ – операція об'єднання, $\uparrow$ – операція сортування, $\|$ – операція проріджування.

Визначимо тепер базові правила синхронізації. Для цього розглянемо можливі інтервальні відношення між дискретними інтервалами $\bar{t}^1 = \langle t_i^1 \rangle_{i=1}^{n_1}$ та $\bar{t}^2 = \langle t_i^2 \rangle_{i=1}^{n_2}$ та сформулюємо для кожного з цих відношень базові правила синхронізації.

Нехай $e_d(\bar{t}^1, \bar{t}^2) = true$, тобто $\bar{t}^1$ та $\bar{t}^2$ зв'язані відношенням *збігається з*. З практичної точки зору це означає, що визначення (вимірювання, ґенерування) значень обох кортежів $\bar{a}^1$ та $\bar{a}^2$ відбувалось одночасно. Можливі випадки для відношення *збігається з* проілюстровані графічно на рис. 3.16 – рис. 3.18.

Перша група випадків для відношення $e_d(\bar{t}^1, \bar{t}^2)$ характеризується тим, що $n_1 \bmod 2 = 0$ та $\bar{t}^1 \sim \bar{t}^2$. Останнє означає. що $|\bar{t}^1| = |\bar{t}^2|$ (тобто $n_1 = n_2$).

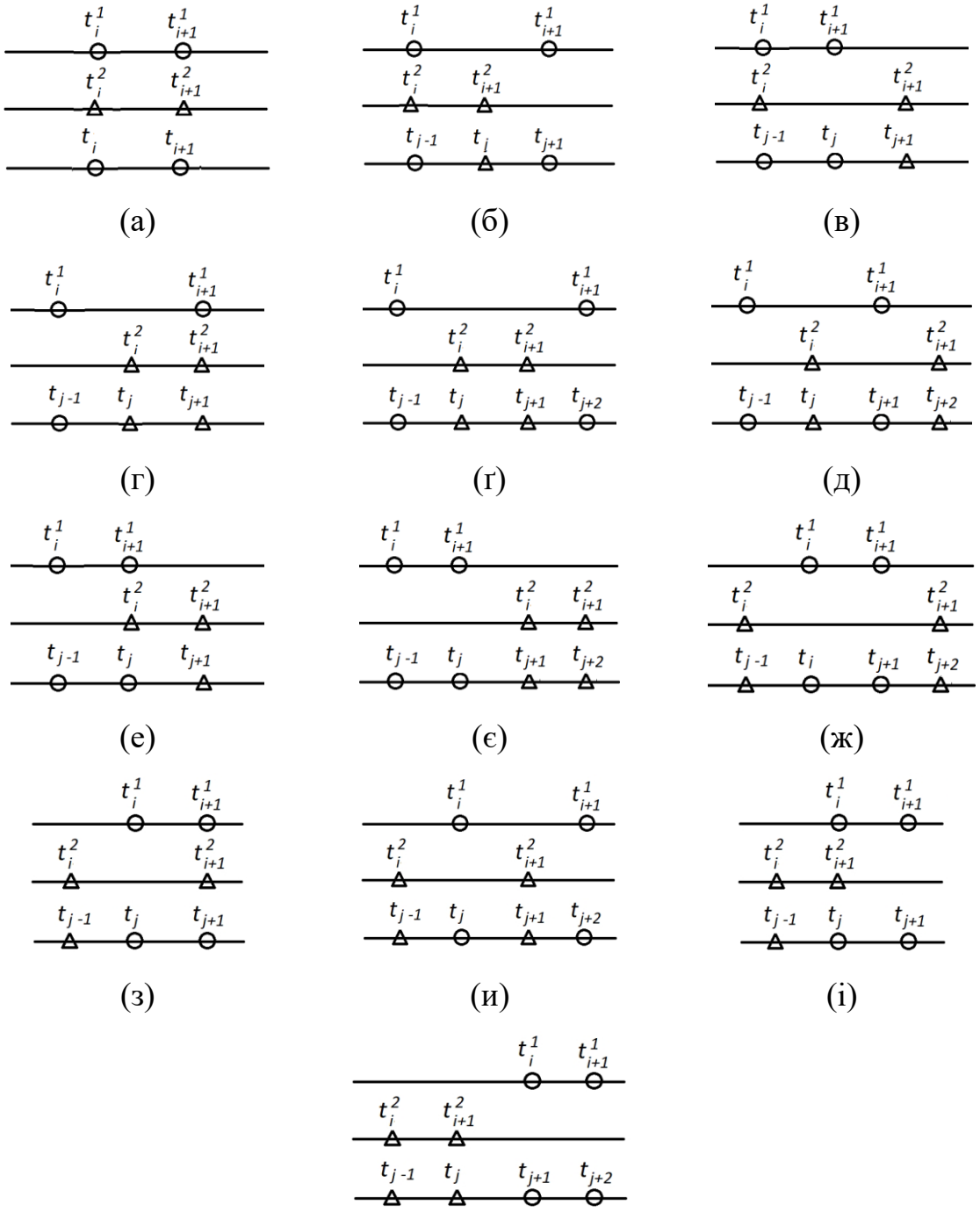

Рис. 3.16. Випадки для $e_d(\overline{t^1}, \overline{t^2})$ при $\overline{t^1} \sim \overline{t^2}$

Якщо $t_i^1 = t_i^2, \quad \forall\, i \in [1 \ldots n_1]$, то $n = n_1$ та значення елементів мультиобразів I_1^S та I_2^S отримуються згідно з базовим правилом синхронізації БП1:

$$t_i = t_i^1$$
$$d_i^1 = a_i^1,$$
$$d_i^2 = a_i^2$$

що може бути графічно проілюстроване рис. 3.16а.

У випадку, коли $t_i^1 = t_i^2$ та $t_{i+1}^1 > t_{i+1}^2$ (рис. 3.16б), значення всіх елементів мультиобразів I_1^S та I_2^S, крім перших та останніх, отримуються наступним чином:

$$\begin{array}{lll} t_{j-1} = t_i^1 & t_j = t_{i+1}^2 & t_{j+1} = t_{i+1}^1 \\ d_{j-1}^1 = a_i^1 & d_j^1 = \emptyset & d_{j+1}^1 = a_{i+1}^1 \\ d_{j-1}^2 = a_i^2 & d_j^2 = a_{i+1}^2 & d_{j+1}^2 = \emptyset \end{array}$$

Це означає, що у момент часу $t_i^1 = t_i^2$ визначеними є обидва значення a_i^1 та a_i^2; у момент часу t_{i+1}^2 визначеним є лише значення a_{i+1}^2; у момент часу t_{i+1}^1 визначеним є лише значення a_{i+1}^1. Індекс j обчислюється як $j = \frac{3i}{2}$, де $i \in [2 \dots (n_1 - 2)]$ та $i \; mod \; 2 = 0$. Загальна кількість елементів в кожному спільному кортежі, а саме, у кортежах $\langle t_j \rangle_{j=1}^n$, $\langle d_j^1 \rangle_{j=1}^n$, $\langle d_j^2 \rangle_{j=1}^n$, складає $n = \frac{3n_1}{2} - 1$. Таким чином, правило синхронізації мультимодальних даних БП2 визначається так:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-1} = t_i^1 & t_j = t_{i+1}^2 & t_{j+1} = t_{i+1}^1 & t_n = t_{n_1}^1 \\ d_1^1 = a_1^1 & d_{j-1}^1 = a_i^1 & d_j^1 = \emptyset & d_{j+1}^1 = a_{i+1}^1 & d_n^1 = a_{n_1}^1 \\ d_1^2 = a_1^2 & d_{j-1}^2 = a_i^2 & d_j^2 = a_{i+1}^2 & d_{j+1}^2 = \emptyset & d_n^2 = a_{n_2}^2 \end{array}$$

Аналогічно, якщо $t_i^1 = t_i^2$ та $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16в), $\forall \; i \in [2 \dots (n_1 - 2)]$, $i \; mod \; 2 = 0$, $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$, то синхронізація значень кортежів визначається правилом БП3, що визначається виразами:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-1} = t_i^1 & t_j = t_{i+1}^1 & t_{j+1} = t_{i+1}^2 & t_n = t_{n_1}^1 \\ d_1^1 = a_1^1 & d_{j-1}^1 = a_i^1 & d_j^1 = a_{i+1}^1 & d_{j+1}^1 = \emptyset & d_n^1 = a_{n_1}^1 \\ d_1^2 = a_1^2 & d_{j-1}^2 = a_i^2 & d_j^2 = \emptyset & d_{j+1}^2 = a_{i+1}^2 & d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 < t_i^2$ та $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16г), $\forall \; i \in [2 \dots (n_1 - 2)]$, $i \; mod \; 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та правило синхронізації даних БП4:

$t_1 = t_1^1$ $t_{j-1} = t_i^1$ $t_j = t_i^2$ $t_{j+1} = t_{i+1}^1$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-1}^1 = a_i^1$ $d_j^1 = \emptyset$ $d_{j+1}^1 = a_{i+1}^1$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-1}^2 = \emptyset$ $d_j^2 = a_i^2$ $d_{j+1}^2 = a_{i+1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$ та $t_{i+1}^1 > t_{i+1}^2$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації даних БП5:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_i^2$ $t_j = t_{i+1}^2$ $t_{j+1} = t_{i+1}^1$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = \emptyset$ $d_{j+1}^1 = a_{i+1}^1$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = a_i^2$ $d_j^2 = a_{i+1}^2$ $d_{j+1}^2 = \emptyset$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ та $t_{i+1}^1 > t_i^2$ (рис. 3.16д), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = 2i$, $n = 2(n_1 - 1)$ та синхронізація значень кортежів визначається правилом БП6, що визначається виразами:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_i^2$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = a_i^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ та $t_{i+1}^1 = t_i^2$ (рис. 3.16е), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та правило синхронізації даних БП7:

$t_1 = t_1^1$ $t_{j-1} = t_i^1$ $t_j = t_i^2$ $t_{j+1} = t_{i+1}^2$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-1}^1 = a_i^1$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-1}^2 = \emptyset$ $d_j^2 = a_i^2$ $d_{j+1}^2 = a_{i+1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_{i+1}^1 < t_i^2$ (рис. 3.16є), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП8:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_{i+1}^1$ $t_j = t_i^2$ $t_{j+1} = t_{i+1}^2$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = a_{i+1}^1$ $d_j^1 = \emptyset$ $d_{j+1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = \emptyset$ $d_j^2 = a_i^2$ $d_{j+1}^2 = a_{i+1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$ та $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16ж), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації даних БП9:

$t_1 = t_1^1$ $t_{j-2} = t_i^2$ $t_{j-1} = t_i^1$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-2}^1 = \emptyset$ $d_{j-1}^1 = a_i^1$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-2}^2 = a_i^2$ $d_{j-1}^2 = \emptyset$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$ та $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16з), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та синхронізація значень кортежів визначається правилом БП10, що визначається наступним чином:

$t_1 = t_1^1$	$t_{j-1} = t_i^2$	$t_j = t_i^1$	$t_{j+1} = t_{i+1}^1$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_{j-1}^1 = \emptyset$	$d_j^1 = a_i^1$	$d_{j+1}^1 = a_{i+1}^1$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_{j-1}^2 = a_i^2$	$d_j^2 = \emptyset$	$d_{j+1}^2 = a_{i+1}^2$	$d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 > t_{i+1}^2$ та $t_i^1 < t_{i+1}^2$ (рис. 3.16и), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = 2i, n = 2(n_1 - 1)$ та правило синхронізації даних БП11:

$t_1 = t_1^1$	$t_{j-2} = t_i^2$	$t_{j-1} = t_i^1$	$t_j = t_{i+1}^2$	$t_{j+1} = t_{i+1}^1$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_{j-2}^1 = \emptyset$	$d_{j-1}^1 = a_i^1$	$d_j^1 = \emptyset$	$d_{j+1}^1 = a_{i+1}^1$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_{j-2}^2 = a_i^2$	$d_{j-1}^2 = \emptyset$	$d_j^2 = a_{i+1}^2$	$d_{j+1}^2 = \emptyset$	$d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_{i+1}^2$ (рис. 3.16і), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та правило синхронізації мультимодальних даних БП12:

$t_1 = t_1^1$	$t_{j-1} = t_i^2$	$t_j = t_i^1$	$t_{j+1} = t_{i+1}^1$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_{j-1}^1 = \emptyset$	$d_j^1 = a_i^1$	$d_{j+1}^1 = a_{i+1}^1$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_{j-1}^2 = a_i^2$	$d_j^2 = a_{i+1}^2$	$d_{j+1}^2 = \emptyset$	$d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_{i+1}^2$ (рис. 3.16ї), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП13:

$t_1 = t_1^1$	$t_{j-2} = t_i^2$	$t_{j-1} = t_{i+1}^2$	$t_j = t_i^1$	$t_{j+1} = t_{i+1}^1$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_{j-2}^1 = \emptyset$	$d_{j-1}^1 = \emptyset$	$d_j^1 = a_i^1$	$d_{j+1}^1 = a_{i+1}^1$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_{j-2}^2 = a_i^2$	$d_{j-1}^2 = a_{i+1}^2$	$d_j^2 = \emptyset$	$d_{j+1}^2 = \emptyset$	$d_n^2 = a_{n_2}^2$

Друга група випадків для $e_d(\overline{t^1}, \overline{t^2})$ характеризується тим, що $n_1 \bmod 2 \neq 0$ та $\overline{t^1} \sim \overline{t^2}$. Часове узгодження *i*-х елементів та (*i*+1)-х елементів є подібним до узгодження для першої групи випадків, проте, оскільки n_1 є непарним, то для синхронізації значень потрібно також аналізувати значення часу $t_{n_1-1}^1$ та $t_{n_2-1}^2$.

Якщо $t_i^1 = t_i^2$ (рис. 3.16а), $\forall\, i \in [1 \ldots n_1]$, то синхронізація відбувається згідно з правилом синхронізації БП1.

Якщо $t_i^1 = t_i^2$, $t_{i+1}^1 > t_{i+1}^2$ (рис. 3.16б), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2}$ та правило синхронізації даних БП14:

$t_1 = t_1^1 \quad t_{j-1} = t_i^1 \quad t_j = t_{i+1}^2 \quad t_{j+1} = t_{i+1}^1 \quad t_{n-1} = t_{n_1-1}^1 \quad t_n = t_{n_1}^1$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = a_i^1 \quad d_j^1 = \emptyset \quad d_{j+1}^1 = a_{i+1}^1 \quad d_{n-1}^1 = a_{n_1-1}^1 \quad d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \quad d_j^2 = a_{i+1}^2 \quad d_{j+1}^2 = \emptyset \quad d_{n-1}^2 = a_{n_2-1}^2 \quad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_i^2$, $t_{i+1}^1 > t_{i+1}^2$ (рис. 3.16б), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило синхронізації БП15:

$t_1 = t_1^1 \quad t_{j-1} = t_i^1 \quad t_j = t_{i+1}^2 \quad t_{j+1} = t_{i+1}^1 \quad t_{n-2} = t_{n_1-1}^1$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = a_i^1 \quad d_j^1 = \emptyset \quad d_{j+1}^1 = a_{i+1}^1 \quad d_{n-2}^1 = a_{n_1-1}^1$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \quad d_j^2 = a_{i+1}^2 \quad d_{j+1}^2 = \emptyset \quad d_{n-2}^2 = \emptyset$

$t_{n-1} = t_{n_2-1}^2 \qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = \emptyset \qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2 \qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_i^2$, $t_{i+1}^1 > t_{i+1}^2$ (рис. 3.16б), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та синхронізація значень кортежів визначається правилом БП16:

$t_1 = t_1^1 \quad t_{j-1} = t_i^1 \quad t_j = t_{i+1}^2 \quad t_{j+1} = t_{i+1}^1 \quad t_{n-2} = t_{n_2-1}^2$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = a_i^1 \quad d_j^1 = \emptyset \quad d_{j+1}^1 = a_{i+1}^1 \quad d_{n-2}^1 = \emptyset$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \quad d_j^2 = a_{i+1}^2 \quad d_{j+1}^2 = \emptyset \quad d_{n-2}^2 = a_{n_2-1}^2$

$t_{n-1} = t_{n_1-1}^1 \qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1 \qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = \emptyset \qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16в), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2}$ та правило синхронізації даних БП17:

$t_1 = t_1^1 \quad t_{j-1} = t_i^1 \quad t_j = t_{i+1}^1 \quad t_{j+1} = t_{i+1}^2 \quad t_{n-1} = t_{n_1-1}^1 \quad t_n = t_{n_1}^1$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = a_i^1 \quad d_j^1 = a_{i+1}^1 \quad d_{j+1}^1 = \emptyset \quad d_{n-1}^1 = a_{n_1-1}^1 \quad d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \quad d_j^2 = \emptyset \quad d_{j+1}^2 = a_{i+1}^2 \quad d_{n-1}^2 = a_{n_2-1}^2 \quad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16в), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило синхронізації БП18:

$t_1 = t_1^1$ $\quad t_{j-1} = t_i^1$ $\quad t_j = t_{i+1}^1$ $\quad t_{j+1} = t_{i+1}^2$ $\quad t_{n-2} = t_{n_1-1}^1$
$d_1^1 = a_1^1$ $\quad d_{j-1}^1 = a_i^1$ $\quad d_j^1 = a_{i+1}^1$ $\quad d_{j+1}^1 = \emptyset$ $\quad d_{n-2}^1 = a_{n_1-1}^1$
$d_1^2 = a_1^2$ $\quad d_{j-1}^2 = a_i^2$ $\quad d_j^2 = \emptyset$ $\quad d_{j+1}^2 = a_{i+1}^2$ $\quad d_{n-2}^2 = \emptyset$

$t_{n-1} = t_{n_2-1}^2$ $\qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = \emptyset$ $\qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2$ $\qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16в), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та синхронізація значень кортежів визначається правилом БП19:

$t_1 = t_1^1$ $\quad t_{j-1} = t_i^1$ $\quad t_j = t_{i+1}^1$ $\quad t_{j+1} = t_{i+1}^2$ $\quad t_{n-2} = t_{n_2-1}^2$
$d_1^1 = a_1^1$ $\quad d_{j-1}^1 = a_i^1$ $\quad d_j^1 = a_{i+1}^1$ $\quad d_{j+1}^1 = \emptyset$ $\quad d_{n-2}^1 = \emptyset$
$d_1^2 = a_1^2$ $\quad d_{j-1}^2 = a_i^2$ $\quad d_j^2 = \emptyset$ $\quad d_{j+1}^2 = a_{i+1}^2$ $\quad d_{n-2}^2 = a_{n_2-1}^2$

$t_{n-1} = t_{n_1-1}^1$ $\qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1$ $\qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = \emptyset$ $\qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2}$ та правило синхронізації даних БП20:

$t_1 = t_1^1$ $\quad t_{j-1} = t_i^1$ $\quad t_j = t_i^2$ $\quad t_{j+1} = t_{i+1}^1$ $\quad t_{n-1} = t_{n_1-1}^1$ $\quad t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $\quad d_{j-1}^1 = a_i^1$ $\quad d_j^1 = \emptyset$ $\quad d_{j+1}^1 = a_{i+1}^1$ $\quad d_{n-1}^1 = a_{n_1-1}^1$ $\quad d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $\quad d_{j-1}^2 = \emptyset$ $\quad d_j^2 = a_i^2$ $\quad d_{j+1}^2 = a_{i+1}^2$ $\quad d_{n-1}^2 = a_{n_2-1}^2$ $\quad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило синхронізації БП21:

$t_1 = t_1^1$ $\quad t_{j-1} = t_i^1$ $\quad t_j = t_i^2$ $\quad t_{j+1} = t_{i+1}^1$ $\quad t_{n-2} = t_{n_1-1}^1$
$d_1^1 = a_1^1$ $\quad d_{j-1}^1 = a_i^1$ $\quad d_j^1 = \emptyset$ $\quad d_{j+1}^1 = a_{i+1}^1$ $\quad d_{n-2}^1 = a_{n_1-1}^1$
$d_1^2 = a_1^2$ $\quad d_{j-1}^2 = \emptyset$ $\quad d_j^2 = a_i^2$ $\quad d_{j+1}^2 = a_{i+1}^2$ $\quad d_{n-2}^2 = \emptyset$

$t_{n-1} = t^2_{n_2-1}$ $\quad$ $t_n = t^1_{n_1}$

$d^1_{n-1} = \emptyset$ $\quad$ $d^1_n = a^1_{n_1}$

$d^2_{n-1} = a^2_{n_2-1}$ $\quad$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} = t^2_{i+1}$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t^1_{n_1-1} > t^2_{n_2-1}$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та синхронізація значень кортежів визначається правилом БП22:

$t_1 = t^1_1$	$t_{j-1} = t^1_i$	$t_j = t^2_i$	$t_{j+1} = t^1_{i+1}$	$t_{n-2} = t^2_{n_2-1}$
$d^1_1 = a^1_1$	$d^1_{j-1} = a^1_i$	$d^1_j = \emptyset$	$d^1_{j+1} = a^1_{i+1}$	$d^1_{n-2} = \emptyset$
$d^2_1 = a^2_1$	$d^2_{j-1} = \emptyset$	$d^2_j = a^2_i$	$d^2_{j+1} = a^2_{i+1}$	$d^2_{n-2} = a^2_{n_2-1}$

$t_{n-1} = t^1_{n_1-1}$ $\quad$ $t_n = t^1_{n_1}$

$d^1_{n-1} = a^1_{n_1-1}$ $\quad$ $d^1_n = a^1_{n_1}$

$d^2_{n-1} = \emptyset$ $\quad$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} > t^2_{i+1}$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t^1_{n_1-1} = t^2_{n_2-1}$then $j = 2i, n = 2n_1 - 3$ та правило синхронізації БП23:

$t_1 = t^1_1$	$t_{j-2} = t^1_i$	$t_{j-1} = t^2_i$	$t_j = t^2_{i+1}$	$t_{j+1} = t^1_{i+1}$
$d^1_1 = a^1_1$	$d^1_{j-2} = a^1_i$	$d^1_{j-1} = \emptyset$	$d^1_j = \emptyset$	$d^1_{j+1} = a^1_{i+1}$
$d^2_1 = a^2_1$	$d^2_{j-2} = \emptyset$	$d^2_{j-1} = a^2_i$	$d^2_j = a^2_{i+1}$	$d^2_{j+1} = \emptyset$

$t_{n-1} = t^1_{n_1-1}$ $\quad$ $t_n = t^1_{n_1}$

$d^1_{n-1} = a^1_{n_1-1}$ $\quad$ $d^1_n = a^1_{n_1}$

$d^2_{n-1} = a^2_{n_2-1}$ $\quad$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} > t^2_{i+1}$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t^1_{n_1-1} < t^2_{n_2-1}$then $j = 2i, n = 2(n_1 - 1)$ та правило синхронізації БП24:

$t_1 = t^1_1$	$t_{j-2} = t^1_i$	$t_{j-1} = t^2_i$	$t_j = t^2_{i+1}$	$t_{j+1} = t^1_{i+1}$	$t_{n-2} = t^1_{n_1-1}$
$d^1_1 = a^1_1$	$d^1_{j-2} = a^1_i$	$d^1_{j-1} = \emptyset$	$d^1_j = \emptyset$	$d^1_{j+1} = a^1_{i+1}$	$d^1_{n-2} = a^1_{n_1-1}$
$d^2_1 = a^2_1$	$d^2_{j-2} = \emptyset$	$d^2_{j-1} = a^2_i$	$d^2_j = a^2_{i+1}$	$d^2_{j+1} = \emptyset$	$d^2_{n-2} = \emptyset$

$t_{n-1} = t^2_{n_2-1}$ $\quad$ $t_n = t^1_{n_1}$

$d^1_{n-1} = \emptyset$ $\quad$ $d^1_n = a^1_{n_1}$

$d^2_{n-1} = a^2_{n_2-1}$ $\quad$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} > t^2_{i+1}$ (рис. 3.16г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t^1_{n_1-1} > t^2_{n_2-1}$then $j = 2i, n = 2(n_1 - 1)$ та правило синхронізації БП25:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_i^2$ $t_j = t_{i+1}^2$ $t_{j+1} = t_{i+1}^1$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = \emptyset$ $d_{j+1}^1 = a_{i+1}^1$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = a_i^2$ $d_j^2 = a_{i+1}^2$ $d_{j+1}^2 = \emptyset$

$t_{n-2} = t_{n_2}^2$- $t_{n-1} = t_{n_1}^1$ $t_n = t_{n_1}^1$
$d_{n-2}^1 = \emptyset$ $d_{n-1}^1 = a_n^1$ $d_n^1 = a_{n_1}^1$
$d_{n-2}^2 = a_{n_2}^2$ $d_{n-1}^2 = \emptyset$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 < t_{i+1}^2$, $t_{i+1}^1 > t_i^2$ (рис. 3.16д), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = 2i$, $n = 2n_1 - 3$ та правило синхронізації мультимодальних даних БП26:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_i^2$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = a_i^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$

$t_{n-1} = t_{n_1-1}^1$ $t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1$ $d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 < t_{i+1}^2$, $t_{i+1}^1 > t_i^2$ (рис. 3.16д), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП27:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_i^2$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = a_i^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$

$t_{n-2} = t_{n_1}^1$- $t_{n-1} = t_{n_2-1}^2$ $t_n = t_{n_1}^1$
$d_{n-2}^1 = a_{n_1}^1$ $d_{n-1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_{n-2}^2 = \emptyset$ $d_{n-1}^2 = a_{n_2-1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$, $t_{i+1}^1 < t_{i+1}^2$, $t_{i+1}^1 > t_i^2$ (рис. 3.16д), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних ПБ28:

$t_1 = t_1^1$ $t_{j-2} = t_i^1$ $t_{j-1} = t_i^2$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$
$d_1^1 = a_1^1$ $d_{j-2}^1 = a_i^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-2}^2 = \emptyset$ $d_{j-1}^2 = a_i^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$

$t_{n-2} = t^2_{n_2-1}$ $t_{n-1} = t^1_{n_1-1}$ $t_n = t^1_{n_1}$
$d^1_{n-2} = \emptyset$ $d^1_{n-1} = a^1_{n_1-1}$ $d^1_n = a^1_{n_1}$
$d^2_{n-2} = a^2_{n_2-1}$ $d^2_{n-1} = \emptyset$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} < t^2_{i+1}$, $t^1_{i+1} = t^2_i$ (рис. 3.16е), $\forall\, i \in [2 \dots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t^1_{n_1-1} = t^2_{n_2-1}$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2}$ та правило синхронізації мультимодальних даних БП29:

$t_1 = t^1_1$ $t_{j-1} = t^1_i$ $t_j = t^2_i$ $t_{j+1} = t^2_{i+1}$
$d^1_1 = a^1_1$ $d^1_{j-1} = a^1_i$ $d^1_j = a^1_{i+1}$ $d^1_{j+1} = \emptyset$
$d^2_1 = a^2_1$ $d^2_{j-1} = \emptyset$ $d^2_j = a^2_i$ $d^2_{j+1} = a^2_{i+1}$

$t_{n-1} = t^1_{n_1-1}$ $t_n = t^1_{n_1}$
$d^1_{n-1} = a^1_{n_1-1}$ $d^1_n = a^1_{n_1}$
$d^2_{n-1} = a^2_{n_2-1}$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} < t^2_{i+1}$, $t^1_{i+1} = t^2_i$ (рис. 3.16е), $\forall\, i \in [2 \dots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t^1_{n_1-1} < t^2_{n_2-1}$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та синхронізація значень кортежів визначається правилом БП30:

$t_1 = t^1_1$ $t_{j-1} = t^1_i$ $t_j = t^2_i$ $t_{j+1} = t^2_{i+1}$ $t_{n-2} = t^1_{n_1-1}$
$d^1_1 = a^1_1$ $d^1_{j-1} = a^1_i$ $d^1_j = a^1_{i+1}$ $d^1_{j+1} = \emptyset$ $d^1_{n-2} = a^1_{n_1-1}$
$d^2_1 = a^2_1$ $d^2_{j-1} = \emptyset$ $d^2_j = a^2_i$ $d^2_{j+1} = a^2_{i+1}$ $d^2_{n-2} = \emptyset$

$t_{n-1} = t^2_{n_2-1}$ $t_n = t^1_{n_1}$
$d^1_{n-1} = \emptyset$ $d^1_n = a^1_{n_1}$
$d^2_{n-1} = a^2_{n_2-1}$ $d^2_n = a^2_{n_2}$

Якщо $t^1_i < t^2_i$, $t^1_{i+1} < t^2_{i+1}$, $t^1_{i+1} = t^2_i$ (рис. 3.16е), $\forall\, i \in [2 \dots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t^1_{n_1-1} > t^2_{n_2-1}$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило синхронізації мультимодальних даних БП31:

$t_1 = t^1_1$ $t_{j-1} = t^1_i$ $t_j = t^2_i$ $t_{j+1} = t^2_{i+1}$ $t_{n-2} = t^2_{n_2-1}$
$d^1_1 = a^1_1$ $d^1_{j-1} = a^1_i$ $d^1_j = a^1_{i+1}$ $d^1_{j+1} = \emptyset$ $d^1_{n-2} = \emptyset$
$d^2_1 = a^2_1$ $d^2_{j-1} = \emptyset$ $d^2_j = a^2_i$ $d^2_{j+1} = a^2_{i+1}$ $d^2_{n-2} = a^2_{n_2-1}$

$t_{n-1} = t^1_{n_1-1}$ $t_n = t^1_{n_1}$
$d^1_{n-1} = a^1_{n_1-1}$ $d^1_n = a^1_{n_1}$
$d^2_{n-1} = \emptyset$ $d^2_n = a^2_{n_2}$

Якщо $t^1_{i+1} < t^2_i$ (рис. 3.16є), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t^1_{n_1-1} = t^2_{n_2-1}$, то $j = 2i$, $n = 2n_1 - 3$ та правило синхронізації мультимодальних даних БП32:

$t_1 = t^1_1$ $t_{j-2} = t^1_i$ $t_{j-1} = t^1_{i+1}$ $t_j = t^2_i$ $t_{j+1} = t^2_{i+1}$
$d^1_1 = a^1_1$ $d^1_{j-2} = a^1_i$ $d^1_{j-1} = a^1_{i+1}$ $d^1_j = \emptyset$ $d^1_{j+1} = \emptyset$
$d^2_1 = a^2_1$ $d^2_{j-2} = \emptyset$ $d^2_{j-1} = \emptyset$ $d^2_j = a^2_i$ $d^2_{j+1} = a^2_{i+1}$

$t_{n-1} = t^1_{n_1-1}$ $t_n = t^1_{n_1}$
$d^1_{n-1} = a^1_{n_1-1}$ $d^1_n = a^1_{n_1}$
$d^2_{n-1} = a^2_{n_2-1}$ $d^2_n = a^2_{n_2}$

Якщо $t^1_{i+1} < t^2_i$ (рис. 3.16є), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t^1_{n_1-1} < t^2_{n_2-1}$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП33:

$t_1 = t^1_1$ $t_{j-2} = t^1_i$ $t_{j-1} = t^1_{i+1}$ $t_j = t^2_i$ $t_{j+1} = t^2_{i+1}$
$d^1_1 = a^1_1$ $d^1_{j-2} = a^1_i$ $d^1_{j-1} = a^1_{i+1}$ $d^1_j = \emptyset$ $d^1_{j+1} = \emptyset$
$d^2_1 = a^2_1$ $d^2_{j-2} = \emptyset$ $d^2_{j-1} = \emptyset$ $d^2_j = a^2_i$ $d^2_{j+1} = a^2_{i+1}$

$t_{n-2} = t^1_{n_1-1}$ $t_{n-1} = t^2_{n_2-1}$ $t_n = t^1_{n_1}$
$d^1_{n-2} = a^1_{n_1-1}$ $d^1_{n-1} = \emptyset$ $d^1_n = a^1_{n_1}$
$d^2_{n-2} = \emptyset$ $d^2_{n-1} = a^2_{n_2-1}$ $d^2_n = a^2_{n_2}$

Якщо $t^1_{i+1} < t^2_i$ (рис. 3.16є), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t^1_{n_1-1} > t^2_{n_2-1}$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП34:

$t_1 = t^1_1$ $t_{j-2} = t^1_i$ $t_{j-1} = t^1_{i+1}$ $t_j = t^2_i$ $t_{j+1} = t^2_{i+1}$
$d^1_1 = a^1_1$ $d^1_{j-2} = a^1_i$ $d^1_{j-1} = a^1_{i+1}$ $d^1_j = \emptyset$ $d^1_{j+1} = \emptyset$
$d^2_1 = a^2_1$ $d^2_{j-2} = \emptyset$ $d^2_{j-1} = \emptyset$ $d^2_j = a^2_i$ $d^2_{j+1} = a^2_{i+1}$

$t_{n-2} = t^2_{n_2-1}$ $t_{n-1} = t^1_{n_1-1}$ $t_n = t^1_{n_1}$
$d^1_{n-2} = \emptyset$ $d^1_{n-1} = a^1_{n_1-1}$ $d^1_n = a^1_{n_1}$
$d^2_{n-2} = a^2_{n_2-1}$ $d^2_{n-1} = \emptyset$ $d^2_n = a^2_{n_2}$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16ж), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = 2i$, $n = 2n_1 - 3$ та синхронізація значень кортежів визначається правилом БП35:

$t_1 = t_1^1$ $t_{j-2} = t_i^2$ $t_{j-1} = t_i^1$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$
$d_1^1 = a_1^1$ $d_{j-2}^1 = \emptyset$ $d_{j-1}^1 = a_i^1$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-2}^2 = a_i^2$ $d_{j-1}^2 = \emptyset$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$

$t_{n-1} = t_{n_1-1}^1$ $t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1$ $d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16ж), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило БП36:

$t_1 = t_1^1$ $t_{j-2} = t_i^2$ $t_{j-1} = t_i^1$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$
$d_1^1 = a_1^1$ $d_{j-2}^1 = \emptyset$ $d_{j-1}^1 = a_i^1$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-2}^2 = a_i^2$ $d_{j-1}^2 = \emptyset$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$

$t_{n-2} = t_{n_1-1}^1$ $t_{n-1} = t_{n_2-1}^2$ $t_n = t_{n_1}^1$
$d_{n-2}^1 = a_{n_1-1}^1$ $d_{n-1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_{n-2}^2 = \emptyset$ $d_{n-1}^2 = a_{n_2-1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 < t_{i+1}^2$ (рис. 3.16ж), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило БП37:

$t_1 = t_1^1$ $t_{j-2} = t_i^2$ $t_{j-1} = t_i^1$ $t_j = t_{i+1}^1$ $t_{j+1} = t_{i+1}^2$
$d_1^1 = a_1^1$ $d_{j-2}^1 = \emptyset$ $d_{j-1}^1 = a_i^1$ $d_j^1 = a_{i+1}^1$ $d_{j+1}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-2}^2 = a_i^2$ $d_{j-1}^2 = \emptyset$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$

$t_{n-2} = t_{n_2-1}^2$ $t_{n-1} = t_{n_1-1}^1$ $t_n = t_{n_1}^1$
$d_{n-2}^1 = \emptyset$ $d_{n-1}^1 = a_{n_1-1}^1$ $d_n^1 = a_{n_1}^1$
$d_{n-2}^2 = a_{n_2-1}^2$ $d_{n-1}^2 = \emptyset$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16з), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2}$ та правило БП38:

$t_1 = t_1^1 \quad t_{j-1} = t_i^2 \quad t_j = t_i^1 \quad t_{j+1} = t_{i+1}^1 \quad t_{n-1} = t_{n_1-1}^1 \quad t_n = t_{n_1}^1$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = \emptyset \quad d_j^1 = a_i^1 \; d_{j+1}^1 = a_{i+1}^1 \quad d_{n-1}^1 = a_{n_1-1}^1 \; d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \; d_j^2 = \emptyset \quad d_{j+1}^2 = a_{i+1}^2 \quad d_{n-1}^2 = a_{n_2-1}^2 \; d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16з), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило БП39:

$t_1 = t_1^1 \quad t_{j-1} = t_i^2 \quad t_j = t_i^1 \quad t_{j+1} = t_{i+1}^1 \quad t_{n-2} = t_{n_1-1}^1$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = \emptyset \quad d_j^1 = a_i^1 \quad d_{j+1}^1 = a_{i+1}^1 \quad d_{n-2}^1 = a_{n_1-1}^1$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \quad d_j^2 = \emptyset \quad d_{j+1}^2 = a_{i+1}^2 \quad d_{n-2}^2 = \emptyset$

$t_{n-1} = t_{n_2-1}^2 \qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = \emptyset \qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2 \qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 = t_{i+1}^2$ (рис. 3.16з), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило БП40:

$t_1 = t_1^1 \quad t_{j-1} = t_i^2 \quad t_j = t_i^1 \quad t_{j+1} = t_{i+1}^1 \quad t_{n-2} = t_{n_2-1}^2$
$d_1^1 = a_1^1 \quad d_{j-1}^1 = \emptyset \quad d_j^1 = a_i^1 \quad d_{j+1}^1 = a_{i+1}^1 \quad d_{n-2}^1 = \emptyset$
$d_1^2 = a_1^2 \quad d_{j-1}^2 = a_i^2 \quad d_j^2 = \emptyset \quad d_{j+1}^2 = a_{i+1}^2 \quad d_{n-2}^2 = a_{n_2-1}^2$

$t_{n-1} = t_{n_1-1}^1 \qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1 \qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = \emptyset \qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 > t_{i+1}^2$, $t_i^1 < t_{i+1}^2$ (рис. 3.16и), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i \bmod 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = 2i$, $n = 2n_1 - 3$ та синхронізація значень кортежів визначається правилом БП41:

$t_1 = t_1^1 \quad t_{j-2} = t_i^2 \quad t_{j-1} = t_i^1 \quad t_j = t_{i+1}^2 \quad t_{j+1} = t_{i+1}^1$
$d_1^1 = a_1^1 \quad d_{j-2}^1 = \emptyset \quad d_{j-1}^1 = a_i^1 \quad d_j^1 = \emptyset \quad d_{j+1}^1 = a_{i+1}^1$
$d_1^2 = a_1^2 \quad d_{j-2}^2 = a_i^2 \quad d_{j-1}^2 = \emptyset \quad d_j^2 = a_{i+1}^2 \quad d_{j+1}^2 = \emptyset$

$t_{n-1} = t_{n_1-1}^1 \qquad t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1 \qquad d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2 \qquad d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 > t_{i+1}^2$, $t_i^1 < t_{i+1}^2$ (рис. 3.16и), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП42:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-2} = t_i^2 & t_{j-1} = t_i^1 & t_j = t_{i+1}^2 & t_{j+1} = t_{i+1}^1 \\ d_1^1 = a_1^1 & d_{j-2}^1 = \emptyset & d_{j-1}^1 = a_i^1 & d_j^1 = \emptyset & d_{j+1}^1 = a_{i+1}^1 \\ d_1^2 = a_1^2 & d_{j-2}^2 = a_i^2 & d_{j-1}^2 = \emptyset & d_j^2 = a_{i+1}^2 & d_{j+1}^2 = \emptyset \end{array}$$

$$\begin{array}{lll} t_{n-2} = t_{n_1-1}^1 & t_{n-1} = t_{n_2-1}^2 & t_n = t_{n_1}^1 \\ d_{n-2}^1 = a_{n_1-1}^1 & d_{n-1}^1 = \emptyset & d_n^1 = a_{n_1}^1 \\ d_{n-2}^2 = \emptyset & d_{n-1}^2 = a_{n_2-1}^2 & d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 > t_i^2$, $t_{i+1}^1 > t_{i+1}^2$, $t_i^1 < t_{i+1}^2$ (рис. 3.16и), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП43:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-2} = t_i^2 & t_{j-1} = t_i^1 & t_j = t_{i+1}^2 & t_{j+1} = t_{i+1}^1 \\ d_1^1 = a_1^1 & d_{j-2}^1 = \emptyset & d_{j-1}^1 = a_i^1 & d_j^1 = \emptyset & d_{j+1}^1 = a_{i+1}^1 \\ d_1^2 = a_1^2 & d_{j-2}^2 = a_i^2 & d_{j-1}^2 = \emptyset & d_j^2 = a_{i+1}^2 & d_{j+1}^2 = \emptyset \end{array}$$

$$\begin{array}{lll} t_{n-2} = t_{n_2-1}^2 & t_{n-1} = t_{n_1-1}^1 & t_n = t_{n_1}^1 \\ d_{n-2}^1 = \emptyset & d_{n-1}^1 = a_{n_1-1}^1 & d_n^1 = a_{n_1}^1 \\ d_{n-2}^2 = a_{n_2-1}^2 & d_{n-1}^2 = \emptyset & d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 = t_{i+1}^2$ (рис. 3.16і), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2}$ та правило синхронізації мультимодальних даних БП44:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-1} = t_i^2 & t_j = t_i^1 & t_{j+1} = t_{i+1}^1 & t_{n-1} = t_{n_1-1}^1 \\ d_1^1 = a_1^1 & d_{j-1}^1 = \emptyset & d_j^1 = a_i^1 & d_{j+1}^1 = a_{i+1}^1 & d_{n-1}^1 = a_{n_1-1}^1 \\ d_1^2 = a_1^2 & d_{j-1}^2 = a_i^2 & d_j^2 = a_{i+1}^2 & d_{j+1}^2 = \emptyset & d_{n-1}^2 = a_{n_2-1}^2 \end{array}$$

$$\begin{array}{l} t_n = t_{n_1}^1 \\ d_n^1 = a_{n_1}^1 \\ d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 = t_{i+1}^2$ (рис. 3.16і), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та правило синхронізації мультимодальних даних БП45:

$t_1 = t_1^1$ $t_{j-1} = t_i^2$ $t_j = t_i^1$ $t_{j+1} = t_{i+1}^1$ $t_{n-2} = t_{n_1-1}^1$
$d_1^1 = a_1^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_i^1$ $d_{j+1}^1 = a_{i+1}^1$ $d_{n-2}^1 = a_{n_1-1}^1$
$d_1^2 = a_1^2$ $d_{j-1}^2 = a_i^2$ $d_j^2 = a_{i+1}^2$ $d_{j+1}^2 = \emptyset$ $d_{n-2}^2 = \emptyset$

$t_{n-1} = t_{n_2-1}^2$ $t_n = t_{n_1}^1$
$d_{n-1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 = t_{i+1}^2$ (рис. 3.16і), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = \frac{3i}{2}$, $n = \frac{3(n_1-1)}{2} + 1$ та синхронізація значень кортежів визначається правилом БП46:

$t_1 = t_1^1$ $t_{j-1} = t_i^2$ $t_j = t_i^1$ $t_{j+1} = t_{i+1}^1$ $t_{n-2} = t_{n_2-1}^2$
$d_1^1 = a_1^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_i^1$ $d_{j+1}^1 = a_{i+1}^1$ $d_{n-2}^1 = \emptyset$
$d_1^2 = a_1^2$ $d_{j-1}^2 = a_i^2$ $d_j^2 = a_{i+1}^2$ $d_{j+1}^2 = \emptyset$ $d_{n-2}^2 = a_{n_2-1}^2$

$t_{n-1} = t_{n_1-1}^1$ $t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1$ $d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = \emptyset$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_{i+1}^2$ (рис. 3.16ї), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 = t_{n_2-1}^2$, то $j = 2i, n = 2n_1 - 3$ та правило синхронізації мультимодальних даних БП47:

$t_1 = t_1^1$ $t_{j-2} = t_i^2$ $t_{j-1} = t_{i+1}^2$ $t_j = t_i^1$ $t_{j+1} = t_{i+1}^1$
$d_1^1 = a_1^1$ $d_{j-2}^1 = \emptyset$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_i^1$ $d_{j+1}^1 = a_{i+1}^1$
$d_1^2 = a_1^2$ $d_{j-2}^2 = a_i^2$ $d_{j-1}^2 = a_{i+1}^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = \emptyset$

$t_{n-1} = t_{n_1-1}^1$ $t_n = t_{n_1}^1$
$d_{n-1}^1 = a_{n_1-1}^1$ $d_n^1 = a_{n_1}^1$
$d_{n-1}^2 = a_{n_2-1}^2$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_{i+1}^2$ (рис. 3.16ї), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 < t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП48:

$t_1 = t_1^1$	$t_{j-2} = t_i^2$	$t_{j-1} = t_{i+1}^2$	$t_j = t_i^1$	$t_{j+1} = t_{i+1}^1$
$d_1^1 = a_1^1$	$d_{j-2}^1 = \emptyset$	$d_{j-1}^1 = \emptyset$	$d_j^1 = a_i^1$	$d_{j+1}^1 = a_{i+1}^1$
$d_1^2 = a_1^2$	$d_{j-2}^2 = a_i^2$	$d_{j-1}^2 = a_{i+1}^2$	$d_j^2 = \emptyset$	$d_{j+1}^2 = \emptyset$

$t_{n-2} = t_{n_1-1}^1$	$t_{n-1} = t_{n_2-1}^2$	$t_n = t_{n_1}^1$
$d_{n-2}^1 = a_{n_1-1}^1$	$d_{n-1}^1 = \emptyset$	$d_n^1 = a_{n_1}^1$
$d_{n-2}^2 = \emptyset$	$d_{n-1}^2 = a_{n_2-1}^2$	$d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_{i+1}^2$ (рис. 3.16ї), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$ та $t_{n_1-1}^1 > t_{n_2-1}^2$, то $j = 2i$, $n = 2(n_1 - 1)$ та правило синхронізації мультимодальних даних БП49:

$t_1 = t_1^1$	$t_{j-2} = t_i^2$	$t_{j-1} = t_{i+1}^2$	$t_j = t_i^1$	$t_{j+1} = t_{i+1}^1$
$d_1^1 = a_1^1$	$d_{j-2}^1 = \emptyset$	$d_{j-1}^1 = \emptyset$	$d_j^1 = a_i^1$	$d_{j+1}^1 = a_{i+1}^1$
$d_1^2 = a_1^2$	$d_{j-2}^2 = a_i^2$	$d_{j-1}^2 = a_{i+1}^2$	$d_j^2 = \emptyset$	$d_{j+1}^2 = \emptyset$

$t_{n-2} = t_{n_2-1}^2$	$t_{n-1} = t_{n_1-1}^1$	$t_n = t_{n_1}^1$
$d_{n-2}^1 = \emptyset$	$d_{n-1}^1 = a_{n_1-1}^1$	$d_n^1 = a_{n_1}^1$
$d_{n-2}^2 = a_{n_2-1}^2$	$d_{n-1}^2 = \emptyset$	$d_n^2 = a_{n_2}^2$

Третя група випадків для $e_d(\overline{t^1}, \overline{t^2})$ відрізняється тим, що $\overline{t^1} \lhd \overline{t^2}$, $\left|\overline{t^1}\right| < \left|\overline{t^2}\right|$. Оскільки коефіцієнт *R* для $\left|\overline{t^1}\right|$ та $\left|\overline{t^2}\right|$ може бути довільним, припустимо, що для кожному *i*-го елементи в $\overline{t^1}$ відповідають два елементи в $\overline{t^2}$, за виключенням t_1^1 та $t_{n_1}^1$ що збігаються з t_1^2 та $t_{n_2}^2$ відповідно (рис. 3.17). Це означає, що $n_2\ mod\ 2 = 0$ та $R = \frac{n_1}{2(n_1-1)}$.

Якщо $t_i^1 = t_i^2$ (рис. 3.17а), $\forall\, i \in [2 \ldots (n_2 - 2)]$, $i\ mod\ 2 = 0$, то $n = n_2$ та правило синхронізації мультимодальних даних БП50:

$t_1 = t_1^1$	$t_i = t_i^1$	$t_{i+1} = t_{i+1}^2$	$t_n = t_{n_2}^1$
$d_1^1 = a_1^1$	$d_i^1 = a_i^1$	$d_{i+1}^1 = \emptyset$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_i^2 = a_i^2$	$d_{i+1}^2 = a_{i+1}^2$	$d_n^2 = a_{n_2}^2$

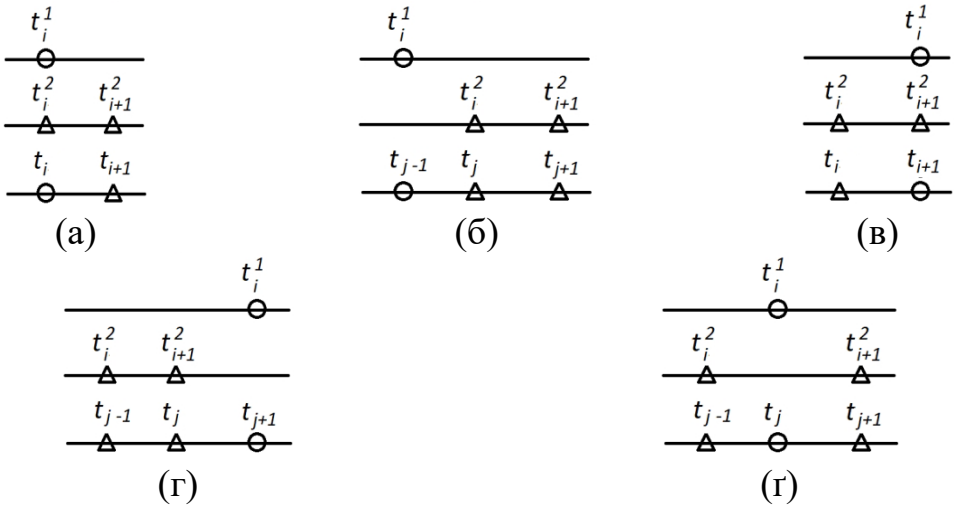

Рис. 3.17. Випадки для $e_d(\overline{t^1}, \overline{t^2})$ при $\overline{t^1} \lhd \overline{t^2}$

Якщо $t_i^1 < t_i^2$ (рис. 3.17б), $\forall\, i \in [2 \ldots (n_2 - 2)]$, $i\ mod\ 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_2}{2} - 1$ та правило синхронізації мультимодальних даних БП51:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-1} = t_i^1 & t_j = t_i^2 & t_{j+1} = t_{i+1}^2 & t_n = t_{n_2}^1 \\ d_1^1 = a_1^1 & d_{j-1}^1 = a_i^1 & d_j^1 = \emptyset & d_{j+1}^1 = \emptyset & d_n^1 = a_{n_1}^1 \\ d_1^2 = a_1^2 & d_{j-1}^2 = \emptyset & d_j^2 = a_i^2 & d_{j+1}^2 = a_{i+1}^2 & d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 = t_{i+1}^2$ (рис. 3.17в), $\forall\, i \in [2 \ldots (n_2 - 2)]$, $i\ mod\ 2 = 0$, то $n = n_2$ та правило синхронізації мультимодальних даних БП52:

$$\begin{array}{llll} t_1 = t_1^1 & t_i = t_i^2 & t_{i+1} = t_i^1 & t_n = t_{n_2}^1 \\ d_1^1 = a_1^1 & d_i^1 = \emptyset & d_{i+1}^1 = a_i^1 & d_n^1 = a_{n_1}^1 \\ d_1^2 = a_1^2 & d_i^2 = a_i^2 & d_{i+1}^2 = a_{i+1}^2 & d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 > t_{i+1}^2$ (рис. 3.17г), $\forall\, i \in [2 \ldots (n_2 - 2)]$, $i\ mod\ 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_2}{2} - 1$ та правило синхронізації мультимодальних даних БП53:

$$\begin{array}{lllll} t_1 = t_1^1 & t_{j-1} = t_i^2 & t_j = t_{i+1}^2 & t_{j+1} = t_i^1 & t_n = t_{n_2}^1 \\ d_1^1 = a_1^1 & d_{j-1}^1 = \emptyset & d_j^1 = \emptyset & d_{j+1}^1 = a_i^1 & d_n^1 = a_{n_1}^1 \\ d_1^2 = a_1^2 & d_{j-1}^2 = a_i^2 & d_j^2 = a_{i+1}^2 & d_{j+1}^2 = \emptyset & d_n^2 = a_{n_2}^2 \end{array}$$

Якщо $t_i^1 > t_i^2$ та $t_i^1 < t_{i+1}^2$ (рис. 3.17ґ), $\forall\, i \in [2 \ldots (n_2 - 2)]$, $i\ mod\ 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_2}{2} - 1$ та правило синхронізації даних БП54:

$t_1 = t_1^1$ $t_{j-1} = t_i^2$ $t_j = t_i^1$ $t_{j+1} = t_{i+1}^2$ $t_n = t_{n_2}^1$
$d_1^1 = a_1^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_i^1$ $d_{j+1}^1 = \emptyset$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-1}^2 = a_i^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = a_{i+1}^2$ $d_n^2 = a_{n_2}^2$

Четверта група випадків для $e_d(\overline{t^1}, \overline{t^2})$ відрізняється тим, що $\overline{t^1} \rhd \overline{t^2}$, тобто $\left|\overline{t^1}\right| > \left|\overline{t^2}\right|$. Подібно випадкам третьої групи ($\overline{t^1} \lhd \overline{t^2}$), коефіцієнт R для $\left|\overline{t^1}\right|$ та $\left|\overline{t^2}\right|$ може бути довільним. Припустимо, що кожному i-му елементу в $\overline{t^2}$ відповідають два елементи в $\overline{t^1}$, за виключенням t_1^2 та $t_{n_2}^2$, які збігаються з t_1^1 та $t_{n_1}^1$ відповідно (рис. 3.18). Тобто $n_1 \, mod \, 2 = 0$ та $R = \frac{n_2}{2(n_2-1)}$.

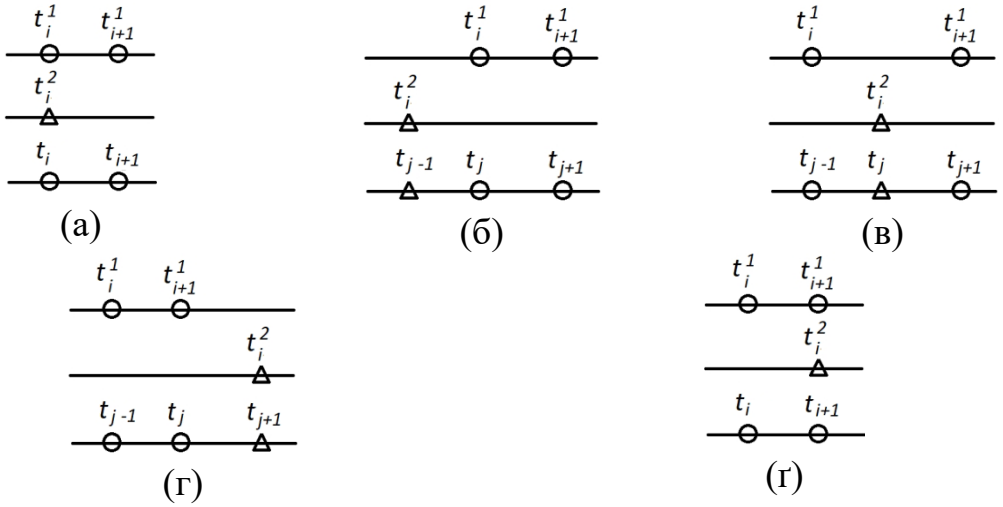

Рис. 3.18. Випадки для $e_d(\overline{t^1}, \overline{t^2})$ при $\overline{t^1} \rhd \overline{t^2}$

Якщо $t_i^1 = t_i^2$ (рис. 3.18а), $\forall \, i \in [2 \ldots (n_1 - 2)]$, $i \, mod \, 2 = 0$, то $n = n_1$ та правило синхронізації мультимодальних даних БП55:

$t_1 = t_1^1$ $t_i = t_i^1$ $t_{i+1} = t_{i+1}^1$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_i^1 = a_i^1$ $d_{i+1}^1 = a_{i+1}^1$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_i^2 = a_i^2$ $d_{i+1}^2 = \emptyset$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 > t_i^2$ (рис. 3.18б), $\forall \, i \in [2 \ldots (n_1 - 2)]$, $i \, mod \, 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та правило синхронізації мультимодальних даних БП56:

$t_1 = t_1^1$ $t_{j-1} = t_i^2$ $t_j = t_i^1$ $t_{j+1} = t_{i+1}^1$ $t_n = t_{n_1}^1$
$d_1^1 = a_1^1$ $d_{j-1}^1 = \emptyset$ $d_j^1 = a_i^1$ $d_{j+1}^1 = a_{i+1}^1$ $d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$ $d_{j-1}^2 = a_i^2$ $d_j^2 = \emptyset$ $d_{j+1}^2 = \emptyset$ $d_n^2 = a_{n_2}^2$

Якщо $t_i^1 < t_i^2$ та $t_{i+1}^1 > t_i^2$ (рис. 3.18в), $\forall\, i \in [2 \ldots (n_1 - 2)],\ i\ mod\ 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та правило синхронізації даних БП57:

$t_1 = t_1^1$	$t_{j-1} = t_i^1$	$t_j = t_i^2$	$t_{j+1} = t_{i+1}^1$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_{j-1}^1 = a_i^1$	$d_j^1 = \emptyset$	$d_{j+1}^1 = a_{i+1}^1$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_{j-1}^2 = \emptyset$	$d_j^2 = a_i^2$	$d_{j+1}^2 = \emptyset$	$d_n^2 = a_{n_2}^2$

Якщо $t_{i+1}^1 < t_i^2$ (рис. 3.18г), $\forall\, i \in [2 \ldots (n_1 - 2)],\ i\ mod\ 2 = 0$, то $j = \frac{3i}{2}$, $n = \frac{3n_1}{2} - 1$ та правило синхронізації мультимодальних даних БП58:

$t_1 = t_1^1$	$t_{j-1} = t_i^1$	$t_j = t_{i+1}^1$	$t_{j+1} = t_i^2$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_{j-1}^1 = a_i^1$	$d_j^1 = a_{i+1}^1$	$d_{j+1}^1 = \emptyset$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_{j-1}^2 = \emptyset$	$d_j^2 = \emptyset$	$d_{j+1}^2 = a_i^2$	$d_n^2 = a_{n_2}^2$

Якщо $t_{i+1}^1 = t_i^2$ (рис. 3.18г), $\forall\, i \in [2 \ldots (n_1 - 2)]$, $i\ mod\ 2 = 0$, то $n = n_1$ та правило синхронізації мультимодальних даних БП59:

$t_1 = t_1^1$	$t_i = t_i^1$	$t_{i+1} = t_{i+1}^1$	$t_n = t_{n_1}^1$
$d_1^1 = a_1^1$	$d_i^1 = a_i^1$	$d_{i+1}^1 = a_{i+1}^1$	$d_n^1 = a_{n_1}^1$
$d_1^2 = a_1^2$	$d_i^2 = \emptyset$	$d_{i+1}^2 = a_i^2$	$d_n^2 = a_{n_2}^2$

Більш загальний випадок синхронізації визначається відношенням *відбувається між*. Він відрізняється тим, що отримання (вимірювань, генерація) однакової кількості значень кортежів даних $\bar{a}^1$ та $\bar{a}^2$ відбувається у один і той самий період часу з моменту α до моменту β, але не одночасно. Відмінність між відношенням *збігається з* та *відбувається між* полягає у способі визначення перших та останніх елементів кортежів мультиобразу, що.

Отже, сформулюємо правило синхронізації БП60, яке полягає у тому, що перші елементи кортежів визначаються відповідно до виразів:

$$t_1 = \begin{cases} t_1^1 \text{ якщо } t_1^1 \le t_1^2 \\ t_1^2 \text{ якщо } t_1^1 > t_1^2 \end{cases}$$

$$d_1^1 = \begin{cases} a_1^1 \text{ якщо } t_1^1 \le t_1^2 \\ \emptyset \text{ якщо } t_1^1 > t_1^2 \end{cases} \quad d_1^2 = \begin{cases} \emptyset \text{ якщо } t_1^1 < t_1^2 \\ a_1^2 \text{ якщо } t_1^1 \ge t_1^2, \end{cases}$$

а останні елементи кортежів визначаються відповідно до виразів:

$$t_n = \begin{cases} t_{n_1}^1 \text{ якщо } t_{n_1}^1 \le t_{n_2}^2 \\ t_{n_2}^2 \text{ якщо } t_{n_1}^1 > t_{n_2}^2 \end{cases}$$

$$d_n^1 = \begin{cases} a_{n_1}^1 \text{ якщо } t_{n_1}^1 \le t_{n_2}^2 \\ \emptyset \text{ якщо } t_{n_1}^1 > t_{n_2}^2 \end{cases} \quad d_n^2 = \begin{cases} \emptyset \text{ якщо } t_{n_1}^1 < t_{n_2}^2 \\ a_{n_2}^2 \text{ якщо } t_{n_1}^1 \ge t_{n_2}^2 \end{cases}.$$

Решта елементів визначається згідно з правилами БП1-БП59. Таким чином, у випадку відношення *відбувається між* при виконання синхронізації мультимодальних даних мультиобразів застосовується не окреме правило синхронізації, а шаблон синхронізації, що складається з правила синхронізації БП60 для визначення перших і останніх елементів та одного з правил БП1-БП59 для визначення всіх інших елементів.

Для зручності будемо використовувати [9] наступний формат запису шаблону синхронізації (ШС), який визначимо за допомогою розширеної форми Бекуса-Наура:

шаблон_синхронізації = [“ШС”, *номер_шаблона*, “=”,] (“УП” | “БП” | “НП”), *номер_правила*, [“(“, *номер_елемента*, (“,” | “–”), *номер_елемента*, ”)” {“&”, (“УП” | “БП” | “НП”), *номер_правила*, “(“, *номер_елемента*, (“,” | “–”), *номер_елемента*, ”)”}]

Розглянемо приклад запису шаблону синхронізації, визначеного згідно із запропонованим форматом: БП60(1,*n*) & БП15(2–(*n*-1)). Цей запис означає, що шаблон синхронізації складається з двох правил: правила БП60, яке застосовується для синхронізації перших та останніх елементів (їх індекси $i = 1$ та $i = n$ відповідно), та правила БП15, яке застосовується для синхронізації всіх інших елементів (ці елементи мають індекси $i = [2 \dots (n-1)]$).

Відповідно до визначеного формату шаблон синхронізації може мати власну назву, крім того, він може складатись з одного чи декількох правил синхронізації різних типів.

Таким чином, шаблон синхронізації для відношення *відбувається між* має вигляд: ШС1=БП60(1,*n*)&*x*(2–(*n*-1)), де *x* – це деяке правило з базових правил БП1-БП59.

Визначимо правило синхронізації мультимодальних даних для випадку, коли отримання значень кортежу даних $\bar{a}^1$ передує отриманню значень кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *передує*, що визначається (2.56). Отже, якщо $b_d(\bar{t}^1, \bar{t}^2) = true$, то $\forall\, i_1 \in [1 \dots n_1]$, $\forall\, i_2 \in [1 \dots n_2]$ та правило синхронізації мультимодальних даних БП61:

$$t_{i_1} = t^1_{i_1} \qquad t_{i_2+n_1} = t^2_{i_2}$$
$$d^1_{i_1} = a^1_{i_1} \qquad d^1_{i_2+n_1} = \emptyset$$
$$d^2_{i_1} = \emptyset \qquad d^2_{i_2+n_1} = a^2_{i_2}$$

Випадок, коли отримання значень кортежу даних $\bar{a}^1$ відбувається після отримання значень кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *настає після*. Отже, якщо $bi_d(\bar{t}^1, \bar{t}^2) = true$, то $\forall\, i_1 \in [1 \dots n_1]$, $\forall\, i_2 \in [1 \dots n_2]$ та правило синхронізації мультимодальних даних БП62:

$$t_{i_2} = t^2_{i_2} \qquad t_{i_1+n_2} = t^1_{i_1}$$
$$d^1_{i_2} = \emptyset \qquad d^1_{i_1+n_2} = a^1_{i_1}$$
$$d^2_{i_2} = a^2_{i_2} \qquad d^2_{i_1+n_2} = \emptyset$$

Випадок, коли отримання значень кортежу даних $\bar{a}^1$ передує отриманню значень кортежу даних $\bar{a}^2$, проте момент отримання останнього значення кортежу даних $\bar{a}^1$ збігається з моментом отримання першого значення кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *стикається з початком*. Таким чином, якщо $m_d(\bar{t}^1, \bar{t}^2) = true$, то $\forall\, i_1 \in [1 \dots (n_1 - 1)]$, $\forall\, i_2 \in [2 \dots n_2]$ та правило синхронізації мультимодальних даних БП63:

$$t_{i_1} = t^1_{i_1} \qquad t_{n_1} = t^1_{n_1} \qquad t_{i_2+n_1-1} = t^2_{i_2}$$
$$d^1_{i_1} = a^1_{i_1} \qquad d^1_{n_1} = a^1_{n_1} \qquad d^1_{i_2+n_1-1} = \emptyset$$
$$d^2_{i_1} = \emptyset \qquad d^2_{n_1} = a^2_1 \qquad d^2_{i_2+n_1-1} = a^2_{i_2}$$

Випадок, коли отримання значень кортежу даних $\bar{a}^1$ відбувається після отримання значень кортежу даних $\bar{a}^2$, проте момент отримання першого значення кортежу даних $\bar{a}^1$ збігається з моментом отримання останнього значення кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *стикається з кінцем*. Отже, якщо $mi_d(\overline{t^1}, \overline{t^2}) = true$, то $\forall\, i_1 \in [2 \dots n_1]$, $\forall\, i_2 \in [1 \dots (n_2 - 1)]$ та правило синхронізації мультимодальних даних БП64:

$$t_{i_2} = t^2_{i_2} \qquad t_{n_2} = t^2_{n_2} \qquad t_{i_1+n_2-1} = t^1_{i_1}$$
$$d^1_{i_2} = \emptyset \qquad d^1_{n_2} = a^1_1 \qquad d^1_{i_1+n_2-1} = a^1_{i_1}$$
$$d^2_{i_2} = a^2_{i_2} \qquad d^2_{n_2} = a^2_{n_2} \qquad d^2_{i_1+n_2-1} = \emptyset$$

Випадок, коли отримання значень кортежу даних $\bar{a}^1$ передує отриманню значень кортежу даних $\bar{a}^2$, проте отримання *K* останніх значень кортежу даних $\bar{a}^1$ збігається з отриманням *K* перших значень кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *перекриває*. Тоді, якщо $o_d(\overline{t^1}, \overline{t^2}) = true$, то $\forall\, i_1 \in [1 \dots (n_1 - K)]$, $\forall\, i_2 \in [(K+1) \dots n_2]$, $\forall k \in [1 \dots K]$ та правило синхронізації мультимодальних даних БП65:

$$t_{i_1} = t^1_{i_1} \qquad t_{n_1+k-K} = t^2_k \qquad t_{n_1+i_2-K} = t^2_{i_2}$$
$$d^1_{i_1} = a^1_{i_1} \qquad d^1_{n_1+k-K} = a^1_{n_1+k-K} \qquad d^1_{n_1+i_2-K} = \emptyset$$
$$d^2_{i_1} = \emptyset \qquad d^2_{n_1+k-K} = a^2_k \qquad d^2_{n_1+i_2-K} = a^2_{i_2}$$

Випадок, коли отримання значень кортежу даних $\bar{a}^1$ відбувається після отримання значень кортежу даних $\bar{a}^2$, проте отримання *K* перших значень кортежу даних $\bar{a}^1$ збігається з отриманням *K* останніх значень кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *перекривається*. Таким чином, якщо $oi_d(\overline{t^1}, \overline{t^2}) = true$, то $\forall\, i_1 \in [(K+1) \dots n_1]$, $\forall\, i_2 \in [1 \dots (n_2 - K)]$, $\forall k \in [1 \dots K]$ та правило синхронізації мультимодальних даних БП66:

$$t_{i_2} = t^2_{i_2} \qquad t_{n_2+k-K} = t^1_k \qquad t_{i_1+n_2-K} = t^1_{i_1}$$
$$d^1_{i_2} = \emptyset \qquad d^1_{n_2+k-K} = a^1_k \qquad d^1_{i_1+n_2-K} = a^1_{i_1}$$
$$d^2_{i_2} = a^2_{i_2} \qquad d^2_{n_2+k-K} = a^2_{n_2+k-K} \qquad d^2_{i_1+n_2-K} = \emptyset$$

Якщо отримання значень кортежу даних $\bar{a}^1$ починається пізніше та завершується раніше отримання значень кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *відбувається під час*. Отже, якщо $d_d(\bar{t}^1, \bar{t}^2) = true$, то $\bar{t}^2 = \overline{t^{2,\mathrm{I}}} \cup \overline{t^{2,\mathrm{II}}} \cup \overline{t^{2,\mathrm{III}}}$, такі, що $b_d(\overline{t^{2,\mathrm{I}}}, \bar{t}^1), w_d^{(t_1^1, t_{n_1}^1)}(\overline{t^{2,\mathrm{II}}}, \bar{t}^1), \quad bi_d(\overline{t^{2,\mathrm{III}}}, \bar{t}^1)$. Тоді, $\forall\, k_1 \in [1 \ldots K_1], \quad \forall\, k_2 \in [K_2 \ldots n_2]$, таких, що $t_{K_1}^2 < t_1^1$ та $t_{K_2}^2 > t_{n_1}^1$, правило синхронізації БП67 для отримання перших значень $t_{k_1}(t_{k_1} \in \overline{t^{2,\mathrm{I}}})$, $d_{k_1}^1$, $d_{k_1}^2$ та останніх значень t_{k_2} $(t_{k_2} \in \overline{t^{2,\mathrm{III}}})$, $d_{k_2}^1$, $d_{k_2}^2$ визначається виразами:

$$t_{k_1} = t_{k_1}^2 \qquad t_{n-n_2+k_2} = t_{k_2}^2$$
$$d_{k_1}^1 = \emptyset \qquad d_{n-n_2+k_2}^1 = \emptyset$$
$$d_{k_1}^2 = a_{k_1}^2 \qquad d_{n-n_2+k_2}^2 = a_{k_2}^2$$

Решту елементів ($t_k \in \overline{t^{2,\mathrm{II}}}$, d_k^1, d_k^2) кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ можна отримати згідно з правилами синхронізації БП1-БП59, тобто синхронізація даних мультиобразів у випадку відношення *відбувається під час* задаються шаблоном синхронізації ШС2=БП67(1,*n*)&*x*(2–(*n*-1)), де *x* – це деяке правило з базових правил БП1-БП59. Довжина кортежу $\bar{t}$ складає $n = K_1 + m + n_2 - K_2 + 1$, де m є довжиною кортежу, отриманого в результаті синхронізації кортежів $\overline{t^{2,\mathrm{II}}}$ та $\bar{t}^1$.

Випадок, коли отримання значень кортежу даних $\bar{a}^1$ починається раніше та завершується пізніше отримання значень кортежу даних $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *містить*. Отже, якщо $di_d(\bar{t}^1, \bar{t}^2)$, то $\bar{t}^1 = \overline{t^{1,\mathrm{I}}} \cup \overline{t^{1,\mathrm{II}}} \cup \overline{t^{1,\mathrm{III}}}$, такі, що $b_d(\overline{t^{1,\mathrm{I}}}, \bar{t}^2), w_d^{(t_1^2, t_{n_2}^2)}(\overline{t^{1,\mathrm{II}}}, \bar{t}^2)$, $bi_d(\overline{t^{1,\mathrm{III}}}, \bar{t}^2)$. Тоді, $\forall\, k_1 \in [1 \ldots K_1]$, $\forall\, k_2 \in [K_2 \ldots n_1]$, таких, що $t_{K_1}^1 < t_1^2$ та $t_{K_2}^1 > t_{n_2}^2$, правило синхронізації БП68 для отримання перших значень $t_{k_1}(t_{k_1} \in \overline{t^{1,\mathrm{I}}})$, $d_{k_1}^1$, $d_{k_1}^2$ та останніх значень t_{k_2} $(t_{k_2} \in \overline{t^{1,\mathrm{III}}})$, $d_{k_2}^1$, $d_{k_2}^2$ визначається виразам:

$$t_{k_1} = t^1_{k_1} \qquad t_{n-n_1+k_2} = t^1_{k_2}$$
$$d^1_{k_1} = a^1_{k_1} \qquad d^1_{n-n_1+k_2} = a^1_{k_2}$$
$$d^2_{k_1} = \emptyset \qquad d^2_{n-n_1+k_2} = \emptyset$$

Решту елементів ($t_k \in \overline{t^{1,\text{II}}}$, d^1_k, d^2_k) кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ можна отримати згідно з правилами синхронізації БП1-БП59, тобто синхронізація даних мультиобразів у разі відношення *містить* задаються шаблоном синхронізації ШС3=БП68(1,*n*)&*x*(2–(*n*-1)), де *x* – це деяке правило з правил БП1-БП59. Довжина кортежу $\bar{t}$ складає $n = K_1 + m + n_1 - K_2 + 1$, де m є довжиною кортежу, отриманого в результаті синхронізації $\overline{t^{1,\text{II}}}$ та $\overline{t^2}$.

Якщо отримання значень кортежів даних $\bar{a}^1$ та $\bar{a}^2$ починається одночасно, проте отримання значень кортежу $\bar{a}^1$ закінчується раніше, ніж отримання значень кортежу $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *починає*, що визначається (2.65). Отже, якщо $s_d(\overline{t^1}, \overline{t^2})$, то $\overline{t^2} = \overline{t^{2,\text{I}}} \cup \overline{t^{2,\text{II}}}$, такі, що $w_d^{(t^1_1, t^1_{n_1})}(\overline{t^{2,\text{I}}}, \overline{t^1})$, $bi_d(\overline{t^{2,\text{II}}}, \overline{t^1})$. Тоді, $\forall\, k_2 \in [K \ldots n_2]$, таких, що $t^2_K > t^1_{n_1}$, правило синхронізації БП69 для отримання першого значення t_1 ($t_1 \in \overline{t^{2,\text{I}}}$), d^1_1, d^2_1 та останніх значень t_{k_2} ($t_{k_2} \in \overline{t^{2,\text{II}}}$), $d^1_{k_2}$, $d^2_{k_2}$ результуючих кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ визначається виразами:

$$t_1 = t^1_1 \qquad t_{n-n_2+k_2} = t^2_{k_2}$$
$$d^1_1 = a^1_1 \qquad d^1_{n-n_2+k_2} = \emptyset$$
$$d^2_1 = a^2_1 \qquad d^2_{n-n_2+k_2} = a^2_{k_2}$$

Решту елементів ($t_{k_1} \in \overline{t^{2,\text{I}}}$, $d^1_{k_1}$, $d^2_{k_1}$, $\forall\, k_1 \in [2 \ldots m]$, де m є довжиною кортежу, отриманого в результаті синхронізації кортежів $\overline{t^{2,\text{I}}}$ та $\overline{t^1}$) кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ можна отримати згідно з правилами синхронізації БП1-БП59, тобто синхронізація мультимодальних даних мультиобразів у випадку відношення *починає* задаються шаблоном синхронізації ШС4=БП69(1,*n*) & *x*(2–(*n*-1)), де *x* – це деяке правило з базових правил БП1-БП59. Довжина кортежу $\bar{t}$ складає $n = m + n_2 - K + 1$.

Випадок, коли отримання значень кортежів даних $\bar{a}^1$ та $\bar{a}^2$ починається одночасно, проте отримання значень кортежу $\bar{a}^1$ закінчується пізніше, ніж отримання значень кортежу $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *починається*. Отже, якщо $si_d(\bar{t}^1, \bar{t}^2)$, то $\bar{t}^1 = \overline{t^{1,\mathrm{I}}} \cup \overline{t^{1,\mathrm{II}}}$, такі, що $w_d^{(t_1^2, t_{n_2}^2)}(\overline{t^{1,\mathrm{I}}}, \bar{t}^2)$, $bi_d(\overline{t^{1,\mathrm{II}}}, \bar{t}^2)$. Тоді, $\forall\, k_2 \in [K \ldots n_1]$, таких, що $t_K^1 > t_{n_2}^2$, правило синхронізації БП70 для отримання першого значення t_1 ($t_1 \in \overline{t^{1,\mathrm{I}}}$), d_1^1, d_1^2 та останніх значень t_{k_2} ($t_{k_2} \in \overline{t^{1,\mathrm{II}}}$), $d_{k_2}^1$, $d_{k_2}^2$ результуючих кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ визначається виразами:

$$t_1 = t_1^1 \qquad t_{n-n_1+k_2} = t_{k_2}^1$$
$$d_1^1 = a_1^1 \qquad d_{n-n_1+k_2}^1 = a_{k_2}^1$$
$$d_1^2 = a_1^2 \qquad d_{n-n_1+k_2}^2 = \emptyset$$

Решту елементів ($t_{k_1} \in \overline{t^{1,\mathrm{I}}}$, $d_{k_1}^1$, $d_{k_1}^2$, $\forall\, k_1 \in [2 \ldots m]$, де m є довжиною кортежу, отриманого в результаті синхронізації кортежів $\overline{t^{1,\mathrm{I}}}$ та $\bar{t}^2$) кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ можна отримати згідно з правилами синхронізації БП1-БП59, тобто синхронізація мультимодальних даних мультиобразів у випадку відношення *починається* задається шаблоном синхронізації ШС5=БП70(1,*n*) & *x*(2–(*n*-1)), де *x* – це деяке правило з базових правил БП1-БП59. Довжина кортежу $\bar{t}$ складає $n = m + n_1 - K + 1$.

Випадок, коли отримання значень кортежів даних $\bar{a}^1$ та $\bar{a}^2$ закінчується одночасно, проте отримання значень кортежу $\bar{a}^1$ починається пізніше, ніж отримання значень кортежу $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *закінчує*. Отже, якщо $f_d(\bar{t}^1, \bar{t}^2)$, то $\bar{t}^2 = \overline{t^{2,\mathrm{I}}} \cup \overline{t^{2,\mathrm{II}}}$, такі, що $b_d(\overline{t^{2,\mathrm{I}}}, \bar{t}^1)$, $w_d^{(t_1^1, t_{n_1}^1)}(\overline{t^{2,\mathrm{II}}}, \bar{t}^1)$. Тоді, $\forall\, k_1 \in [1 \ldots K]$, таких, що $t_K^2 < t_1^1$, правило синхронізації БП71 для отримання перших значень t_{k_1}($t_{k_1} \in \overline{t^{2,\mathrm{I}}}$), $d_{k_1}^1$, $d_{k_1}^2$ та останнього значення t_n ($t_n \in \overline{t^{2,\mathrm{II}}}$), d_n^1, d_n^2 результуючих кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ визначається виразами:

$t_{k_1} = t^2_{k_1}$ $\quad\quad$ $t_n = t^1_{n_1}$

$d^1_{k_1} = \emptyset$ $\quad\quad$ $d^1_n = a^1_{n_1}$

$d^2_{k_1} = a^2_{k_1}$ $\quad\quad$ $d^2_n = a^2_{n_2}$

Решту елементів $(t_{k_2} \in \overline{t^{2,\text{II}}},\ d^1_{k_2},\ d^2_{k_2},\ \forall\, k_2 \in [(K+1) \ldots (n-1)]$, де довжина кортежу $\bar{t}$ складає $n = K + m$; m є довжиною кортежу, отриманого в результаті синхронізації кортежів $\overline{t^{2,\text{II}}}$ та $\overline{t^1}$) кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ можна отримати згідно з правилами синхронізації БП1-БП59, тобто синхронізація даних мультиобразів у разі відношення *закінчує* задаються шаблоном ШС6=БП71(1,*n*) & *x*(2–(*n*-1)), де *x* – це деяке правило з базових правил БП1-БП59.

Випадок, коли отримання значень кортежів даних $\bar{a}^1$ та $\bar{a}^2$ закінчується одночасно, проте отримання значень кортежу $\bar{a}^1$ починається раніше, ніж отримання значень кортежу $\bar{a}^2$, означає, що дискретні інтервали $\bar{t}^1$ та $\bar{t}^2$ пов'язані відношенням *закінчується*. Отже, якщо $fi_d(\overline{t^1}, \overline{t^2})$, то $\overline{t^1} = \overline{t^{1,\text{I}}} \cup \overline{t^{1,\text{II}}}$, такі, що $b_d(\overline{t^{1,\text{I}}}, \overline{t^2}), w_d^{(t^2_1, t^2_{n_2})}(\overline{t^{1,\text{II}}}, \overline{t^2})$. Тоді, $\forall\, k_1 \in [1 \ldots K]$, таких, що $t^1_K < t^2_1$, правило синхронізації БП72 для отримання перших значень $t_{k_1}(t_{k_1} \in \overline{t^{1,\text{I}}})$, $d^1_{k_1}$, $d^2_{k_1}$ та останнього значення t_n $(t_n \in \overline{t^{1,\text{II}}})$, d^1_n, d^2_n результуючих кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ визначається виразами:

$t_{k_1} = t^1_{k_1}$ $\quad\quad$ $t_n = t^2_{n_2}$

$d^1_{k_1} = a^1_{k_1}$ $\quad\quad$ $d^1_n = a^1_{n_1}$

$d^2_{k_1} = \emptyset$ $\quad\quad$ $d^2_n = a^2_{n_2}$

Решту елементів $(t_{k_2} \in \overline{t^{1,\text{II}}},\ ,d^1_{k_2},\ d^2_{k_2})$, $\forall\, k_2 \in [(K+1) \ldots (n-1)]$, де довжина кортежу $\bar{t}$ складає $n = K + m$; m є довжиною кортежу, отриманого в результаті синхронізації кортежів $\overline{t^{1,\text{II}}}$ та $\overline{t^2}$) кортежів $\bar{t}$, $\overline{d^1}$ та $\overline{d^2}$ можна отримати згідно з правилами синхронізації БП1-БП59, тобто синхронізація даних мультиобразів у випадку відношення *закінчується* задаються шаблоном синхронізації ШС7=БП72(1,*n*) & *x*(2–(*n*-1)), де *x* – це деяке правило з базових правил БП1-БП59.

Оскільки існує значна кількість випадків точної синхронізації даних, то для спрощення процедури синхронізації мультимодальних даних у задачах, де визначення часу з деякою похибкою є припустимим, доцільно застосовувати нечітку синхронізацію даних [9].

Контрольні запитання

1. У чому полягає властивість мультимодальності даних цифрового двійника?
2. У чому полягає властивість темпоральності даних цифрового двійника?
3. Що таке агрегат в алгебраїчній системі агрегатів?
4. Якими є визначальні риси агрегата?
5. Що відіграє роль нейтрального елементу в алгебраїчній системі агрегатів?
6. Що визначає сумісність агрегатів і чому це важливо?
7. За допомогою яких операцій алгебраїчної системи агрегатів приховано сумісні агрегати можуть бути перетворені на сумісні?
8. У чому полягає відмінність між операціями виключного перерізу та перерізу агрегатів?
9. Чому операції впорядкування є особливо важливими в алгебраїчній системі агрегатів?
10. У чому полягає суттєва відмінність інтервальних відношень в алгебраїчній системі агрегатів від відношень інтервальної алгебри Аллена?

Завдання для самостійної підготовки

1. Сформуйте набори мультимодальних даних у вигляді агрегатів та проаналізуйте їхню сумісність.
2. Сформуйте набори мультимодальних даних у вигляді агрегатів та застосуйте до них логічні операції алгебраїчної системи агрегатів.
3. Сформуйте набори мультимодальних даних у вигляді агрегатів та застосуйте до них операції впорядкування алгебраїчної системи агрегатів.
4. Сформуйте набори мультимодальних даних у вигляді агрегатів та проаналізуйте їхні відношення.

5. Проаналізуйте шаблони синхронізації з точки зору модальностей даних, типів давачів цих даних та задач синхронізації потоків таких даних.

4. СТАНДАРТИЗАЦІЯ ПРОЄКТУВАННЯ ЦИФРОВИХ ДВІЙНИКІВ

У розділі розглянуто стандартизацію проєктування цифрових двійників як ключову передумову їх семантичної інтероперабельності, масштабованості та повторного використання. Проаналізовано підходи до семантичного моделювання з використанням мови DTDL, що забезпечує формалізований опис структури, поведінки та взаємозв'язків цифрових двійників у вигляді графів знань. Пояснено концепцію оболонки адміністрування активу (Asset Administration Shell, AAS) як стандартизованого інформаційного ядра цифрового двійника, що уніфікує представлення даних про актив протягом усього життєвого циклу. Наведено інформацію про стандарт OPC UA як універсальній сервіс-орієнтованій архітектурі безпечного та семантично збагаченого обміну даними між фізичними пристроями та програмними системами. Розділ формує методологічну основу для проєктування відкритих, сумісних і довготривало еволюційних програмних систем цифрових двійників.

4.1. Семантичне моделювання та специфікація DTDL

Для побудови інтелектуальних систем на основі цифрових двійників необхідна семантична інтероперабельність. Для вирішення цієї задачі використовується DTDL (Digital Twins Definition Language) – мова моделювання, призначена для опису моделей цифрових двійників. DTDL надає можливість описувати фізичні об'єкти, їхні властивості, телеметрію, компоненти та взаємозв'язки між ними, формуючи у такий спосіб граф знань [56].

DTDL – це мова опису схеми (schema definition language), що ґрунтується на стандарті JSON-LD (JavaScript Object Notation for Linked Data), розробленому консорціумом W3C [75]. Це означає, що будь-яка модель DTDL є JSON-документом, який може бути оброблений будь-яким сучасним парсером, але при цьому містить семантичний контекст.

Ключовими характеристиками DTDL є відкритість, об'єктно-орієнтованість та графоцентричність. Відкритість полягає у тому, що специфікація DTDL є відкритою і розвивається спільнотою на GitHub, хоча початково була ініційована компанією Microsoft. Об'єктно-орієнтований підхід полягає у тому, що DTDL оперує поняттями, близькими до ООП (інтерфейси, наслідування, компоненти), що знижує поріг входження для розробників програмного забезпечення. Графоцентричність визначається тим, що на відміну

від реляційних схем DTDL спроєктована для опису топологій – мереж взаємопов'язаних сутностей, що дає змогу будувати графи знань.

Атомарною одиницею моделювання в DTDL є інтерфейс (Interface). Інтерфейс описує можливості цифрового двійника. Кожен інтерфейс ідентифікується унікальним ідентифікатором моделі – DTMI (Digital Twin Model Identifier). Формат DTMI:

`dtmi:<домен>:<унікальний_ідентифікатор>;<версія>`

Інтерфейс містить набір елементів, які поділяються на п'ять основних типів. Розглянемо їх докладніше.

1. Тип «Телеметрія» (Telemetry) описує потік даних, що генерується фізичним об'єктом (давачами) і передається у цифровий двійник. Семантика цього типу полягає у визначенні того, що відбувається з об'єктом у поточний момент. Телеметрія не зберігається як стан. Це потік подій, який обробляється або архівується.
2. Тип «Властивість» (Property) описує стан двійника. Властивості мають персистентне сховище (останнє відоме значення). Семантика цього типу полягає у визначенні того, яким є стан системи. Підтримуються два варіанти властивості:
   - *read-only:* інформація, що надходить від пристрою (наприклад, серійний номер, версія прошивки);
   - *writable:* параметри конфігурації, які можна змінити з боку цифрового двійника (наприклад, цільова температура `targetTemperature`).
3. Тип «Команда» (Command) описує функцію або дію, яку можна виконати над цифровим двійником. Семантика цього типу полягає у визначенні того, що потрібно виконати. У синхронних системах це виклик RPC (Remote Procedure Call). Прикладами команд можуть слугувати `reboot()`, `openValve(45)`.
4. Тип «Відносини» (Relationship) описує, як один цифровий двійник або складова двійника пов'язані з іншими. Наприклад, інтерфейс `Room` має

відношення `contains` до інтерфейсу `Thermostat`. Відносини в DTDL завжди є спрямованими.

5. Тип «Компонент» (Component) надає можливість вбудовувати один інтерфейс в інший як складову частину. Наприклад, смартфон (основний інтерфейс) має камеру (компонент) і GPS-модуль (компонент).

Для оптимізації аналізу даних у DTDL застосовуються семантичні анотації. Наприклад, для типізації значень телеметрії використовуються такі визначення, як *Acceleration* (прискорення), *Density* (густина), *Force* (сила) тощо.

У лістингу 4.1 наведено приклад застосування мови DTDL для опису промислового охолоджувача.

Лістинг 4.1. Опис промислового охолоджувача

```
    {
      "@context": "dtmi:dtdl:context;1",
      "@id": "dtmi:com:university:hvac:Chiller;1",
      "@type": "Interface",
      "displayName": "Industrial Chiller Unit",
      "description": "Digital Twin model for a water-cooled
chiller.",
      "contents": [
        {
          "@type": "Telemetry",
          "name": "currentTemperature",
          "schema": "double",
          "unit": "degreeCelsius",
          "displayName": "Current Temperature"
        },
        {
          "@type": "Property",
          "name": "fanSpeed",
          "schema": "integer",
          "writable": true,
          "displayName": "Target Fan Speed (RPM)"
        },
        {
          "@type": "Property",
          "name": "maintenanceDate",
          "schema": "date",
          "writable": false
        },
        {
          "@type": "Command",
          "name": "emergencyStop",
```

```
      "displayName": "Emergency Stop Command"
    },
    {
      "@type": "Relationship",
      "name": "feeds",
      "target": "dtmi:com:university:hvac:AirHandler;1",
      "displayName": "Feeds Coolant To"
    }
  ]
}
```

DTDL є строго типізованою мовою, що забезпечує цілісність даних.

Однією з можливостей DTDL є механізм наслідування, реалізований через ключове слово `extends`. Це дозволяє будувати ієрархії онтологій, від загального до конкретного, що відповідає принципам об'єктно-орієнтованого проєктування.

Цифровий двійник може наслідувати до двох інших інтерфейсів (множинне наслідування обмежене для уникнення неоднозначності, що виникає при множинному наслідуванні (Diamond Problem), хоча глибина наслідування не обмежена). Приклад коду з наслідуванням наведено на Лістингу 4.2.

Лістинг 4.2. Фрагмент коду з наслідуванням

```
{
  "@context": "dtmi:dtdl:context;1",
  "@id": "dtmi:com:example:TemperatureSensor;1",
  "@type": "Interface",
  "extends": "dtmi:com:example:Sensor;1",
  "contents": [ ... ]
}
```

Це уможливлює створення поліморфних запитів. Наприклад, запит «Знайти всі `Sensor`, що мають помилки» поверне і базові давачі, і давачі температури, оскільки вони всі є нащадками `Sensor`.

Для ілюстрації принципів модульності та повторного використання коду в DTDL, спроєктуємо цифровий двійник 6-осьового промислового маніпулятора, який характеризується наступним чином:

1) має власний стан (активний/помилка);
2) передає телеметрію (кути нахилу осей);

3) містить вкладений компонент – захват (Gripper), який є окремою змінною деталлю;
4) пов’язаний з лінією безпеки (Safety Zone).

Створення цифрового двійника почнемо з визначення компонента «Захват» (Gripper). Опишемо його як окремий інтерфейс (Лістинг 4.3). Це дозволить у подальшому використовувати цю модель для інших маніпуляторів, що забезпечить повторне використання (reusability).

Лістинг 4.3. JSON-документ з описом компонента «Захват»

```
/* Filename: Gripper.json */
{
  "@context": "dtmi:dtdl:context;1",
  "@id": "dtmi:com:university:robotics:Gripper;1",
  "@type": "Interface",
  "displayName": "Vacuum Gripper",
  "contents": [
    {
      "@type": "Property",
      "name": "isGrasping",
      "schema": "boolean",
      "displayName": "Grasping State",
      "description": "True if the gripper is currently holding an
object."
    },
    {
      "@type": "Telemetry",
      "name": "airPressure",
      "schema": "double",
      "unit": "kiloPascal",
      "displayName": "Vacuum Pressure"
    },
    {
      "@type": "Command",
      "name": "release",
      "displayName": "Release Object Command"
    }
  ]
}
```

Далі розробимо модель маніпулятора (Robotic Arm), який включає в себе модель захвату через ключове слово `Component` (Лістинг 4.4).

Лістинг 4.4. JSON-документ з описом маніпулятора

```
/* Filename: RoboticArm.json */
{
  "@context": "dtmi:dtdl:context;1",
  "@id": "dtmi:com:university:robotics:6AxisRobot;1",
  "@type": "Interface",
  "displayName": "6-Axis Industrial Robot",
  "contents": [

 /* Властивість – серійний номер (незмінний) */
    {
      "@type": "Property",
      "name": "serialNumber",
      "schema": "string",
      "writable": false
    },

    /* Телеметрія (Object schema) */
    {
      "@type": "Telemetry",
      "name": "jointPositions",
      "displayName": "Joint Angles",
      "schema": {
        "@type": "Object",
        "fields": [
          {"name": "j1", "schema": "double"},
          {"name": "j2", "schema": "double"},
          {"name": "j3", "schema": "double"},
          {"name": "j4", "schema": "double"},
          {"name": "j5", "schema": "double"},
          {"name": "j6", "schema": "double"}
        ]
      }
    },

    /* Підключення моделі захвату */
    {
      "@type": "Component",
      "name": "endEffector",
      "schema": "dtmi:com:university:robotics:Gripper;1",
      "description": "The tool attached to the robot flange."
    },

    /* Зв'язок з зоною безпеки */
    {
      "@type": "Relationship",
      "name": "assignedToZone",
      "target": "dtmi:com:university:factory:SafetyZone;1",
      "displayName": "Located In Zone"
    }
  ]
}
```

Програмна реалізація розроблених моделей може бути здійснена з використанням Azure Digital Twins SDK для створення екземпляра цифрового двійника та оновлення телеметрії вкладеного компонента. Приклад такої реалізації наведено у Лістингу 4.5.

Лістинг 4.5. Псевдокод на основі Azure Digital Twins SDK

```
// Створення цифрового двійника промислового маніпулятора

var twinData = new BasicDigitalTwin
{
    Id = "Robot_Cell_05",
    Metadata = { ModelId =
"dtmi:com:university:robotics:6AxisRobot;1" },
    Contents =
    {
        { "serialNumber", "KR-2024-XA" }
    }
};
await client.CreateOrReplaceDigitalTwinAsync("Robot_Cell_05",
twinData);

// ... (пройшов час, маніпулятор схопив деталь) ...

// Оновлення стану вкладеного компонента (Gripper)

await client.UpdateDigitalTwinAsync("Robot_Cell_05",
updateOperation);

// Відправка телеметрії тиску повітря (від компонента)

var telemetryData = new Dictionary<string, object>
{
    { "airPressure", 85.5 } // значення в кПа
};

// Відправка повідомлення з вказанням ID компонента

await client.PublishComponentTelemetryAsync(
    "Robot_Cell_05",
    "endEffector",
    Guid.NewGuid().ToString(),
    JsonSerializer.Serialize(telemetryData)
);
```

Цей приклад демонструє три важливі концепції. По-перше, композиція: тиск повітря не описаний всередині маніпулятора, оскільки він був описаний в `Gripper`, тож якщо у подальшому виникне необхідність змінити захват на

інший механізм, буде потрібно лише змінити посилання в полі `schema` компонента, не переписуючи всю модель маніпулятора. По-друге, структурована телеметрія: використання типу `Object` для `jointPositions` гарантує, що кути всіх 6 осей прийдуть в одному пакеті (Time-aligned data), що є критичним для забезпечення точності кінематичної моделі. І по-третє, маршрутизація даних: у прикладі коду видно, що можна надсилати телеметрію адресно конкретному компоненту (`endEffector`), що дозволяє ізолювати логіку оброблення даних.

4.2. Стандартизована оболонка адміністрування активу

Оболонка адміністрування активу (Asset Administration Shell – AAS) [6, 26, 30] є реалізацією цифрового двійника згідно з архітектурною моделлю RAMI (Reference Architectural Model Industrie) [76]. AAS – це стандартизоване цифрове представлення фізичного або логічного ресурсу (активу), яке містить усю інформацію про нього, від креслень до даних про утилізацію, та надає універсальні інтерфейси для взаємодії з цим активом у кіберфізичному середовищі. Основна ідея AAS полягає у розмежуванні активу (asset) та його оболонки (shell).

Актив є ключовим поняттям AAS. Це реальний або логічний об’єкт, як-от, фізичний пристрій, програмний компонент, документ тощо, що має цінність для організації. Актив може бути фізичним об’єктом (мотор, давач тощо), нематеріальним об’єктом (програмне забезпечення, план виробництва) чи процесом. Тоді оболонка адміністрування активу – це програмний шар, який «огортає» актив; зовнішні сервіси отримують доступ до активу лише через його оболонку. В AAS також визначене поняття *підмоделі*. Підмодель описує конкретний аспект активу.

На рис. 4.1 наведено діаграму класів, яка відображає структуру AAS.

Метамодель AAS має ієрархію, яка складається з трьох рівнів [57–59, 77]:

1) заголовок (Header) – унікальний ідентифікатор активу (Global Asset ID) та інформація про саму оболонку;

2) тіло (Body) – набір підмоделей (Submodels), які використовуються для впорядкування даних за функціональним призначенням;
3) елементи підмоделі (Submodel Elements) – конкретні дані (властивості, файли, колекції, події).

AAS не є симуляційною моделлю, натомість вона забезпечує інформаційну та сервісну репрезентацію активу. Саме тому AAS трактується в стандартах ISO / IEC як «оболонка адміністрування» (administration shell), а не як «симуляційний двійник» (simulation twin).

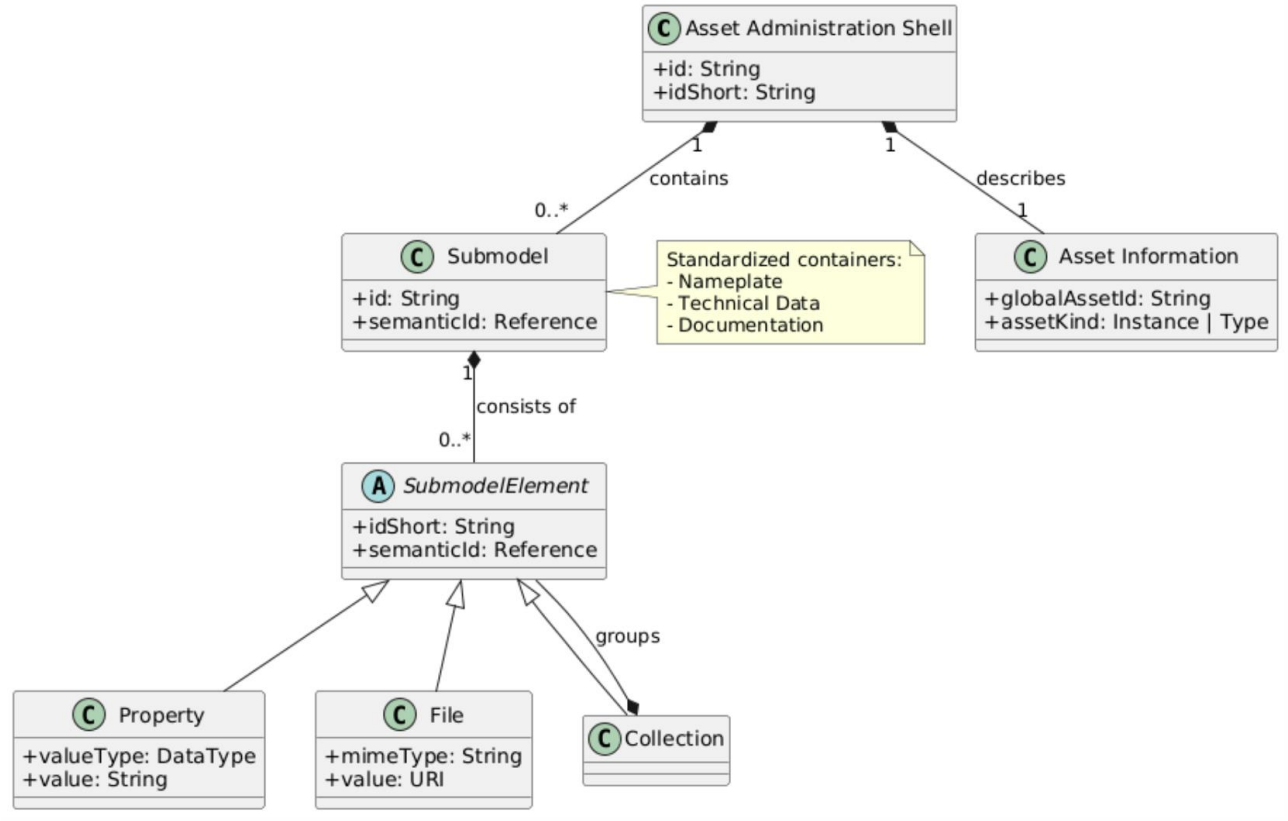

Рис. 4.1. Структура AAS

В AAS передбачено використання стандартизованих підмоделей. Типові стандартні підмоделі:

- цифрова табличка (Digital Nameplate), що містить серійний номер, рік випуску, виробника, сертифікати (CE, IP65) та замінює металеву шильду;
- технічні дані (Technical Data), до яких відносяться різноманітні характеристики (потужність, напруга, вага);

- документація (Documentation), зокрема, PDF-інструкції, 3D-моделі (STEP файли), електричні схеми тощо;
- операційні дані (Operational Data), тобто поточні показники (телеметрія) в реальному часі.

Специфічним прикладом підмоделі є Carbon Footprint – дані про викиди CO2 при виробництві певного активу.

Підмодель є основним механізмом моделювання аспектів: кожен Submodel описує одну «аспектну» модель інформації (технічні дані, цифрове ім'я, серійний номер, експлуатаційні параметри, сертифікати, технічна документація, телеметрія тощо). Підмоделі можуть бути шаблонами або екземплярами. Шаблони підмоделей розробляє асоціація промислових цифрових двійників – IDTA (Industrial Digital Twin Association) [58]. Якщо два виробники насосів використовують стандартну підмодель «Технічні дані», їхні цифрові двійники стають автоматично сумісними.

Для пов'язування елементів AAS із загальними словниками/довідниками (ECLASS, IEC CDD тощо) використовується механізм ConceptDescription, який дає змогу забезпечити семантичну інтероперабельність. Ідентифікація в AAS базується на унікальних ідентифікаторах (IRI/URI), що надає можливість посилатися на AAS, підмодель або елемент підмоделі у глобальному контексті.

AAS може існувати у трьох формах:

1) пасивний файл, який використовується для обміну даними між постачальником і замовником;
2) AAS API на сервері, який надає хостинг цифрового двійника;
3) AAS Registry, тобто реєстр, що працює за принципом «телефонної книги» та надає відповідну IP-адресу сервера цифрового двійника за серійним номером фізичного двійника.

Для організації обміну даними передбачено два основні варіанти: API (REST/HTTP, Web/OPC UA mappings) та файловий обмін. Файловий обмін звичайно здійснюється у вигляді пакетів з метаданими й файлами у форматі AASX [78]. Формат .AASX – це архів, який згідно з Open Packaging Conventions

містить JSON/XML описи та всі супровідні файли (PDF, PNG). Стандарт AAS підтримує також формати JSON, XML, RDF.

У Лістингу 4.6 наведено фрагмент коду, який описує цифрову табличку (Nameplate) для електромотора.

Лістинг 4.6. Спрощена JSON-структура AAS

```
{
  "assetAdministrationShells": [
    {
      "id": "https://university.edu/ids/aas/motor_001",
      "idShort": "Motor_001_Shell",
      "assetInformation": {
        "assetKind": "Instance",
        "globalAssetId":
"https://university.edu/ids/asset/motor_001_physical"
      },
      "submodels": [
        {
          "type": "ModelReference",
          "keys": [
            {
              "type": "Submodel",
              "value":
"https://university.edu/ids/sm/nameplate_001"
            }
          ]
        }
      ]
    }
  ],
  "submodels": [
    {
      "id": "https://university.edu/ids/sm/nameplate_001",
      "idShort": "DigitalNameplate",
      "semanticId": {
        "type": "ExternalReference",
        "keys": [
          {
            "type": "GlobalReference",
            "value": "https://admin-
shell.io/zvei/nameplate/1/0/Nameplate"
          }
        ]
      },
      "submodelElements": [
        {
          "idShort": "ManufacturerName",
          "modelType": "Property",
          "valueType": "xs:string",
```

```
          "value": "Kyiv Polytechnic Motors"
        },
        {
          "idShort": "YearOfConstruction",
          "modelType": "Property",
          "valueType": "xs:gYear",
          "value": "2024"
        },
        {
          "idShort": "MaxRotationSpeed",
          "modelType": "Property",
          "valueType": "xs:integer",
          "value": "5000",
          "semanticId": {
             "keys": [{"type": "ConceptDescription", "value":
"0173-1#02-BAA120#008"}]
          }
        }
      ]
    }
  ]
}
```

У цьому прикладі semanticId – це посилання на словник ECLASS або IEC CDD. Відповідно, код 0173-1#02-BAA120#008 є не випадковим набором цифр, а унікальним ідентифікатором параметру «швидкість обертання» у міжнародному стандарті. Саме це забезпечує семантичну інтероперабельність. Тому цей код буде інтерпретований однаково у будь-якій країні, де підтримується стандарт AAS, навіть якщо назви полів будуть різними.

AAS розрізняє два варіанти цифрового двійника: AAS Type та AAS Instance. AAS Type (тип) визначає цифровий двійник проєкту майбутнього фізичного об’єкта. Наприклад, конструкторське бюро розробило нову модель двигуна. Він ще не виготовлений, але у нього вже є AAS Type з кресленнями та номінальними характеристиками. AAS Instance (екземпляр) визначає цифровий двійник реального фізичного об’єкта. Так, коли двигун зійшов з конвеєра, для нього створюється AAS Instance. Він наслідує дані від Type, але містить унікальні дані: серійний номер, дату виготовлення та історію експлуатації.

Таким чином, AAS формалізує структуру даних про актив, забезпечує семантичну інтероперабельність та зв’язок між системами (MES, ERP, PLM, IoT тощо), визначає уніфікований програмний інтерфейс для зберігання інформації

та надання сервісів протягом життєвого циклу об’єкта. AAS є центральним елементом забезпечення інтероперабельності та інформаційним ядром цифрового двійника у промислових системах.

4.3. Стандарт OPC UA

OPC UA (Open Platform Communications Unified Architecture) є платформо-незалежною, сервіс-орієнтованою архітектурою для промислової взаємодії («машина – машина», «машина – підприємство»), що надає стандартизований спосіб подання даних і поведінки пристроїв/систем у вигляді простору адрес та сервісів (читання/запис/події/історичні дані тощо) [47, 48, 79, 80]. Стандарт OPC UA забезпечує передавання не лише самих даних, а й їхньої семантики. Його основна мета – забезпечити безпечний, надійний та семантично багатий обмін інформацією між пристроями й програмними системами різних виробників та рівнів (від PLC до MES/ERP).

Архітектурно OPC UA підтримує розподіл відповідальностей: від виявлення серверів і встановлення безпечних з’єднань до доступу до інформації чи подій. На відміну від класичних OPC-технологій, що були прив’язані до COM/DCOM, OPC UA реалізує незалежні від платформи механізми комунікації і безпеки, що дозволяє використовувати його на Linux, macOS, вбудованих системах, IoT-шлюзах і в хмарних середовищах.

Ключовою концепцією OPC UA є уніфікована модель інформаційного простору (Address Space Model), в якій дані представляються у вигляді типізованих вузлів, що можуть містити змінні, об’єкти, методи та відносини між собою. Це дає змогу не лише передавати значення, а й повністю описувати поведінку пристрою чи процесу в термінах об’єктно-орієнтованої моделі даних.

Безпека в OPC UA інтегрована на всіх рівнях архітектури. Стандарт пропонує механізми автентифікації користувачів і застосунків, шифрування та перевірки цілісності повідомлень, а також управління сертифікатами та політиками безпеки. Система безпеки побудована так, що захищене з’єднання

підтримується протягом життєвого циклу сесії, що знижує ризики несанкціонованого доступу.

Стандарт визначає набір абстрактних сервісів (читання, запис, підписки на події, історичні дані тощо), а конкретні мережеві мапінги реалізуються через різні транспортні протоколи (UA TCP, HTTPS, WebSockets, Pub/Sub з MQTT/AMQP тощо). Завдяки цьому OPC UA може ефективно застосовуватися як у традиційних SCADA/MES-середовищах, так і в архітектурах IIoT.

В OPC UA використовуються галузеві специфікації Companion Specifications, які уможливлюють створення галузевих інформаційних моделей для конкретних типів обладнання чи застосувань, забезпечуючи семантичну сумісність і повторне використання даних у складних системах.

Таким чином, OPC UA представляє собою не просто протокол передачі даних, а повноцінну уніфіковану модель комунікацій, даних та поведінки, що робить його важливим технологічним базисом для побудови цифрових двійників, інтеграції OT/IT-систем та створення відкритої, безпечної і масштабованої промислової архітектури.

Типовими прикладами використання OPC UA є підключення PLC, роботів, машин до SCADA, MES, IIoT-шлюзів та створення цифрових двійників як інформаційної моделі пристрою.

Контрольні запитання

1. Чому семантична інтероперабельність є критичною вимогою для систем цифрових двійників?
2. Яке призначення мови DTDL і на яких стандартах вона ґрунтується?
3. У чому полягає роль інтерфейсу як атомарної одиниці моделювання в DTDL?
4. Поясніть семантичну різницю між Telemetry, Property та Command у DTDL.
5. Яким чином механізми наслідування та компонентів у DTDL сприяють повторному використанню моделей?

6. У чому полягає концептуальна відмінність між цифровим двійником і AAS як оболонкою адміністрування активу?
7. Яку роль відіграють підмоделі в структурі AAS і як вони пов’язані з аспектним моделюванням?
8. Як механізм ConceptDescription в AAS забезпечує семантичну узгодженість даних?
9. Які основні архітектурні принципи та можливості стандарту OPC UA роблять його придатним для цифрових двійників?
10. Як DTDL, AAS та OPC UA можуть бути використані спільно в межах однієї платформи цифрового двійника?

Завдання для самостійної підготовки

1. Розробіть DTDL-модель цифрового двійника простого фізичного об’єкта з використанням телеметрії, властивостей, команд і відносин.
2. Побудуйте ієрархію інтерфейсів DTDL із застосуванням механізму наслідування та обґрунтуйте її доцільність.
3. Спроєктуйте структуру AAS для обраного активу, визначивши заголовок, підмоделі та елементи підмоделей.
4. Розробіть концептуальну схему обміну даними між цифровим двійником, реалізованим на основі AAS, та фізичним активом через OPC UA.
5. Сформулюйте науково-дослідну задачу, пов’язану з уніфікацією або узгодженням стандартів семантичного моделювання цифрових двійників.

5. ПРИКЛАДНІ АСПЕКТИ ПРОЄКТУВАННЯ ЦИФРОВИХ ДВІЙНИКІВ

У розділі розглянуто прикладні аспекти проєктування програмних систем на основі технології цифрових двійників у галузях охорони здоров'я та освіти. Проаналізовано архітектурні підходи до створення медичних діагностичних систем, що працюють із темпоральними мультимодальними даними, отриманими з гетерогенних медичних джерел. Запропоновано механізми уніфікації, синхронізації та агрегації медичних даних із використанням файлів-обгорток і мультиобразів для формування цифрового двійника пацієнта. Окрему увагу приділено проєктуванню програмних систем онлайн-лабораторій та імерсійних навчальних середовищ, у яких цифровий двійник виступає інтеграційною основою для поєднання поведінкових і мультимедійних даних. Розділ демонструє, як узагальнені архітектурні принципи цифрових двійників адаптуються до конкретних прикладних доменів з урахуванням їхніх функціональних, технологічних та ергатичних особливостей.

5.1. Програмне забезпечення медичних діагностичних систем на основі технології цифрових двійників

Впровадження інформаційної технології цифрових двійників у галузі охорони здоров'я створює умови для розроблення систем підтримки прийняття медичних рішень на основі синхронізованих та консолідованих темпоральних мультимодальних медичних даних про стан здоров'я пацієнта, які надаватимуть лікарю інформацію про стан здоров'я пацієнта на якісно новому рівні [9]. Система підтримки прийняття медичних рішень на основі технології цифрових двійників дозволить лікарю:

- аналізувати інформацію, подану з позицій часу;
- краще розуміти залежності між подіями лікування та результатами лікування (включаючи побічні ефекти для інших органів та систем);
- шукати подібні випадки, що трапились у пацієнта в минулому;
- моделювати можливий вплив нових схем лікування тощо.

Технологія цифрових двійників передбачає накопичення даних, які реєструються в результаті тривалого моніторингу реального об'єкта. Це відповідає випадку довготривалого моніторингу стану здоров'я пацієнта, особливо для пацієнтів із хронічними захворюваннями. Таким чином, ідея застосування цифрових двійників у галузі охорони здоров'я полягає у створенні та динамічному оновленні індивідуального набору даних пацієнта. Цей

індивідуальний набір даних містить синхронізовані та агреговані мультимодальні дані, які отримуються в результаті різних медичних досліджень та подаються згідно з персоналізованою семантичною моделлю пацієнта, органів та систем органів його тіла. Ця персоналізована семантична модель має відображати хронічні захворювання та індивідуальні стани організму пацієнта. Тіло людини в цілому, а також окремі його органи та системи органів, можуть характеризуватися багатьма параметрами, виміряними за допомогою широкого спектру медичного обладнання. Дані, які отримуються в результаті різних медичних досліджень та тестів, є мультимодальними, оскільки вони описують певний специфічний аспект функціонування людського організму. Для створення цифрового двійника фізичного двійника ці мультимодальні дані повинні бути синхронізовані та агреговані.

Мультимодальні дані для створення цифрового двійника пацієнта можуть бути отримані за допомогою широкого спектру пристроїв, інструментів та інструментів, які генерують мультимодальні дані у спеціалізованих, іноді унікальних форматах. Для адаптації узагальненої архітектури програмної системи оброблення темпоральних мультимодальних даних цифрових двійників до специфіки медичної галузі, потрібно проаналізувати види медичного діагностичного обладнання та типи даних, що можуть бути отримані з його допомогою.

Усі медичні пристрої та інструменти можна розділити на декілька груп, включаючи діагностичне обладнання, обладнання медичної лабораторії, медичні монітори, терапевтичне обладнання, обладнання для життєзабезпечення тощо. Діагностичні прилади та інструменти можна класифікувати як вимірювальний медичний інструмент (термометри, сфігмоманометри, глюкометри тощо), лабораторне обладнання (біохімічні аналізатори, гематологічні аналізатори тощо), медичне обладнання для візуалізації (апарати для ультразвукового дослідження, МРТ-апарати, ПЕТ- та КТ-сканери, апарати для рентгенографії тощо), функціональне діагностичне обладнання (електрокардіографи, електроенцефалографи, електроміографи, реографічні прилади, спірометри

тощо). Ці пристрої та інструменти можуть бути цифровими або нецифровими. Вимірювання, отримані за допомогою нецифрових інструментів (наприклад, ртутного термометра, ручного сфігмоманометра), звичайно записуються вручну або як паперові, або як електронні записи. Вимірювання, отримані за допомогою цифрового обладнання, подаються у вигляді файлів певних форматів залежно від типу використовуваного медичного обладнання. У деяких діагностичних процедурах, таких як ендоскопія, зокрема, колоноскопія, де результатом дослідження є потік відеоданих, використовуються формати файлів загального призначення. Найпоширеніші формати файлів медичних даних [9, 81, 82] представлені в табл. 5.1. Аналіз цієї таблиці дозволяє зробити висновок про те, що більшість форматів медичних даних передбачає поле метаданих для зберігання часових міток, проте, структура файлів істотно відрізняється. Тому доцільно включити до архітектури програмного забезпечення медичної діагностичні системи на основі технології цифрових двійників модуль конвертації, який буде перетворювати файл кожного типу, що передбачається застосовувати у цій системі, у формат файлу-обгортки. Компонентна діаграма цього модуля наведена на рис. 5.1.

Компонент *DataReceiver* призначений для отримання медичних даних з діагностичного обладнання. Компонент *TypeAnalyzer* призначений для аналізу типу вхідного файлу медичних даних. Компонент *TimeExtraction* призначений для отримання з файлу часових даних. Компонент *InputTimeLabel* призначений для введення часових даних людиною-оператором, що є альтернативним або допоміжним способом отримання часових даних програмною системою. Компонент *Wrapper* виконує формування файлу-обгортки, який є готовим до подальшого використання у модулі синхронізації та агрегації темпоральних даних.

Компонентна діаграма модуля синхронізації та агрегації медичних темпоральних мультимодальних даних показана на рис. 5.2.

Таблиця 5.1 – Формати файлів медичних даних

Тип файлу даних	Розширення	Основний тип даних, що зберігається	Тип дослідження	Наявність часових міток
FEF	.FEF	Біологічний сигнал	Електро-кардіограма	Є поле для послідовності часових значень
UFF	.BDF	Багатоканальний біологічний сигнал	Електро-енцефало-грама	Є чотири поля для значень часу: – “Start date of recording”, – “Start time of recording”, – “Number of data records”, – “Duration of a data record, in seconds”
LabPas HL7	.HL7	Результат аналізу крові	Аналіз крові	Є поле “Date/Time of the Observation”
UFF	.UFF	Дані УЗД	Ультра-звукове дослідження	Є поле “local_time”
NIFTI	.NII	Дані МРТ	Магнітно-резонансна томографія	Є зарезервоване поле для значень часу
DICOM	.DC3, .DCM, .DIC	Медичні зображення та метадані	Комп’ютерна томографія, рентгено-графія тощо	Є поле “Study date” та “Study time”
CSV	.CSV	Цілі та дійсні числа (ASCII)	Спірометрія, термометрія, сфігмомано-метрія тощо	Немає

Компонент *TemporalDataExtracter* призначений для перетворення даних з формату файлу-обгортки у внутрішнє подання темпоральних даних. Компонент *TemplateFormer* реалізує шаблони синхронізації, які використовуються компонентом *Synchronizer*. Компонент *Synchronizer* призначений для синхронізації темпоральних мультимодальних даних, отриманих з послідовності вхідних файлів, що подані у форматі файлу-обгортки [9].

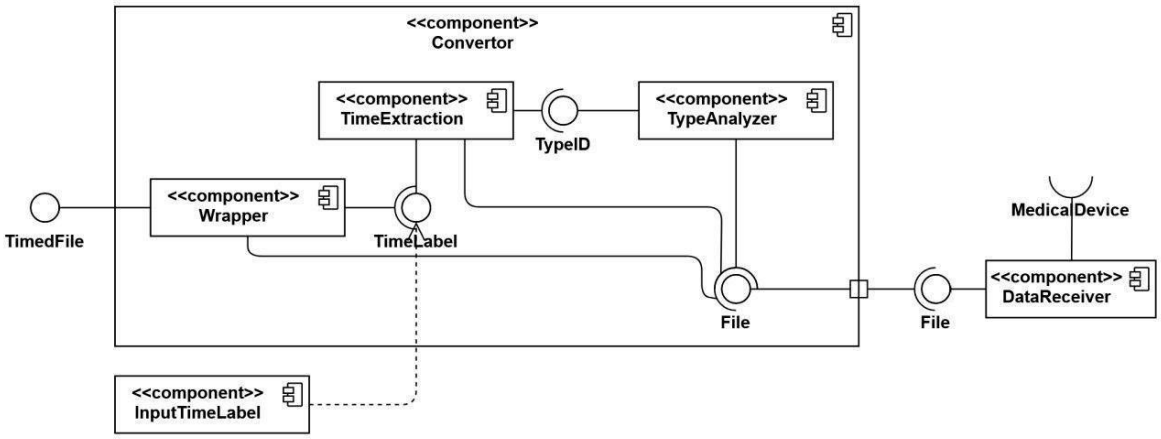

Рис. 5.1. Компонентна діаграма модуля конвертації медичних даних [9]

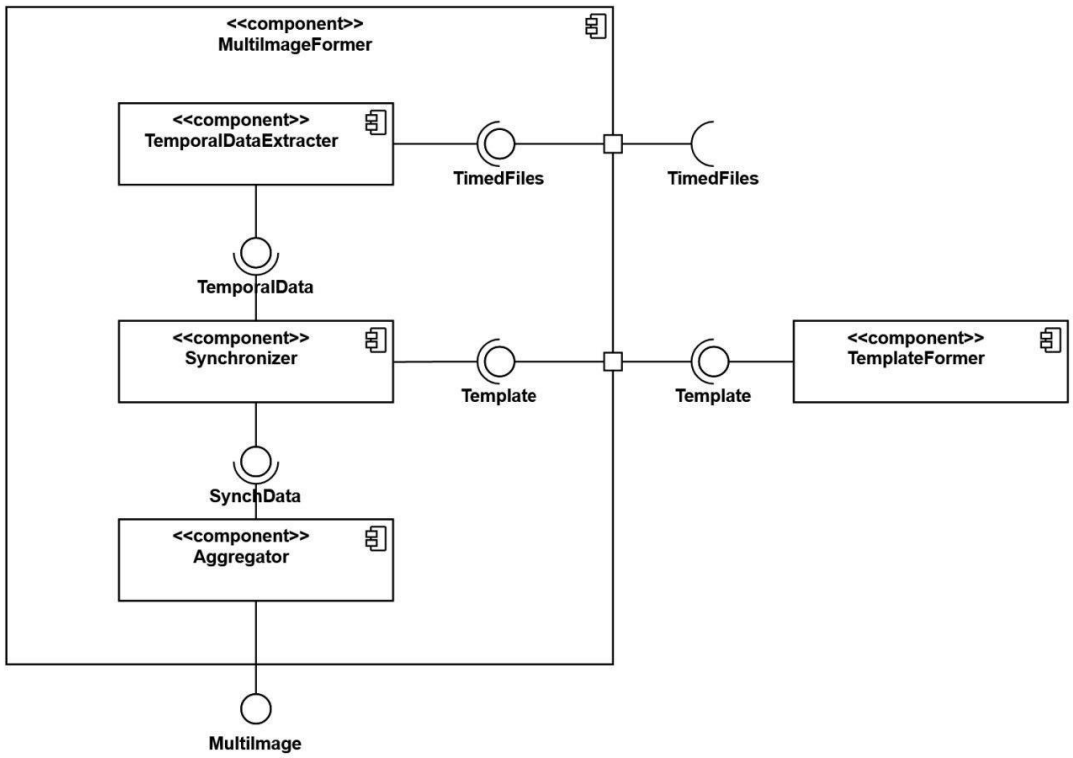

Рис 5.2. Компонентна діаграма модуля синхронізації та агрегації медичних темпоральних мультимодальних даних [9]

Структура цього файлу (рис. 5.3) включає три розділи: розділ метаданих, розділ часових значень та головний розділ, де знаходиться обгорнутий файл.

Розділ з метаданими містить маркер "TIME", ідентифікатор типу файлу (наприклад, "nii" для файлового формату NIFTI [82], "dcm" для формату файлу DICOM тощо) та розмір кортежу значень часу. Розділ часових значень містить значення часу, впорядковані за зростанням.

Основний розділ містить обгорнутий файл; цей файл є готовим до використання даних, наприклад, для їх відтворення або оброблення програмним шляхом. Така процедура модифікації оригінального файлу дозволяє уніфікувати процедуру синхронізації послідовностей даних, отриманих з джерел різних типів, та при цьому зберегти дані у їх оригінальному форматі.

Компонент *Aggregator* призначений для агрегації синхронізованих темпоральних мультимодальних даних та формування мультиобразу, який готовий до подальшого використання для формування цифрового двійника.

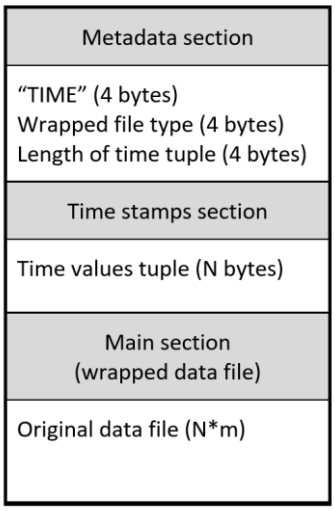

Рис. 5.3. Структура файлу-обгортки [9]

Процес створення цифрового двійника пацієнта представлений на рис. 5.4. Перший етап цього процесу передбачає отримання даних про біомедичний об’єкт. Джерелом цих даних можуть бути медичні прилади, медичний інструментарій, лабораторне обладнання, медична документація. Дані можуть мати цифровий та нецифровий характер. Можливість отримання деяких медичних даних з нецифрових джерел, як-то, ртутний термометр, паперова медична картка пацієнта тощо, перетворює цю медичну діагностичну систему на ергатичну.

Другим етапом процесу створення цифрового двійника пацієнта є перетворення кожного файлу в універсальний формат, який містить часові мітки у визначеному форматі файлу-обгортки. В результаті отримуємо файл з часовими мітками (*timed*-файл) для кожного оригінального файлу певного типу.

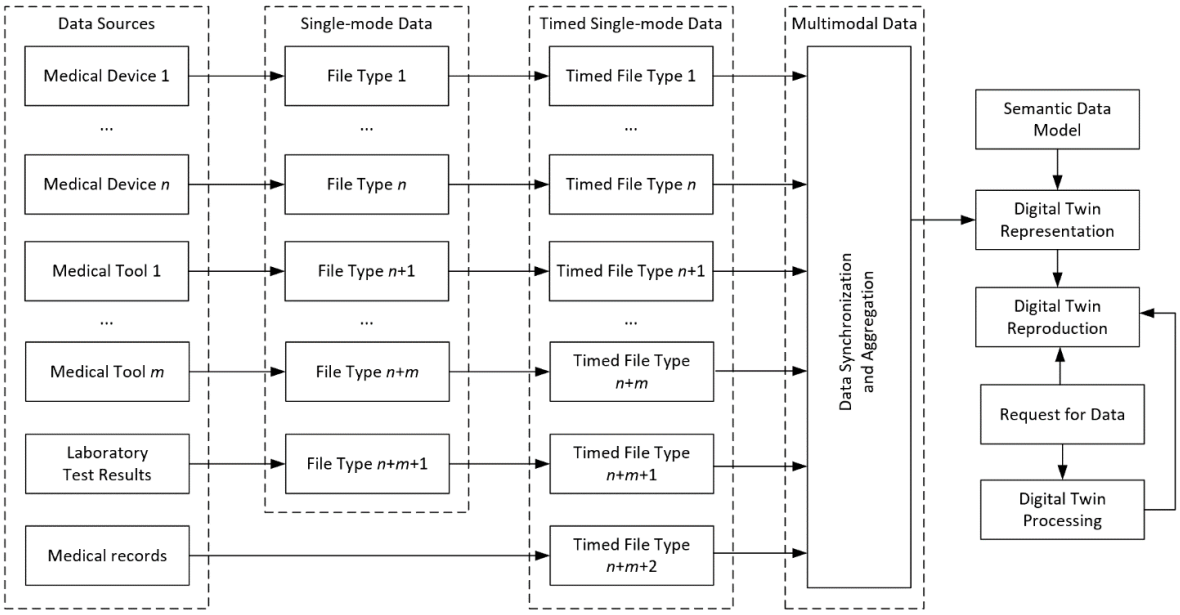

Рис. 5.4. Процес створення цифрового двійника пацієнта [9]

На третьому етапі відбувається синхронізація та агрегація даних, що подані у форматі файлу-обгортки. Процедура синхронізації використовує часові мітки, наявні у метаданих *timed*-файлів. Отриманий мультиобраз надає комплексне подання даних цифрового двійника пацієнта.

Для створення цифрового двійника потрібно знати семантичну модель відповідного фізичного двійника, тобто пацієнта. Ця семантична модель повинна базуватися на медичних стандартах. Персоніфікована семантична модель пацієнта – це вихідні дані для цієї інформаційної технології, а також дані, отримані з діагностичного медичного обладнання. Синхронізовані та агреговані дані зберігаються відповідно до семантичної моделі. Спосіб цільової оброблення даних та варіанти відтворення цифрового двійника залежать від конкретних завдань моніторингу та дослідження стану здоров’я пацієнта. Для покращення якості візуалізації цифрового двійника передбачається використання додаткових етапів оброблення візуальної моделі.

Взаємодія модулів програмної системи, що реалізує процес створення цифрового двійника пацієнта, показана у вигляді діаграми послідовності, що наведена на рис. 5.5.

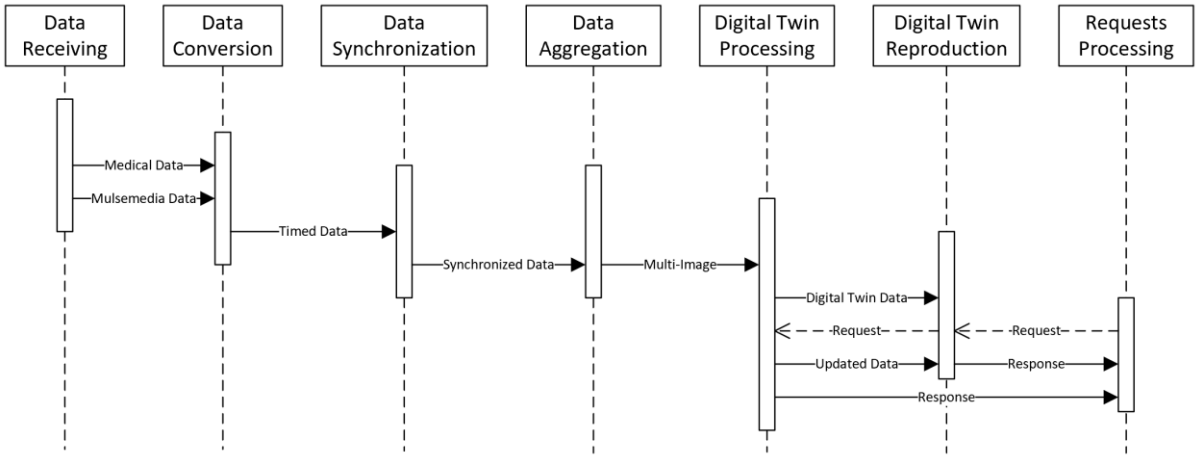

Рис. 5.5. Взаємодія модулів медичної програмної системи [9]

Архітектура програмного забезпечення медичної діагностичної системи на основі технології цифрових двійників наведена на рис. 5.6.

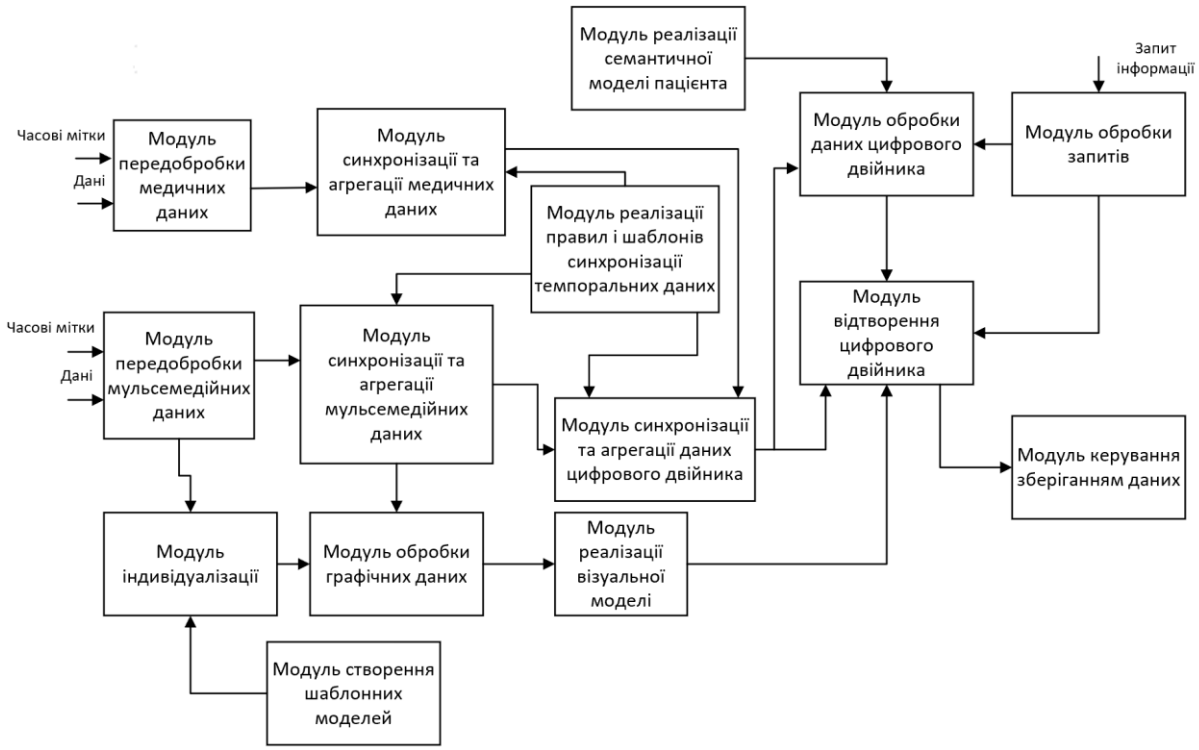

Рис. 5.6. Архітектура програмного забезпечення медичної діагностичної системи на основі технології цифрових двійників [9]

Для спрощення візуального моделювання передбачається використання шаблонних моделей тіла людини, його частин та окремих органів. На основі вимірювань та медичних досліджень, проведених для конкретного пацієнта, шаблонна модель індивідуалізується шляхом зміни розмірів, пропорцій та інших

показників, які впливають на візуальну модель. Додатково до архітектури медичної програмної системи можуть бути включені компоненти, які забезпечують застосування процедур покращення якості медичних графічних даних та інших видів оброблення даних візуальної моделі.

Запропонована архітектура медичної програмної системи на основі технології цифрових двійників може бути застосована для розроблення експертних медичних систем з розширеними функціональними можливостями, програмного забезпечення систем e-Health [83] та m-Health [84], програмного забезпечення Hospital-at-Home [85] тощо.

5.2. Програмні системи дистанційного навчання на основі технології цифрових двійників та технології мульсемедіа

Розглянемо особливості розроблення програмного забезпечення з використанням технології цифрових двійників та технології мульсемедіа [9, 86] для двох різновидів систем, що використовуються у галузі освіти: онлайн-лабораторії та імерсійного середовища [9, 87]. В обох випадках передбачається, що особи, що навчаються, досліджують (вивчають) певний реальний об’єкт (процес, явище тощо). Дослідження цього об’єкта відбувається шляхом реєстрації його характеристик за допомогою набору цифрових давачів. Оскільки характеристики відображають різні сторони природи або поведінки об’єкта, отримуємо послідовності темпоральних мультимодальних даних, на основі яких формується та відтворюється цифровий двійник досліджуваного об’єкта.

Онлайн-лабораторія [9, 88] – це програмна система, яка дозволяє учням використовувати віддалене обладнання для проведення експериментів для навчальних цілей. Програмне забезпечення онлайн-лабораторії включає програмні засоби для оброблення, аналізу, зберігання, передачі даних та візуалізації результатів.

Архітектуру програмного забезпечення онлайн-лабораторії, що ґрунтується на використанні технології цифрових двійників та технології мульсемедіа, наведено на рис. 5.7.

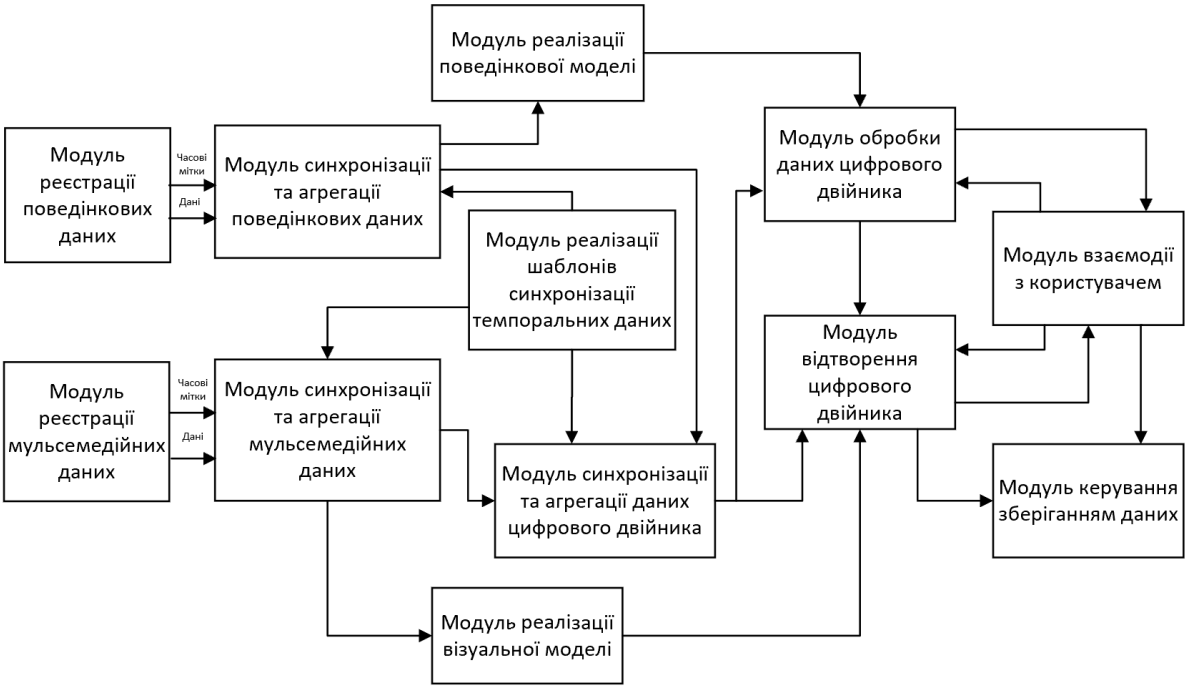

Рис. 5.7. Архітектура програмного забезпечення онлайн-лабораторії [9]

Послідовності темпоральних даних двох категорій (поведінкові дані та мульсемедійні дані) надходять до програмної системи у режимі реального часу з N цифрових давачів, з якими безпосередньо взаємодіють модуль реєстрації поведінкових даних та модуль реєстрації мульсемедійних даних. Причому, $N = N_1 + N_2$, де N_1 – кількість давачів, що реєструють поведінкові дані та N_2 – кількість давачів, що реєструють мульсемедійні дані. В результаті вимірювання характеристик досліджуваного об'єкта утворюються N пар послідовностей ($\bar{t}^k$, $\bar{s}^k$), таких, що $\bar{t}^k = \langle t_i^k \rangle_{i=1}^{n_k},\ t_i^k \in T,\ \bar{s}^k = \langle s_i^k \rangle_{i=1}^{n_k},\ s_i^k \in M_k, t_{n_k}^k \leq L_{exp}$, де $\bar{t}^k$ – це послідовність значень, що визначають моменти часу t_i^k, коли вимірюються значення s_i^k, які відображають характеристику об'єкта, T – множина часових значень, M_k – множина даних k-ї модальності, L_{exp} – тривалість експерименту. Ці темпоральні мультимодальні дані, залежно від їх категорії, надходять до модулю синхронізації та агрегації поведінкових даних або до модулю синхронізації та агрегації мульсемедійних даних.

Синхронізація має особливе значення, коли віддалений зв'язок із давачами є нестабільним через високе навантаження мережі або інші зовнішні умови.

Синхронізація відбувається згідно з шаблонами синхронізації, що визначені відповідно до режимів роботи давачів даних.

Синхронізовані та агреговані дані обох категорій надходять до модуля синхронізації та агрегації даних цифрового двійника, а також дані кожної категорії надходять до модуля реалізації моделі відповідного типу (поведінкової або візуальної). Результати моделювання поведінки об’єкта надходять до модуля оброблення даних цифрового двійника. Результати моделювання зовнішнього вигляду та інших зовнішніх проявів об’єкта надходять до модуля відтворення цифрового двійника. Спосіб відтворення цифрового двійника залежить від технічних можливостей на стороні користувача онлайн-лабораторії. В разі відсутності у користувача обладнання для відтворення мульсемедійних ефектів візуалізація відбувається лише у графічний спосіб, тобто відтворюються візуальні або аудіовізуальні дані, а дані інших мульсемедійних типів відтворюються у вигляді графіків, таблиць, текстових описів тощо. Безпосередня комунікація користувача з системою відбувається через модуль взаємодії з користувачем, який може бути реалізований, наприклад, як веб-застосунок.

Модуль оброблення даних цифрового двійника реалізує алгоритми оброблення даних відповідно до мети онлайн-лабораторії, характеру експериментів, наявного обладнання тощо. Вхідними даними цього модуля є консолідовані дані цифрового двійника та дані, що отримуються в результаті моделювання поведінки досліджуваного об’єкта, а також запити, що отримуються від користувача системи у процесі його роботи (експериментів) з онлайн-лабораторією. Для збереження результатів експериментів та інших даних користувача використовується локальна база даних; для збереження даних великого обсягу, що оброблюються у онлайн-лабораторії, передбачається використання сховища даних.

Діаграма розгортання основних компонентів онлайн-лабораторії наведена на рис. 5.8.

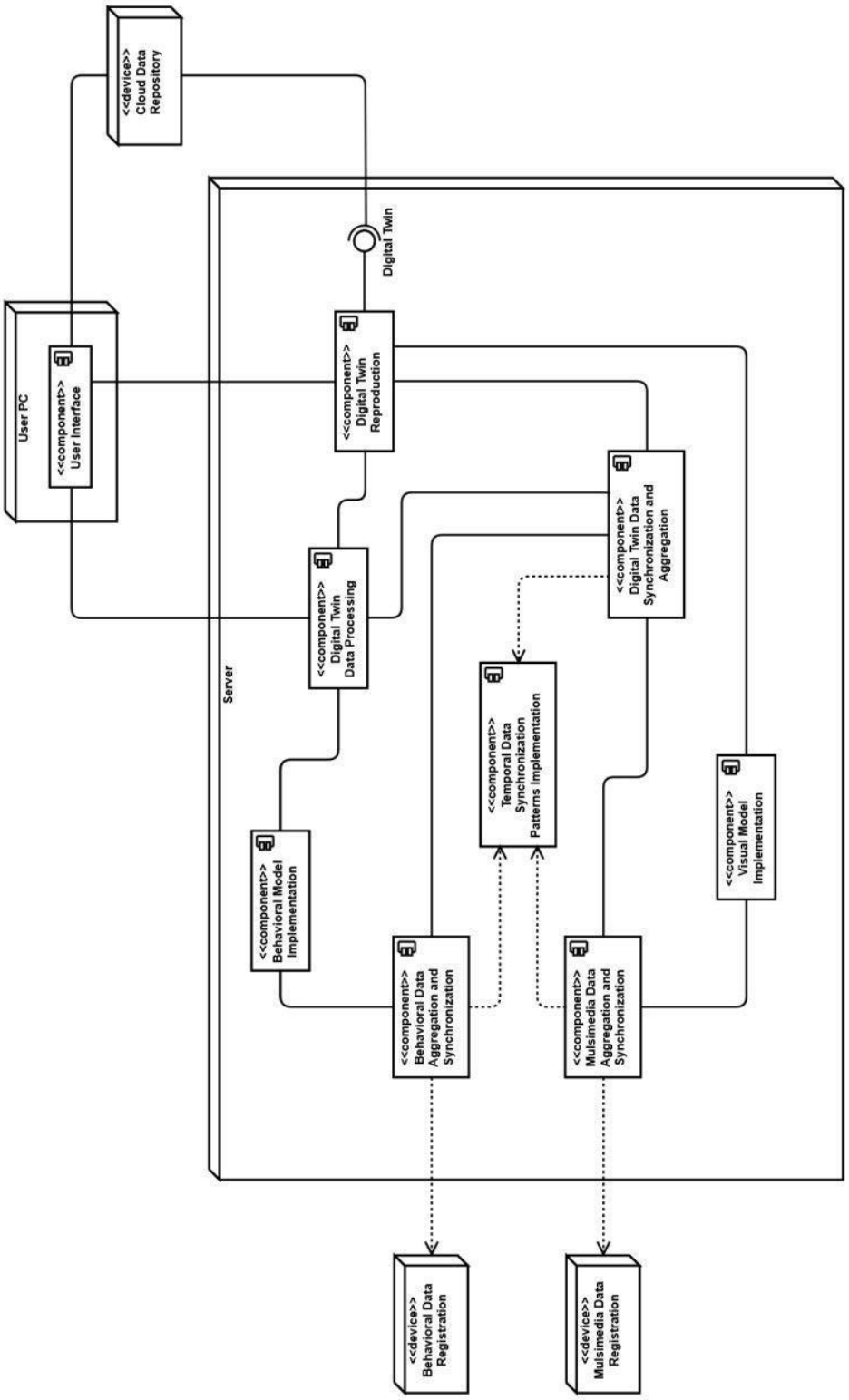

Рис. 5.8. Діаграма розгортання програмного забезпечення онлайн-лабораторії [9]

Як показано на діаграмі розгортання, компоненти програмного забезпечення онлайн-лабораторії встановлюються на вузли п’ятьох типів. Програмне забезпечення для реєстрації поведінкових даних та мульсемедійних

даних встановлюється на комп’ютерне обладнання системи давачів. Програмний застосунок, що призначений для взаємодії користувача з онлайн-лабораторією встановлюються на пристрій користувача (ноутбук, планшет, смартфон тощо). Програмне забезпечення, що реалізує решту функцій системи, встановлюються на сервер.

Як було зазначено вище, онлайн-лабораторія передбачає, але не вимагає відтворення мульсемедійних даних на стороні користувача. Цим онлайн-лабораторія відрізняється від імерсійного середовища, де відтворення мульсемедійних даних є обов’язковим. Розробимо архітектуру імерсійного середовища, що ґрунтується на технології цифрових двійників.

Імерсійне середовище [9, 87] – це апаратно-програмна система, яка дозволяє імітувати присутність користувача у штучно створеній сцені (віртуальному світі), де користувач може взаємодіяти у віртуальний спосіб з реальними та віртуальними об’єктами, процесами або явищами.

Апаратна частина імерсійного середовища складається з цифрових давачів, які відстежують дії користувача в імерсійному середовищі, та актуаторів, що роблять можливим тривимірну візуалізацію, просторове відтворення звуку, тактильну взаємодію користувача з віртуальними об’єктами тощо. Програмна частина імерсійного середовища включає програмні засоби для візуалізації віртуальних об’єктів та віртуальної взаємодії з реальними об’єктами. Архітектура програмного забезпечення імерсійного середовища наведена на рис. 5.9. Основними даними для програмної системи, що реалізує імерсійне середовище на основі технології цифрових двійників, є наступні дані:

- поведінкові дані фізичного двійника;
- мульсемедійні дані фізичного двійника;
- дані цифрового двійника;
- дані віртуальної модулі сцени.

Поведінкові та мульсемедійні дані фізичного двійника надходять для синхронізації та агрегації до відповідного модуля синхронізації та агрегації, де відбувається їх оброблення відповідно до шаблонів синхронізації, вид яких

залежить від типу давачів, що застосовуються для моніторингу фізичного двійника.

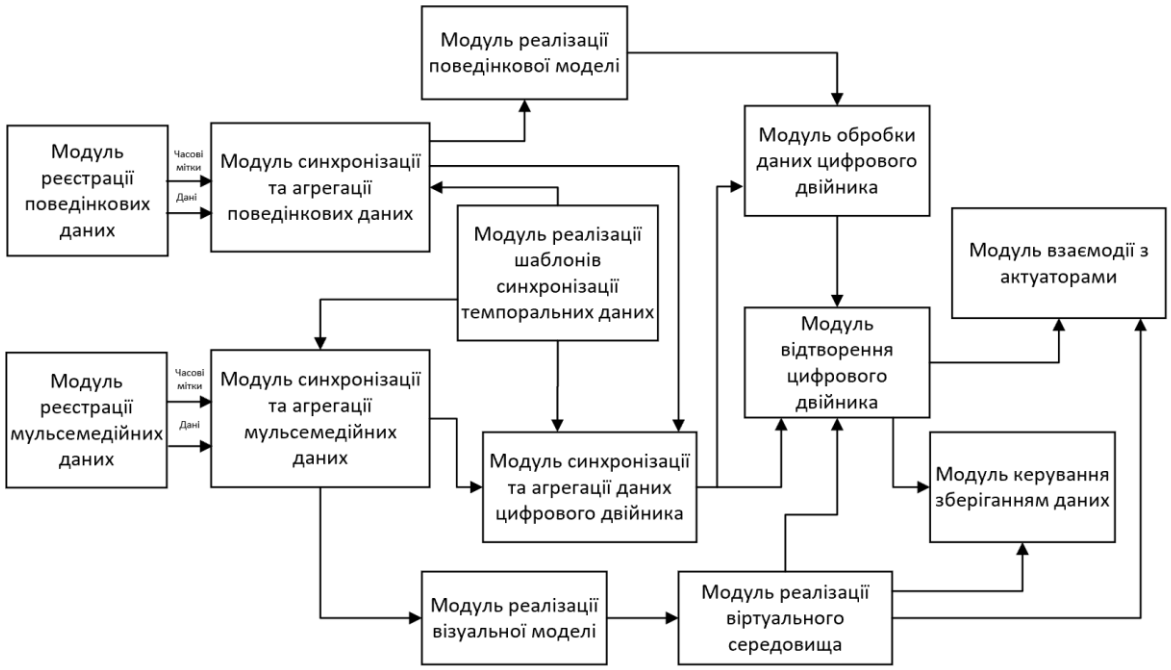

Рис. 5.9. Архітектура програмного забезпечення імерсійного середовища [9]

Консолідовані дані кожної категорії (поведінкові дані, мульсемедійні дані) надходять до модулю реалізації моделі відповідного виду, а також до модулю синхронізації та агрегації даних цифрового двійника. Сформована візуальна модель цифрового двійника надається для оброблення до модуля реалізації віртуального середовища, яке формує вигляд віртуальної сцени. Дані про сформовану віртуальну сцену надходять до модуля відтворення цифрового двійника та подаються на модуль взаємодії з актуаторами.

Результати моделювання поведінки фізичного двійника та консолідовані дані цифрового двійника передаються з модуля реалізації поведінкової моделі на модуль оброблення даних цифрового двійника, де відбувається цільова оброблення цих даних, яка залежить від призначення імерсійного середовища. Результати цього оброблення надходять до модуля відтворення цифрового двійника для уточнення та динамічної зміни цифрового двійника.

Оскільки дані цифрового двійника та віртуального середовища мають великий об’єм, передбачається їх зберігання у сховищах даних; керування процесом зберігання цих даних забезпечує відповідний модуль.

Якщо передбачається активна взаємодія користувача з елементами віртуального простору, то додатково до архітектури імерсійної системи мають бути включені модуль реєстрації дій користувача, який передбачає взаємодію з відповідними давачами даних, та модуль зворотного зв’язку, який ґенерує реакцію системи на дії користувача.

Діаграма розгортання основних компонентів імерсійного середовища наведена на рис. 5.10.

Як показано на діаграмі розгортання, компоненти програмного забезпечення імерсійного середовища встановлюються на вузли п’ятьох типів. Програмне забезпечення для реєстрації поведінкових даних встановлюється на комп’ютерне обладнання системи давачів, які реєструють технічні характеристики фізичного двійника. Програмне забезпечення для реєстрації мульсемедійних даних встановлюється на комп’ютерне обладнання системи давачів, які реєструють зовнішні прояви фізичного двійника. Програмне забезпечення, що реалізує решту функцій системи, встановлюються на сервер. Окремими вузлами є вузол сховища даних та вузол системи актуаторів.

Одним з варіантів імерсійного середовища є імерсійна телеконференц-система, яка призначена для реєстрації, передавання та відтворення не лише аудіовізуальної інформації, а й інформації про навколишнє середовище, де присутні користувачі.

Метою застосування імерсійної телеконференц-системи створення у її учасників ефекту присутності, що досягається за рахунок реєстрації, передачі та відтворення мультимодальної інформації про всю сцену, де присутні учасники телеконференції. Основною вимогою до імерсійної телеконференц-системи є забезпечення достатньо високого рівня QoE та QoS. Це може бути досягнуто за рахунок використання технології WebRTC.

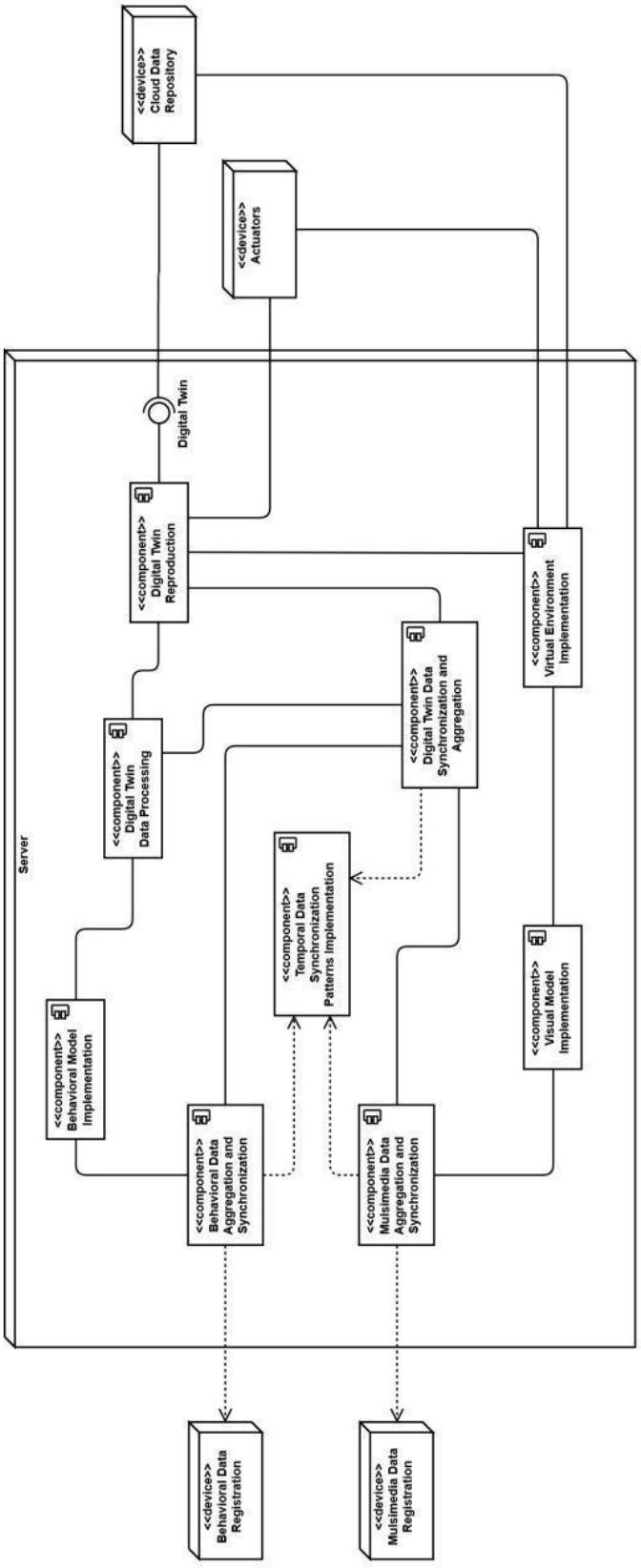

Рис. 5.10. Діаграма розгортання програмного забезпечення імерсійного середовища [9]

Таким чином, розроблена узагальнена архітектура програмної системи оброблення темпоральних мультимодальних даних цифрових двійників може

бути адаптована для розроблення різноманітного програмного забезпечення, що ґрунтується на технології цифрових двійників, зокрема для розроблення програмних систем оброблення темпоральних мультимодальних даних для галузі охорони здоров’я і освіти.

Контрольні запитання

1. У чому полягає специфіка застосування технології цифрових двійників у медичних діагностичних системах?
2. Чому для побудови цифрового двійника пацієнта необхідна синхронізація темпоральних мультимодальних медичних даних?
3. Які типи медичного діагностичного обладнання є основними джерелами даних для цифрового двійника пацієнта?
4. Яке призначення файлу-обгортки та яку роль він відіграє в уніфікації медичних даних різних форматів?
5. У чому полягає різниця між синхронізацією та агрегацією темпоральних мультимодальних даних?
6. Чому медична діагностична система на основі цифрового двійника може розглядатися як ергатична система?
7. Які функціональні модулі є ключовими в архітектурі програмного забезпечення цифрового двійника пацієнта?
8. У чому полягає відмінність між онлайн-лабораторією та імерсійним середовищем з точки зору використання цифрового двійника?
9. Яку роль відіграють поведінкові та мультимедійні дані у навчальних системах на основі цифрових двійників?
10. Які вимоги до якості обслуговування (QoS) та якості сприйняття (QoE) є критичними для імерсійних систем?

Завдання для самостійної підготовки

1. Класифікуйте мультимодальні медичні дані, що можуть бути використані для формування цифрового двійника пацієнта, та обґрунтуйте потребу в їх синхронізації.
2. Проаналізуйте роль семантичної моделі пацієнта у зберіганні та обробленні агрегованих медичних даних.
3. Спроєктуйте узагальнену архітектуру програмного забезпечення онлайн-лабораторії на основі технології цифрових двійників.
4. Порівняйте архітектурні рішення онлайн-лабораторії та імерсійного середовища з точки зору оброблення мультимедійних даних.
5. Сформулюйте науково-дослідну задачу, пов’язану з обробленням темпоральних мультимодальних даних цифрових двійників у медицині або освіті.

СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Grieves M., Vickers J. Origins of the Digital Twin Concept // Working paper. Florida Institute of Technology / NASA, 2016. DOI: 10.13140/RG.2.2.26367.61609.
2. Lee E. A. Cyber Physical Systems: Design Challenges // IEEE ISORC. 2008.
3. Cai Yi, Starly B., Cohen P., Lee Y.-S. Sensor data and information fusion to construct digital-twins virtual machine tools for cyber-physical manufacturing // Procedia Manufacturing. 2017. Vol. 10. P. 1031–1042.
4. Gubbi J., Buyya R., Marusic S., Palaniswami M. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions // arXiv:1207.0203. 2012. DOI: 10.48550/arXiv.1207.0203.
5. Čolaković A., Hadžialić M. Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues // Computer Networks. 2018. Vol. 144. P. 17–39. DOI: 10.1016/j.comnet.2018.07.017.
6. Glaessgen E. H., Stargel D. The digital twin paradigm for future NASA and US Air Force vehicles // AIAA 53rd Structures, Structural Dynamics, and Materials Conference. Honolulu, Hawaii, 2012.
7. Grieves M. Virtually Intelligent Product Systems: Digital and Physical Twins // Complex Systems Engineering: Theory and Practice. American Institute of Aeronautics and Astronautics, 2019. P. 175–200.
8. Kritzinger W., Karner M., Traar G., Henjes J., Sihn W. Digital Twin in manufacturing: a categorical literature review and classification // IFAC-PapersOnLine. 2018. Vol. 51, Issue 11. P. 1016–1022.
9. Сулема Є. С. Методи, моделі та засоби обробки мультимодальних даних цифрових двійників досліджуваних об’єктів : дис. … д-ра техн. наук : 01.05.03. Київ, 2020. 343 с.
10. Grieves M., Vickers J. Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems // Transdisciplinary perspectives on complex systems. Springer, 2017. P. 85–113.
11. NASA. Technology Roadmaps. Technology Area 12: Materials, structures, mechanical systems, and manufacturing. National Aeronautics and Space Administration, 2015. 138 p.
12. Madni A. et al. Leveraging Digital Twin Technology in Model-Based Systems Engineering // Systems. 2019. No 7. P. 1–13.
13. Lu Y., Liu C., Wang K. I-K., Huang H., Xu X. Digital Twin-driven smart manufacturing: connotation, reference model, applications and research issues // Robotics and Computer Integrated Manufacturing. 2020. Vol. 61. P. 1–14.
14. Sturrock D. T. Using Commercial Software to Create a Digital Twin // Simulation for Industry 4.0. Springer, 2019. 288 p.
15. Redelinghuys A. J. H., Basson A. H., Kruger K. A Six-Layer Digital Twin Architecture for a Manufacturing Cell // Studies in Computational Intelligence. 2018. Vol. 803. P. 412–423.
16. AWS IoT TwinMaker. https://aws.amazon.com/iot-twinmaker/

17. IBM. Watson IoT Platform. Product architecture. https://www.ibm.com/support/knowledgecenter/SSQP8H/iot/overview/architecture.html
18. NoSQL Databases. https://www.ibm.com/cloud/learn/nosql-databases
19. Data Lake – An Overview. https://www.yash.com/blog/data-lake-an-overview/
20. Data Lake vs Data Warehouse. https://luminousmen.com/post/data-lake-vs-data-warehouse
21. ANSYS Twin Builder. https://www.ansys.com/products/digital-twin/ansys-twin-builder
22. Seebo Digital Twin Software. https://www.seebo.com/digital-twin-software/
23. Simio. Digital Twin. https://www.simio.com/applications/industry-40/Digital-Twin.php
24. Smith J. S., Sturrock D. T., Kelton W. D. Simio and simulation: modeling, analysis, applications. Simio LLC, 2019.
25. Microsoft Azure Digital Twins. https://azure.microsoft.com/services/digital-twins/
26. Introduction to Azure Data Lake Storage Gen2. https://learn.microsoft.com/azure/storage/blobs/data-lake-storage-introduction
27. Use the Azure Digital Twins APIs and SDKs. https://learn.microsoft.com/azure/digital-twins/how-to-use-apis-sdks
28. What is Azure Synapse Analytics? https://learn.microsoft.com/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-overview-what-is
29. Understand digital twins and their twin graph. https://learn.microsoft.com/azure/digital-twins/concepts-twins-graph
30. Microsoft. A walkthrough of the Azure Digital Twins Explorer // Internet of Things Show. Microsoft Learn, 2021.
31. CosmoTech. Enterprise Digital Twin Software Platform. https://cosmotech.com/software-solutions/software-solution-platform
32. SAP. Digital Manufacturing Cloud. https://help.sap.com/viewer/c86ca4026fae4cb3ba66ed751866175b/latest/en-US/055988415fe64a7f9d43c615ab4eda5c.html
33. Autodesk. DNA for Digital Twin. https://www.autodesk.com/campaigns/digital-twin
34. ISO/IEC/IEEE 12207:2017. Systems and software engineering – Software life cycle processes.
35. IBM. What is the software development life cycle (SDLC)? https://www.ibm.com/think/topics/sdlc
36. ISO/IEC/IEEE 29148:2011. Systems and software engineering — Requirements engineering.
37. Guide to the Software Engineering Body of Knowledge (SWEBOK® Guide). Version 3.0. IEEE Computer Society, 2014. 335 p.
38. Bass L., Weber I., Zhu L. DevOps: A Software Architect's Perspective. Boston : Addison-Wesley, 2015. 368 p.
39. Kim G., Humble J., Debois P., Willis J. The DevOps Handbook. Portland : IT Revolution Press, 2016. 480 p.

40. Shahin M., Babar M. A., Zhu L. Continuous Integration, Delivery and Deployment: A Systematic Review // IEEE Access. 2017. Vol. 5. P. 3909–3943. DOI: 10.1109/ACCESS.2017.2685629.
41. Pressman R. S. Software Engineering: A Practitioner's Approach. McGraw-Hill, 2010.
42. Sommerville I. Software Engineering. 10th ed. Pearson, 2015.
43. Schwaber K., Sutherland J. The Scrum Guide. 2017.
44. Stark J. Product Lifecycle Management. Springer, 2015.
45. Tao F. et al. Digital Twin Driven Smart Manufacturing. Elsevier, 2019.
46. OASIS. MQTT Version 5.0. OASIS Standard, 2019.
47. IEC 62541-1:2015. OPC Unified Architecture — Part 1.
48. IEC 62541-2:2015. OPC Unified Architecture — Part 2.
49. Sharda R., Delen D., Turban E. Analytics, Data Science, and Artificial Intelligence. Wiley, 2018.
50. Keith D. Understanding Key-Value Databases. Dataversity, 2020. https://www.dataversity.net/understanding-key-value-databases
51. Object Storage. https://www.ibm.com/cloud/learn/object-storage
52. Wei W. Time Series Analysis. Pearson Addison Wesley, 2006. 614 p.
53. Петренко Т. Г. Побудова моделі розумного вагона за технологією цифрового двійника // Збірник наукових праць УкрДУЗТ. 2018. Вип. 177. С. 42–43.
54. Праховнік Н. А., Сізов Д. Ризики помилок під час діагностики методом цифрового двійника // Проблеми охорони праці, промислової та цивільної безпеки. 2019. С. 358–361.
55. Michelson B. M. Event-Driven Architecture Overview // Middleware Journal. 2006.
56. Digital Twins Definition Language (DTDL). https://github.com/Azure/opendigitaltwins-dtdl
57. Heinonen M. et al. Semantic Modeling of Asset Administration Shell for Interoperability // Computers in Industry. 2020.
58. IDTA. Asset Administration Shell Specifications. https://industrialdigitaltwin.org
59. Platform Industrie 4.0. Details of the Asset Administration Shell. Berlin, 2020.
60. Zhu Z., Liu C., Xu X. Visualisation of the Digital Twin data using Augmented Reality // Procedia CIRP. 2019. Vol. 81. P. 898–903.
61. ISO 23247-1:2021. Digital Twin framework for manufacturing. Geneva : ISO, 2021.
62. ISO/IEC 25010. https://iso25000.com/en/iso-25000-standards/iso-25010
63. IEC 62264-1:2013. Enterprise-control system integration. https://www.iso.org/standard/57308.html
64. ISO/IEC 14977:1996. Extended BNF. https://www.iso.org/standard/26153.html
65. ISO/IEC 23005-1:2016. Media context and control. https://www.iso.org/standard/65394.html
66. ISO/IEC 23005-5:2019. Data formats for interaction devices. https://www.iso.org/standard/73438.html

67. Introducing JSON. https://www.json.org/json-en.html
68. Introduction to XML. https://www.w3schools.com/xml/xml_whatis.asp
69. Bass L., Clements P., Kazman R. Software Architecture in Practice. 3rd ed. Addison-Wesley, 2012.
70. Dychka I. A., Sulema Ye. S. Logical Operations in Algebraic System of Aggregates // Наукові вісті КПІ. 2018. № 6. С. 44–52.
71. Dychka I. A., Sulema Ye. S. Ordering Operations in Algebraic System of Aggregates // Наукові вісті КПІ. 2019. № 1. С. 15–23.
72. Mathematical Methods in Interdisciplinary Sciences / Sulema Ye. et al. Wiley, 2020. 464 p.
73. Sulema Ye. Multimodal data processing based on algebraic system of aggregates // Radio Electronics, Computer Science, Control. 2020. № 1. С. 169–180.
74. Allen J. F., Hayes P. J. Moments and points in interval-based temporal logic // Computational Intelligence. 1989. Vol. 5. Issue 3. P. 225–238.
75. JSON-LD 1.1. https://www.w3.org/TR/json-ld11/
76. RAMI 4.0 Reference Architectural Model. https://www.isa.org/intech-home/2019/march-april/features/rami-4-0-reference-architectural-model-for-industr
77. Bader S. et al. The Asset Administration Shell as the Digital Twin Backbone // IECON Proceedings. 2021.
78. Plattform Industrie 4.0. AASX Package Explorer – Documentation. 2020.
79. Pfrommer J. et al. OPC UA vs. REST: A Comparison // IEEE ETFA. 2015.
80. Ye X., Hong S. AutomationML/OPC UA-based Industry 4.0 Solution // IEEE ETFA. 2018. P. 543–550.
81. Olivier B. et al. The Ultrasound File Format (UFF) // IEEE International Ultrasonics Symposium. 2018.
82. The NIFTI file format. https://brainder.org/2012/09/23/the-nifti-file-format
83. eHealth. https://www.who.int/ehealth/en/
84. Mobile Health: A Technology Road Map. Springer Cham, 2015. https://doi.org/10.1007/978-3-319-12817-7
85. What is Hospital at Home? https://www.hospitalathome.org.uk/whatis
86. Ghinea G., Andres F., Gulliver S. Multiple Sensorial Media Advances and Applications: New Developments in MulSeMedia. Information Science Reference, 2012. 320 p.
87. Kovács P.T., Murray N., Rozinaj G., Sulema Ye., Rybárová R. Application of Immersive Technologies for Education: State of The Art. Proceedings of the International Conference on Interactive Mobile Communication, Technologies and Learning IMCL2015. Thessaloniki, Greece, 2015, P. 283–288.
88. Pester A., Sulema Y. Multimodal Data Representation Based on Multi-image Concept for Immersive Environments and Online Labs Development. Advances in Intelligent Systems and Computing, Vol. 1231, Springer Nature Switzerland AG, 2021.