ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КИЇВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Кафедра інформаційних систем та технологій

І.Р. ПАРХОМЕЙ
О.С. БОНДАРЕНКО

ПРОЕКТУВАННЯ IOT-РІШЕНЬ ДЛЯ ПРОМИСЛОВИХ СИСТЕМ

навчальний посібник
для здобувачів освітнього ступеня «магістр» спеціальності
F6 «Інформаційні системи та технології»
освітня програма «Програмні технології інтернет речей»

Київ – 2025

Рецензенти:

д.т.н., професор Жураковський Б.Ю.
к.т.н, старший науковий співробітник Яровий О.В.

Рекомендовано до публікації кафедрою інформаційних систем та технологій, протокол № 09_24/25 від «06» лютого 2025 р.

Рекомендовано до публікації Вченою радою факультету інформаційних технологій, протокол №13 від «12» травня 2025 р.

І.Р. Пархомей, О.С. Бондаренко
Проектування IoT-рішень для промислових систем [Електронний ресурс]: Навчальний посібник для здобувачів освітнього ступеня «магістр» спеціальності F6 «Інформаційні системи та технології» освітня програма «Програмні технології інтернет речей» / Укл. І.Р. Пархомей, О.С. Бондаренко. – К.: КНУ імені Тараса Шевченка, 2025.– 127 с.

Навчальний посібник спрямований на поглиблення знань студентів у сфері проектування IoT-рішень для промислових систем. Розкриті такі основні аспекти: основи технологій інтернету речей, сучасні методології розробки, налаштування IoT-систем та їх інтеграція у промислові процеси. Посібник містить теоретичний матеріал, приклади використання у середовищах Smart City та Big Data. Призначений для студентів 2-го курсу спеціальності F6 «Інформаційні системи та технології» освітньої програми «Програмні технології інтернет речей» освітнього ступеня «магістр».

Публікується в авторській редакції.

Електронна версія цього видання опублікована на сайті кафедри інформаційних систем та технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка за адресою: https://www.ist.fit.knu.ua/

_____

(дата публікації «___» _____ 20___ р.)

ЗМІСТ

ВСТУП

Сучасний світ переживає революційні зміни в області інформаційних технологій. Одним із найбільш значущих напрямів є впровадження технологій Інтернету речей (IoT), які інтегрують фізичні пристрої, датчики, програмне забезпечення та мережеві технології для створення розумних систем. У промислових системах технології IoT відкривають нові можливості для автоматизації виробничих процесів, підвищення їх ефективності, зменшення витрат і забезпечення високого рівня контролю та управління.

Навчальний посібник «Проектування IoT-рішень для промислових систем» розроблено з метою формування теоретичних знань і сприяння розвитку практичних навичок, необхідних для проектування IoT-рішень у складних промислових середовищах. Основна увага приділяється використанню сучасних методологій та технологій для створення адаптивних, ефективних і безпечних систем, здатних вирішувати актуальні задачі виробничої сфери.

Посібник охоплює ключові теми, зокрема сучасні підходи до розробки IoT-рішень, інтеграцію технологій IoT з Big Data та Smart City, а також забезпечення інформаційної безпеки в таких системах. Особливий акцент зроблено на вирішенні завдань, пов’язаних із впровадженням IoT у промислових системах, таких як управління обладнанням, моніторинг процесів та оптимізація ресурсів.

Посібник розроблено як практичний інструмент для здобувачів освіти, які прагнуть опанувати методи проектування IoT-рішень та розвивати інноваційні технології у промисловій сфері, що сприятиме ефективному впровадженню новітніх рішень у реальних проектах.

ТЕМА 1. СУЧАСНІ ТЕХНОЛОГІЇ ПРОЕКТУВАННЯ IOT ПРОМИСЛОВИХ СИСТЕМ

1.1. Інтернет речей

Інтернет речей (IoT) являє собою розподілену мережу мереж, що об'єднує унікально ідентифіковані об'єкти («речі»), здатні взаємодіяти між собою в автономному режимі – без участі людини – через IP-з'єднання. Принципово важливо, що йдеться про значно складніший феномен, ніж просте поєднання сенсорних пристроїв.

Збір та аналіз даних про фізичні об'єкти – механізми, будівлі, людей тощо – за допомогою датчиків практикується вже тривалий час. Проте промисловий інтернет речей (IIoT) кардинально відрізняється від традиційних підходів: датчики тут інтегруються в єдину мережеву інфраструктуру разом із системами аналітики та/або керування. Унаслідок цього на рівні окремого об'єкта формується відносно автономна мережева структура. У її межах відбувається безперервний обмін даними, на підставі якого автоматично виробляються рішення та реалізуються керуючі впливи на об'єкт. Фактично утворюється саморегульована система з елементами штучного інтелекту.

Існує п'ять основних секторів для Інтернету речей, апаратних інновацій та зростання ринку в споживчому просторі: мережеві будинки, охорона здоров'я, роботи, дрони та транспортування [1].

1.1.1. Вплив IoT на умови і середовище виробничого процесу

В індустріальному середовищі IoT – це дані і способи їх отримання. Використання технологічного обладнання у виробничих процесах може сприяти підвищенню прибутковості підприємств за умови впровадження стратегічного підходу до збору й аналізу даних про загальну ефективність. Зокрема, йдеться про забезпечення безвідмовної роботи та підвищення надійності обладнання, інтеграцію механізмів ручного введення даних для оперативного коригування

технологічного процесу, а також проведення оцінки енергоефективності та витрат на виробництво продукції.

Кількість публікацій про технологію IoT, як і саме її впровадження, зростає надзвичайно швидко. У нещодавньому прес-релізі вказано, що до 2024 року частка цифрових пристроїв у виробничому середовищі збільшиться до $13 млрд. З урахуванням таких високих темпів зростання, компанії намагаються отримати значну конкурентну перевагу шляхом активного впровадження нових рішень для підвищення прибутку [2].

У промисловому контексті IoT насамперед асоціюється з отриманням даних та способами їх отримання. Цього можна досягти шляхом підключення інтелектуальних пристроїв до виробничих мереж і подальшого аналізу вхідної інформації. Водночас значна частина уваги була зумовлена маркетинговими стратегіями, що формували у компаній враження про необхідність активного інвестування у даний напрям. Таке позиціювання базується переважно на оцінках потенційного зростання застосування технології, а не на її безпосередній практичній цінності у конкретних умовах.

У загальному випадку концепція IoT не є новою. Ця технологія вже понад 15 років присутня у повсякденному житті. Люди використовують свої смартфони для нагадування про заплановані зустрічі або отримання інформації щодо найоптимальніших маршрутів. Застосування інтелектуальних пристроїв у споживчому середовищі дозволяє підвищити ефективність повсякденних дій без значного ризику. Проте така стратегія не завжди є ефективною в індустріальному середовищі, безпосередньо пов’язаному з виробничими процесами.

1.1.2. Переваги виробництва з використанням технології IoT

Незважаючи на явні переваги, технологія IoT не якась наднова сила з області наукової фантастики, яка переміщається по всьому вашому підприємству уздовж і поперек і чарівним чином забезпечує 100% ефективність обладнання (англ. Overall equipment effectiveness, OEE). Тому будь-яка реалізація

підключених пристроїв повинна бути ретельно розглянута, причому не постфактум, а заздалегідь. Для того щоб заробити більше грошей для бізнесу, залучення у «революцію IoT» має ґрунтуватися на стратегічному підході до збору даних, а не просто бути частиною деякого загального модного руху.

Звичайно, будь-який фахівець, пов'язаний з індустріальним виробництвом, усвідомлює, що розширення доступу до інформації забезпечує додаткові переваги. Збільшення джерел доходу ґрунтується на розширенні збору даних, що охоплює моніторинг показників ефективності, підвищення часу безвідмовної роботи та покращення експлуатаційної надійності обладнання, можливість своєчасного ручного втручання для коригування процесів, оцінювання енергоефективності, а також здатність оперативно розраховувати собівартість продукції на одиницю обладнання. Завдяки великій кількості технологічних змін і вдосконалень, що відбуваються щодня, перелік потенційних переваг постійно зростає. Втім, у стратегічному розрізі компаніям варто поставити запитання: яких саме результатів можна досягти на конкретному виробничому об’єкті шляхом покращення контролю над процесами, подовження терміну служби обладнання та зниження загальної вартості його володіння (англ. Total Cost of Ownership, TCO)?

У дискретному виробництві, де кінцевий продукт формується шляхом збирання окремих деталей і компонентів, спостерігається тенденція до спрощення контролю процесів завдяки обмеженій кількості змінних і чітко визначеним параметрам. Це створює сприятливі умови для впровадження підходів, орієнтованих на обробку великих обсягів даних. Проте ефективність таких рішень у межах дискретного виробництва не означає, що вони можуть бути однаково успішними в усіх типах виробничих процесів.

У безперервному виробництві – зокрема, у харчовій та фармацевтичній галузях – навіть незначні зміни умов можуть істотно впливати на якість кінцевого продукту. Візьмемо для прикладу додавання ароматизатора до напою: це важливий етап виробництва, де правильно обраний момент додавання відіграє

вирішальну роль. Контакт із повітрям або надмірна вологість можуть викликати небажані реакції, зокрема ферментацію. Тому час і умови додавання компонентів мають суворо контролюватися відповідно до технологічних вимог. Загалом, якість готової продукції залежить від взаємодії багатьох виробничих факторів, які потребують постійного спостереження для забезпечення стабільної якості кожної виробничої партії.

Виявляючи і аналізуючи стійкі варіації як загальних, так і окремих параметрів процесу, що впливають на результат виробництва, інженери-технологи розробляють методи контролю, які сприяють підвищенню відтворюваності технологічних операцій. Із залученням розширеного масиву даних ці методи забезпечення відтворюваності можуть бути вдосконалені з більшою точністю та деталізацією, що безпосередньо визначає якісні характеристики готової продукції.

Ключовою перевагою впровадження технології Інтернету речей (IoT) у виробництві є забезпечення доступності та прозорості даних і процесів у режимі реального часу. Однак, попри важливість доступу до оперативної інформації, не варто обмежуватися лише цим аспектом технології.

Виробничим підприємствам доцільно не лише аналізувати дані, отримані в режимі реального часу, але й систематично використовувати їх для розробки прогнозуючих моделей і алгоритмів, які в подальшому скоротять виробничі похибки до прийнятного рівня. здатних мінімізувати виробничі відхилення до допустимого рівня. Превентивний підхід становить основу оптимальної стратегії в умовах сучасного виробництва.

Принципове значення має визначення характеру діяльності підприємства: чи функціонує воно за реактивною моделлю, реагуючи на події постфактум, чи застосовує проактивний підхід до управління виробничими процесами? За умови постійного застосування реактивного підходу, єдиним механізмом удосконалення виробництва залишається прискорення реакції на швидкозмінні умови та інтенсифікація робочих зусиль. Доцільніше реалізувати

високопродуктивні графічні інтерфейси з інтегрованими алгоритмами інтелектуального моделювання, аніж ретроспективно реагувати на події та формувати звітність щодо причин низької ефективності виробничих показників.

У високопродуктивному індустріальному середовищі оператор має можливість своєчасно ідентифікувати сигнальні індикатори при їх появі та оцінити ступінь критичності ситуації. Прогностичний модуль системи управління, запрограмований на автоматичне виконання дій відповідно до попередньо затверджених алгоритмів корекції помилок, активує запит на втручання оператора виключно у випадках виникнення нестандартних проблем, для яких не було розроблено типових процедур реагування.

Інтернет речей набуває все більшого поширення, і важко передбачити, до яких масштабів може сягнути його розвиток. Багато фахівців вважають, що ця технологія стане однією з складових четвертої промислової революції, відомої як «Індустрія 4.0». Однак замість того, щоб просто долучатися до швидкого розвитку цієї технології, необхідно провести низку досліджень, щоб оцінити переваги та можливі ризики. Використання даних в режимі реального часу може підвищити ефективність і продуктивність підприємства, проте зростання обсягів даних також спричиняє нові виклики в контексті безпеки [2].

Особливо важливо проводити обговорення щодо конкретних потреб виробничих процесів з інженерами та системними інтеграторами, які мають досвід у вдосконаленні цих процесів та впровадженні рішень для роботи з великими обсягами даних. Досі було поширене переконання, що чим більше даних збирається, тим ефективнішим стає виробництво. Це справедливо, однак взаємозв'язок між рентабельністю інвестицій (англ. Return on investment, ROI) та обсягами зібраних даних не є лінійним і може змінюватися під впливом ризиків, пов'язаних з технологією Інтернету речей.

1.3. Сучасні концепції створення інформаційних систем

Сучасні концепції створення ІС ґрунтуються на таких підходах.

Концепція об’єктно-орієнтованого підхіду дозволяє представити завдання ділення інформаційних систем як задачу створення ієрархії елементів, що сполучаються. Тому елементи на рівнях є елементами визначених множин і відповідно мають визначені ознаки властивості. Інформаційні системи в об’єктно-орієнтованому представленні визначаються спеціалізованими діаграмами та іншими ознаками. Перевагою є наглядність графічного представлення і практичне його застосування мовою UML, яка є уніфікованим засобом моделювання [1,2,3].

Використання UML (Unified modeling language), як мови моделювання і графічної інтерпритації дозволяє візуалізувати, розподілити, обєднати та описати елементи систем. В них вирішальну роль отримує програма та її реалізація. UML-діаграма відображує подробиці плану утвореної системи. Це віддзеркалює її концептуальні елементи, а саме, системні функції та бізнес-процеси. Конкретні способи реалізації, детальні класи, які створені вищими мовами програмування, схеми та бази даних, а також елементи програм, які багаторазово використовуються.

Під терміном Computer Aided System Engineering (CASE) розуміють технологію автоматизованого проектування інформаційних систем (ІС), орієнтовану на моделювання складних інформаційних середовищ. До складу CASE входять програмні компоненти, що забезпечують підтримку процесів створення та супроводження ІС, зокрема аналізування й формування вимог. Крім того, дана технологія охоплює проектування додатків і прикладного програмного забезпечення для роботи з базами даних, автоматичне генерування коду, тестування, документування, управління конфігурацією та проектом, а також низку інших суміжних процесів.

Інструментарій CASE-технологій надає засоби для наочного моделювання довільної предметної області. Аналіз побудованих моделей на кожному етапі

розробки та супроводження ІС дає змогу створювати прикладні програмні рішення, що відповідають інформаційним потребам користувачів. Більшість сучасних CASE-засобів базується на методах структурного та об'єктно-орієнтованого проектування [4,5], що передбачає застосування формальних специфікацій у вигляді діаграм або текстових описів, які відображають зовнішні вимоги, взаємозв'язки між моделями системи та динаміку її поведінки як елемента програмної архітектури [5].

Технологія структурного моделювання SADT (Structure Analysis and Design Technique) призначена для побудови моделей функціонування окремих елементів предметної області. Її основна мета полягає в описі складних ієрархічних багаторівневих модульних систем із використанням обмеженого набору типових конструктивних елементів. До ключових характеристик SADT-технології належать:

1) ієрархічна декомпозиція за принципом «згори вниз»;
2) підтримка моделювання багаторівневих систем;
3) спеціалізована структуризація баз даних.

Сучасні концепції побудови систем підтримки прийняття рішень (СППР) дозволяють подолати суперечність між браком корисної інформації з точки зору споживача та надлишком даних з точки зору технічних можливостей системи. Серед підходів, що забезпечують підвищення ефективності зберігання й використання інформації, виділяють такі:

- Data Warehouse – технологія організації структур даних та їх сховищ;
- Data Mart – інформаційна вітрина для цільового представлення даних;
- OLAP (On-Line Analitical Processing) – багатовимірний аналіз даних у режимі реального часу;
- Data Mining (DM) – інтелектуальний аналіз даних із застосуванням методів машинного навчання [4].

1.4. Особливості проектування інформаційних систем

Сучасна тенденційність інформаційних технологій створює зростаюче ускладнення інформаційних систем. Створювані у різноманітних областях повсякденного життя, сучасні big - проекти ІС мають , часто-густо наступні особливості:

1) значна кількість функціональних процесів, складових даних і розгалужених взаємозв'язків між ними суттєво ускладнює їх опис та потребує ретельного моделювання й аналізу;

2) наявність сукупності взаємопов'язаних підсистем, що функціонують у тісній взаємодії та виконують локальні завдання — зокрема обробку транзакцій, виконання регламентних задач, аналітичну обробку та підтримку прийняття рішень — зумовлює необхідність формування нерегламентованих запитів до великих масивів даних;

3) обмежена можливість застосування типових проектних рішень і готових прикладних систем за відсутності прямих аналогів у конкретній предметній області;

4) необхідність забезпечення сумісності наявних додатків із перспективними розробками;

5) потреба у функціонуванні в неоднорідному середовищі з використанням кількох апаратних платформ одночасно;

6) усталені традиції застосування різнорідних наборів інструментальних засобів у командах розробників у поєднанні з неоднорідністю рівня кваліфікації персоналу;

7) обмежені можливості команди розробників та необхідність врахування масштабів організації-замовника з різним ступенем готовності окремих її підрозділів до впровадження ІС, що суттєво впливає на тривалість реалізації проєкту.

Успішне виконання проєкту та досягнення цілей проектування передусім залежить від детального опису, повноти побудови та несуперечливості

функціонування інформаційної моделі ІС. Це складна, трудомістка і тривала робота, що вимагає високого рівня кваліфікації залучених фахівців, що підтверджує накопичений досвід проектування ІС. Ще донедавна розробка ІС здійснювалася переважно на інтуїтивному рівні із застосуванням неформалізованих методів, що спиралися на практичний досвід, експертні оцінки та дорогі експериментальні дослідження якості функціонування систем. Додаткову складність становить те, що в процесі створення та експлуатації ІС інформаційні потреби користувачів можуть змінюватися або уточнюватися, що ще більше ускладнює розробку та подальший супровід таких проєктів.

На початку століття при розробці ІС широко застосовувалася структурна методологія, що надає розробникам чіткі формалізовані методи опису системи та прийнятих технічних рішень. В її основі лежить наочна графічна техніка представлення різноманітних моделей ІС із використанням схем і діаграм. Завдяки наочності та адекватності засобів структурного аналізу проектувальники і майбутні користувачі системи могли брати участь у її створенні вже з початкових етапів, формуючи та закріплюючи спільне розуміння ключових технічних рішень. Проте широке практичне застосування цієї методології було суттєво обмежене: за умов неавтоматизованого виконання робіт ручна розробка та інтерпретація формалізованих специфікацій, перевірка їх на повноту й несуперечливість, а тим більше внесення змін, були практично нездійсненними. Ручна розробка, як правило, призводила до таких проблем:

1) неадекватність сформульованих вимог;
2) помилки у визначенні проектних рішень;
3) погіршення якості експлуатаційної документації;
4) тривалість і незадовільні результати тестування.

Підходи до створення інформаційних систем

Базовою ідеєю є традиційність підходів до побудови інформаційних систем. На початкових етапах проєкту, як правило, складно визначити повний обсяг даних і перелік аналітичних задач, які вирішуватимуться кінцевими

користувачами. Методологія Oracle Data Warehouse Method Fast Track (DWM FT) передбачає впорядкування сховищ даних за принципом «швидкісної траси», де вимоги до сховища визначаються та аналізуються впродовж усього життєвого циклу системи. Заснована на концепції Dynamic System Development Method (DSDM), вона узагальнює підхід до розробки динамічних систем і використовує методологію Rapid Application Development (RAD) — швидкого створення застосунків.

Цикл проектування за DSDM і Oracle DWM FT реалізується через послідовне створення прототипів аж до повного задоволення вимог кінцевих користувачів. Щоб запобігти нескінченному повторенню циклу, розробка поділяється на фіксовані 120-денні часові відрізки (timebox), протягом яких виконується чітко визначений перелік вимог. За аналогією з контейнером фіксованого розміру, до якого не можна додати нові предмети без вилучення наявних, цей підхід дисциплінує процес розробки. Гнучкість та простота використання інструментів Business Intelligence роблять прототипування доступним і ефективним, що є особливо зручним для IT-фахівців.

Методологія RAD набула широкого застосування у сучасній практиці створення автоматизованих інформаційних систем і охоплює всі етапи їх життєвого циклу. Її основні принципи такі:

1) використання спіральної моделі розробки;

2) необов'язковість повного завершення робіт на кожному етапі життєвого циклу;

3) прискорена розробка додатків із застосуванням CASE-засобів;

4) паралельне здійснення тестування та розвитку проєкту.

RAD забезпечує реалізацію нової технології створення ІС, за якої об'єкти формуються у вигляді діючих моделей або прототипів. Їх функціональність узгоджується з користувачем, після чого розробник переходить до остаточного формування додатків, не втрачаючи цілісного бачення проектованої системи.

Одним із ключових засобів методології RAD є візуальне програмування, що ґрунтується на принципах об'єктно-орієнтованого підходу. Його перевага полягає у використанні стандартних інтерфейсів та об'єктів двох основних груп: перша охоплює списки, вікна і текстові поля, що легко пов'язуються з базами даних і відображаються на екрані; друга — стандартні елементи керування: кнопки, перемикачі, прапорці та меню, за допомогою яких здійснюється управління даними.

Метод RAD добре зарекомендував себе при розробці відносно невеликих програмних продуктів. Однак при створенні складних корпоративних транзакційних систем виникає потреба у трансформації бізнес-процесів і формуванні архітектури, орієнтованої на реалізацію чітко визначеної поведінки. Сьогодні найперспективнішим напрямом у цьому контексті є бізнес-орієнтований підхід, що базується на системі збалансованих показників Balanced Scorecard (BSC).

З самого початку проєктування визначається бізнес-аспект за допомогою підходів BSC, після чого на основі аналізу даних здійснюється проектування інформаційно-аналітичної системи за принципом низхідного проектування паралельно з впровадженням на підприємстві управління за цілями — Management by Objectives (MBO) [4].

Сучасні засоби візуальної розробки додатків поділяються на дві категорії: універсальні та спеціалізовані. Серед універсальних систем програмування найбільшого поширення набули Java та C++, тоді як до спеціалізованих належать Oracle та MySQL.

Для проектування відносно невеликих баз даних може застосовуватися ручний підхід. Проте за наявності десятків і сотень взаємопов'язаних таблиць організація даних та встановлення зв'язків між ними стає надзвичайно складним завданням. Для вирішення цієї проблеми протягом останніх десятиліть сформувався окремий напрям у сфері програмної інженерії — технології

Computer-Aided Software/System Engineering (CASE), що являють собою системи автоматизованої розробки програмного забезпечення.

1.5. Методології розробки інформаційних систем

Кожна теоретична або практична галузь діяльності має притаманні їй способи розв'язання поставлених завдань і досягнення визначеної мети. Такі способи прийнято називати методами [4,6]. Метод у загальному розумінні [8] — це цілеспрямований спосіб досягнення певної мети, що охоплює сукупність прийомів і операцій як практичного, так і теоретичного освоєння дійсності, застосованих для вирішення конкретної задачі.

Методологія [6] являє собою систему методів, що використовуються в певній галузі людської діяльності. Надалі під методологією розумітимемо впорядковану сукупність методів, об'єднаних спільним філософським підходом і застосовуваних протягом усього життєвого циклу розробки.

Методологія науки [6] характеризує основні складові наукового дослідження: його об'єкт, предмет аналізу, дослідницькі завдання, а також сукупність засобів, необхідних для їх вирішення. Крім того, вона формує уявлення про логіку та послідовність дій дослідника в процесі досягнення поставленої мети.

Методологія створення інформаційних систем [6] визначає порядок організації процесу побудови ІС та забезпечує керування ним таким чином, щоб гарантувати дотримання вимог як до функціональних характеристик самої системи, так і до параметрів процесу її розробки.

Методологія створення інформаційних систем забезпечує виконання наступних основних завдань:

- забезпечення відповідність цілям і завданням та висунутим вимогам створюваних інформаційних систем;

- створення системи з заданими параметрами та гарантія протягом заданого часу за визначених обсягів коштів;

- розширення системи на підставі спрощення супроводу, модифікації та за мінливих умов функціонування системи;

- відповідність інформаційних систем вимогам відкритості, перенесення та масштабованості;

- можливість використання програмного забезпечення, баз даних, засобів обчислювальної техніки, телекомунікацій у створюваній системі.

Обмежена кількість методологій сьогодні, особливо комплексних, охоплюють усі відтинки існування програмного забезпечення. Тому методологія визначає набір мов програмування та систем. Вона застосовується під час розробки програмного забезпечення. Це значною мірою визначає технологічність підходів.

Контрольні питання

1. Опишіть UML, його призначення та основні переваги у проектуванні інформаційних систем.

2. Охарактеризуйте методологію RAD, її ключові принципи та випадки застосування для розробки інформаційних систем.

3. Назвіть основні принципи методології SADT та поясніть, у чому полягають її переваги для моделювання предметних областей.

4. Опишіть CASE-технологію, її основні компоненти та роль у створенні й супроводженні інформаційних систем.

5. Охарактеризуйте концепції Data Warehouse, OLAP та Data Mining та поясніть їхній вплив на процеси зберігання, аналізу та обробки даних у сучасних інформаційних системах.

ТЕМА 2. МЕТОДОЛОГІЯ РОЗРОБКИ ІОТ ПРОМИСЛОВИХ СИСТЕМ

2.1 Методології розробки інформаційних систем

Аналізуючи вітчизняну літературу на цю тему, звернемося до класифікації методологій, поданої в книзі Одинцова І.О. "Професійне програмування. Системний підхід".

Методології створення інформаційних систем можна класифікувати за низкою характерних ознак.

2.1.1. Класифікація за ядрами методологій

Існує ядро методології зі своїми методами, яке доповнюється певними специфічними особливостями. Зазначений підхід є аналогічним принципу словотворення в українській мові, де наявний корінь слова збагачується префіксами, суфіксами та закінченнями, що конкретизують його значення.

В основі класифікації методологій лежить спосіб опису алгоритмів. Відповідно до цього критерію виокремлюють такі базові ядра методологій:

1) методологія імперативного програмування;
2) методологія об'єктно-орієнтованого програмування;
3) методологія функціонального програмування;
4) методологія логічного програмування;
5) методологія програмування в обмеженнях.

Розглянемо ядра методологій докладніше.

Методологія імперативного програмування – підхід, що ґрунтується на принципі послідовної зміни стану обчислювальної системи у покроковому режимі. Імперативне програмування є історично першою методологією, апаратно підтримуваною на рівні машинної архітектури. Вона орієнтована на класичну модель фон Неймана, яка протягом тривалого часу залишалася

домінуючою апаратною архітектурою і набула широкого практичного застосування.

Основні методи і концепції:

- Метод зміни станів полягає в послідовному переході між станами системи. Його реалізація ґрунтується на концепції алгоритму.
- Метод управління потоком виконання передбачає покроковий контроль виконання команд. Його реалізація базується на концепції потоку управління.

Методологія об'єктно-орієнтованого програмування являє собою підхід, в основу якого покладено принцип об'єктної декомпозиції. Відповідно до цього принципу статична структура системи подається через сукупність об'єктів та зв'язків між ними, тоді як динамічна поведінка системи описується через механізм передачі повідомлень між цими об'єктами. Формуванню об'єктного мислення сприяли дослідження у сфері моделювання та подання даних, графічних інтерфейсів і системного програмування (зокрема, пов’язаного з поняттям «процес»). Дослідження в області хешування реальних систем виявили потребу у створенні засобів опису сутностей, що існують у них, – об'єктів і подій. Згодом було встановлено, що такі концепції, як інкапсуляція (абстрактні типи даних), успадкування і поліморфізм, є корисним доповненням до традиційного структурного програмування. Ефективність їхньої реалізації стала передумовою для появи об’єктно-орієнтованих мов програмування, які набули широкого поширення у сучасній практиці.

Основні методи і концепції:

- Метод об'єктно-орієнтованої декомпозиції полягає у виділенні об'єктів і зв'язків між ними. Метод підтримується концепціями інкапсуляції, успадкування та поліморфізму.
- Метод абстрактних типів даних лежить в основі інкапсуляції. Метод підтримується концепцією абстрагування.

• Метод пересилки повідомлень полягає в описі поведінки системи в термінах обміну повідомленнями між об'єктами. Метод підтримується концепцією повідомлення.

Методологія функціонального програмування – підхід до створення програм, в якому основною ідеєю є виклик функції; основним засобом розподілу програми на складові – надання імен функціям і визначення для них виразів, що обчислюють відповідні значення; а основним принципом побудови – суперпозиція функцій. Функціональна методологія є однією з найстаріших. За походженням вона тісно пов'язана з лямбда-обчисленням, запропонованим ще на початку 90-х років XX століття американським логіком Алонзо Черченом. Ця методологія широко застосовується у теоретичному програмуванні та слугує інструментом для проведення досліджень у галузі штучного інтелекту [6].

Основні методи і концепції:

• Метод аплікативного програмування ґрунтується на представленні програми у вигляді виразу, що описує зв'язки між функціями та їхніми аргументами. Програма являє собою сукупність визначень функцій, кожна з яких, своєю чергою, є викликом інших вкладених функцій. В основі цього методу лежить концепція функції як базового будівельного елемента програми.

• Метод рекурсивної поведінки базується на принципі самоповторення, коли виконання процесу повертається до власного початку. Теоретичним підґрунтям цього методу є концепція рекурсії.

• Метод настроюваності забезпечує можливість зручного створення нових програмних об'єктів за заданим зразком як результату застосування функцій до параметрів цього зразка. Це стає можливим завдяки тому, що будь-який програмний об'єкт, як і сама програма, в ідеалі є виразом.

Методологія логічного програмування – це підхід, за якого програма містить формалізований опис задачі у термінах фактів і логічних формул, а її розв'язання здійснюється системою автоматично за допомогою механізмів логічного виведення. Витоки логічного програмування сягають кінця 1960-х

років, коли Корделл Ґрін запропонував використовувати метод резолюції як основу логічного виведення у програмуванні. У 1972 році Ален Колмерое (Alain Colmerauer) розробив мову логічного програмування Prolog. Найвищого поширення ця парадигма набула в середині 1980-х років, коли її було покладено в основу масштабного проєкту зі створення програмного та апаратного забезпечення обчислювальних систем п'ятого покоління.

Основні методи і концепції:

- Метод однаковості полягає в універсальному застосуванні єдиного механізму логічного доведення до всіх частин програми без винятку.
- Метод уніфікації реалізує механізм зіставлення зі зразком, що використовується для побудови та декомпозиції структур даних.

Методологія програмування в обмеженнях – це підхід, за якого в програмі задаються тип шуканого розв’язку, предметна область та обмеження на допустимі значення змінних. Пошук розв’язку здійснюється системою автоматично. Методологія передбачає дворівневу архітектуру, що інтегрує компонент обмежень і програмний компонент. Компонент обмежень забезпечує базові операції й містить систему виведення, побудовану на фундаментальних властивостях обмежень. Операції, що супроводжують роботу компонента обмежень, реалізуються засобами мов програмування. Ця методологія виникла на початку 1990-х років як перспективний напрям досліджень на перетині символьних обчислень, штучного інтелекту, дослідження операцій та інтервальної арифметики [6].

Метод описової моделі обчислень полягає в тому, що програма містить декларативний опис понять і завдань, які вона має реалізовувати. Метод підтримується концепцією моделі обчислень.

2.1.2. Класифікація за топологічною специфікою методологій

Топологічна специфіка (топологія) методологій визначається як спосіб вибору методів для отримання уточненого ядра методології. Критерієм якості

топологій є кількість загальних витрат на розробку програмного забезпечення. Витрати формуються під впливом різноманітних факторів, у тому числі пов'язаних з абстракціями даних, організацією управління та структурою модулів. Наприклад, до якісної топології призводить відмова від використання глобальних даних та оператора безумовного переходу (окрім спеціальних випадків), висока зв'язність модулів і слабке зчеплення між ними.

Методологія структурного імперативного програмування є підходом, спрямованим на формування оптимальної топологічної організації імперативних програм із метою мінімізації сукупних витрат на розробку програмного забезпечення. Досягнення цієї мети забезпечується завдяки чіткій структурованості як проектних моделей, так і програмного коду, що безпосередньо сприяє скороченню кількості помилок у процесі розробки. У межах даної методології до бажаної топології ведуть такі принципи: відмова від використання глобальних даних і, здебільшого, від безумовних переходів, створення модулів із високою внутрішньою узгодженістю та мінімальною залежністю від інших модулів. Підхід базується на двох ключових принципи побудови: послідовна ієрархічна декомпозиція алгоритму розв'язання задачі; використання структурованого програмного коду. Ця методологія є важливим етапом розвитку імперативної методології. Її автором вважається Едсгер Дейкстра, який також зробив спробу поєднати структурне програмування з формальними методами доведення коректності програм.

Метод алгоритмічної декомпозиції полягає у покроковому уточненні задачі, починаючи з найбільш загального рівня і переходячи до деталей. Даний метод забезпечує структурованість програми і підтримується концепцією алгоритмічної побудови.

Метод модульної організації програми передбачає розподіл програми на логічно завершені компоненти – модулі. Метод ґрунтується на концепції модуля.

Метод структурного кодування полягає у використанні трьох базових керуючих конструкцій: слідування, розгалуження та повторення. Метод реалізує концепцію управління виконанням програми.

2.1.3. Класифікація за реалізаційною специфікою методологій

Кожне з ядер методологій має певну специфіку, що визначає організацію апаратної підтримки, орієнтованої на цю методологію. На даний момент найбільш поширеними є дві основні організації: централізована та паралельна.

Ядра методологій первинно розроблялися для централізованих архітектур. Згодом з'явилися паралельні апаратні реалізації, до яких почали адаптувати вже існуючі методології. Прикладом паралельної методології є методологія імперативного паралельного програмування – підхід, що передбачає використання явних конструкцій для паралельного виконання обраних фрагментів програм. Вважається, що паралельне програмування виникло з появою каналів – незалежних апаратних контролерів, які давали центральному процесору можливість виконувати нову прикладну програму одночасно з операціями введення-виведення інших програм. На початковому етапі з паралельним програмуванням мали справу переважно розробники операційних систем.

До методів даної методології можна віднести метод синхронізації виконуваного коду, який полягає у використанні спеціальних атомарних операцій для забезпечення взаємодії між одночасно виконуваними фрагментами коду. Цей метод підтримується концепцією примітивної синхронізації [6].

2.2. Змішані методології

Змішані методології передбачають комбінування методів, запозичених з різних методологій. Найчастіше поєднуються методології функціонального та логічного програмування. Також існують наукові дослідження, присвячені інтеграції об'єктно-орієнтованого та логічного програмування. Окремий

напрямок досліджень зосереджено на питаннях уніфікації методологій програмування.

2.2.1. Передумови швидкої розробки

На ранніх етапах розвитку комп'ютерних інформаційних систем їх створення здійснювалося за допомогою традиційних мов програмування. Проте зі зростанням складності систем і підвищенням вимог користувачів виникла необхідність у нових інструментах, які дозволяли б суттєво скоротити час розробки. Це сприяло виникненню окремого напряму в сфері програмного забезпечення – засобів для швидкого створення додатків. Подальший розвиток цього напряму привів до появи програмних рішень, що автоматизують майже всі етапи життєвого циклу інформаційних систем. Такі засоби отримали назву методології швидкої розробки додатків RAD (Rapid Application Development).

2.2.2. Методологія швидкої розробки додатків RAD

RAD – це сукупність спеціалізованих інструментів для оперативного проєктування прикладних інформаційних систем, які дають змогу працювати з набором графічних елементів, що функціонально відображають складові додатка.

Під методологією швидкої розробки додатків зазвичай розуміють підхід до створення інформаційних систем, що ґрунтується на трьох ключових складових:

- невелика команда розробників (як правило, від 2 до 10 осіб);
- детально спланований графік робіт, орієнтований на відносно короткий термін виконання (приблизно 2–6 місяців);
- ітераційний підхід до розробки, який передбачає постійну взаємодію із замовником – у процесі виконання проєкту вимоги уточнюються та поступово реалізуються в продукті.

Основні принципи методології:

- застосування ітераційної (спіральної) моделі розробки;

- відсутність необхідності повного завершення кожного етапу життєвого циклу перед переходом до наступного;
- активна взаємодія із замовником і майбутніми користувачами протягом усього процесу;
- використання CASE-засобів, інструментів швидкої розробки, систем керування конфігурацією, засобів внесення змін і супроводу, а також прототипування для точнішого визначення потреб користувачів;
- паралельне виконання тестування та розвитку проєкту;
- робота невеликої, добре організованої команди фахівців;
- необхідність ефективного управління, чіткого планування та контролю виконання завдань.

Засоби RAD забезпечили можливість реалізації принципово нового підходу до створення додатків, який відрізняється від традиційного: інформаційні об’єкти спочатку формуються у вигляді функціональних моделей (прототипів), узгоджених із кінцевим користувачем. Після цього розробник переходить до створення повноцінного програмного продукту, зберігаючи цілісне бачення його архітектури.

Методології, технології та інструментальні засоби проєктування становлять основу розробки будь-якої інформаційної системи [9]. Методологія реалізується через відповідні технології та підтримується стандартами, методиками й інструментами, що забезпечують виконання всіх етапів життєвого циклу системи.

2.3. Класичні та адаптивні методології розробки інформаційних систем

Формально методології можна розділити на два типи – класичні (монументальні, передбачувані, де всі етапи детально проектується заздалегідь і не допускається відхилення від початкового плану) та їх протилежність – адаптивні (гнучкі, де в процесі роботи допустимо і навіть типово

перепроектування). У залежності від початкових умов і характеру поставленого завдання, може застосовуватися як класична, так і адаптивна методологія.

2.3.1. Класична (монументальна) методологія

Ця методологія застосовується за таких умов:

1. Замовник має чітке уявлення про функціональні можливості та дизайн майбутньої програми, тобто він точно усвідомлює та може коректно сформулювати очікувані результати розробки.

2. Завдання, яке вирішується програмним засобом, може бути чітко формалізовано та задокументовано. Це означає можливість створення повноцінного технічного завдання, що охоплює всі елементарні стани функціонування програми, містить повний опис спеціалізованих алгоритмів (у разі їх використання), однозначно інтерпретується всіма сторонами та включає всі аспекти дизайну інтерфейсу користувача. Розробку технічного завдання можуть здійснювати як фахівці виконавця, так і представники замовника.

3. Вимоги до програмного продукту є стабільними, а замовник не передбачає внесення додаткових змін до функціональності в процесі реалізації поточного контракту. Як наслідок, обсяг робіт чітко визначений і фіксований, так само як і загальна вартість розробки.

4. Ітеративність процесу на даному етапі зміщується в площину аналізу, проектування та створення технічного завдання. Процес розробки суворо регламентований і визначений у часі. Формування технічного завдання зазвичай займає близько 60% від загального терміну розробки.

2.3.2. Адаптивна методологія

Адаптивна (agile – гнучка або lightweight – легка) методологія застосовується, якщо:

1) Вимоги до системи не є чітко визначеними або схильні до змін.

2) Замовник має загальне уявлення про розробку програмного забезпечення, але передбачає внесення змін до функціональності або дизайну протягом розробки.

3) Існує потреба у швидкому отриманні перших робочих версій програмного продукту.

4) Розв'язувана програмою задача важко піддається документуванню.

5) На реалізацію всіх вимог є достатній запас часу.

При застосуванні такої методології розробка програми є послідовністю великої кількості ітерацій, кожна з яких займає від тижня до місяця. За результатами кожної ітерації вимоги до програми уточнюються, і, за необхідності, ітерація повторюється. Процес розробки в рамках адаптивної методології є менш бюрократичний, оскільки основний акцент робиться на практичній реалізації, а не на створенні повного продукту чи написанні документації. Замовник має можливість на практиці відшукати найбільш підходяще рішення, яке задовольняє не лише початкові вимоги, але й ті, що можуть з'явитися у процесі розробки (адже рідко який проект обходиться без появи нових вимог). Взаємодія з розробниками є постійною, що дозволяє замовнику бути в курсі поточного стану робіт, активно брати участь у проектуванні наступних етапів та аналізувати результати кожної ітерації [10,12].

Незалежно від обраної методології, робочий процес підпорядковується таким простим, але важливим правилами:

1) Використовується система контролю версій.

2) Код оформлюється згідно з єдиними стандартами.

3) Помилки виправляються першочергово, перед внесенням будь-яких інших змін.

4) Виконується регулярне резервне копіювання всіх проектних даних.

5) Використовуються інструментальні засоби для автоматизації документування вихідного коду і ведення списків помилок.

6) Виконуються різні види тестування, починаючи від створення спеціальних тестових проектів до застосування спеціалізованих інструментів.

Однією з важливих складових успішного проекту є забезпечення ефективної та безперервної комунікації з замовником. Ефективність співпраці та відповідність виконуваної роботи вимогам замовника забезпечуються регулярними візитами до клієнта відповідальних осіб на всіх етапах виконання проекту, а також звітами та доповідями про статус проекту.

Потреба в альтернативі класичним (монументальним) методологіям, які базуються на документації, стала передумовою для проведення семінару у 2020 році, на якому зібралися прихильники різних адаптивних (гнучких) методологій. Результатом роботи став маніфест гнучкої розробки програмного забезпечення (Manifest for Agile Software Development).

Адаптивні (гнучкі) методології набули значного розвитку і активно використовуються в сучасній практиці розробки програмного забезпечення. У подальшому розглянемо їх детальніше, оскільки вони є одними з найбільш перспективних і ефективних підходів на сьогодні.

2.3.3. Методологія SCRUM

Ця методологія призначена для невеликих команд розробників. Проект починається зі створення "резерву властивостей системи" (backlog). Резерв властивостей – це набір функцій системи, які необхідно реалізувати. Окремі функції можуть бути описані через індивідуальні сценарії або традиційні вимоги. Контроль над резервом здійснюється лише однією особою, зазвичай це замовник системи. Резерв постійно оновлюється: функції доповнюються та сортуються за пріоритетами.

Після початкового заповнення резерву розпочинається перша ітерація. Усі роботи поділяються на ітерації (спринти) тривалістю в 30 днів. Тривалість ітерації можна варіюватися залежно від конкретного проекту. Для кожної ітерації обираються функції, що будуть реалізовані. Найважливішою умовою є

незмінність функцій протягом однієї ітерації. Це дозволяє структурувати великий проект з динамічними вимогами на серію фіксованих, короткострокових ітерацій із стабільними вимогами.

Запланована дата закінчення ітерації повинна бути обов'язково дотримана. Команда може не встигнути реалізувати деякі з вибраних функцій, але переносити терміни завершення неможливо.

Особливістю SCRUM є проведення щоденних нарад тривалістю 15-30 хвилин, що називаються scrum-сесіями. На цих нарадах лідер команди ставить кожному члену команди наступні питання:

1) Що вдалося реалізувати з вибраних функцій протягом попереднього дня?
2) Чи виникли будь-які проблеми з реалізацією?
3) Що планується зробити впродовж поточного дня?

Це дозволяє ефективно відстежувати хід проекту, швидко виявляти проблеми та оперативно реагувати на них.

Наприкінці спринту команда представляє працюючий продукт із запланованою функціональністю. Після цього проводиться нарада, на якій обговорюються всі труднощі, з якими зіткнулися розробники, і планується наступна ітерація. Крім того, під час цієї наради можуть бути переглянуті пріоритети та у разі потреби – внесені корективи у подальший план.

2.3.4. Методологія екстремального програмування

Методологія XP (eXtreme Programming або XP) є найбільш відомою серед гнучких підходів до розробки програмного забезпечення. Основними принципами методології є: простота рішень, інтенсивна розробка малими командами (до 10 осіб), активна комунікація як всередині команди, так і між командами, постійна залученість замовника до процесу розробки, а також готовність до ризику та гнучкість у прийнятті рішень. Розробка програмного забезпечення при використанні даної методології здійснюється короткими

ітераціями (тривалістю від одного тижня до місяця) з використанням парного програмування – коли два програмісти одночасно працюють над створенням коду за одним робочим місцем, постійно обговорюючи рішення.

Важливою особливістю методології є прийняття першого рішення. Це пов'язано з високим рівнем невизначеності на ранніх етапах – через обмежений аналіз і стислий графік. Спочатку реалізується лише мінімально необхідна функціональність системи, яка поступово розширюється в наступних ітераціях.

При використанні XP детальне попереднє проектування замінюється безперервною взаємодією з замовником, який входить до складу команди і постійно доступний для уточнення вимог чи оцінки проміжних результатів. Основою проектної документації вважається ретельно прокоментовані коди.

Особливу увагу в методології приділено тестуванню. Як правило, для кожного нового методу спочатку створюється відповідний тест, і лише після цього пишеться код, що забезпечує проходження цього тесту. Усі тести зберігаються у наборах, які автоматично виконуються після будь-якої зміни коду [10].

2.3.5. Сімейство методологій Crystal

Crystal – це не просто методологія, це ціле сімейство методологій, розроблене Алістером Коберном.

Коберн розглядає процес створення ПЗ як кінцеву цілеспрямовану гру і стверджує, що у цієї гри є всього дві мети: головна і допоміжна. Головна мета полягає в тому, щоб успішно закінчити проект, тобто створити працюючий продукт. Допоміжна мета – підготуватися до наступної гри. Для досягнення цієї мети може знадобитися документація, якісне написання коду, тобто все, що полегшує подальший розвиток та підтримку продукту.

Якщо в процесі розробки досягнуті обидві мети, то проект вважається успішним. Крім кінцевого продукту, завжди створюються допоміжні проміжні продукти: моделі, схеми, описи. Їх має бути рівно стільки, скільки необхідно для

досягнення кінцевої мети. Однією з основних проблем є неможливість точно передбачити наперед, які проміжні артефакти проекту будуть дійсно затребуваними, а які – виявляться надлишковими.

Коберн класифікує проекти за двома параметрами: критичність і розмір команди. Під критичністю розуміється величина збитку, нанесеного в результаті використання продукту. Наприклад, помилка у ПЗ для космічного корабля або для апарату штучного дихання непорівняна з помилкою у ПЗ для форуму на сайті. Життєво важливі проекти мають категорію L, а ті, помилка в яких позначає втрату зручностей, категорію C.

Ступінь критичності системи зростає по вертикальній осі, розмір команди – по горизонтальній. Таким чином формується своєрідне сімейство методологій розробки. Чим нижча критичність проекту та менша команда розробників, тим легшу і менш формалізовану методологію доцільно застосовувати. Найпростішою у цьому сімействі є методологія Crystal Clear.

Основні принципи даної методології:

1) Вся команда розробників (до 6 осіб) знаходиться в одному приміщенні. Це дозволяє скоротити часові витрати на комунікацію.

2) Регулярне постачання проміжних версій продукту сприяє формуванню стабільного робочого ритму в проекті. Вчасна поставка результату суттєво підвищує мотивацію команди.

3) Постійна взаємодія з реальними користувачами забезпечує оперативне отримання зворотного зв'язку та усунення недоліків на ранніх етапах розробки.

4) Використання систем контролю версій забезпечує колективний доступ до коду та сприяє ефективній командній роботі.

Сімейство методологій Crystal базується на ітеративному підході до розробки програмного забезпечення. Тривалість кожної ітерації варіюється від одного до чотирьох місяців, при цьому скорочення її тривалості вважається перевагою.

Визначальною рисою Crystal є принцип безперервного вдосконалення та коригування робочих процесів, що забезпечує гнучке пристосування методології до специфіки конкретної команди і проєкту. Серед усіх відомих методологій розробки ПЗ Crystal вирізняється найбільшою увагою до питань адаптації та індивідуального налаштування.

Після завершення кожної ітерації команда проводить спільне обговорення доцільності створення проміжних продуктів, аналізує виявлені недоліки тощо. На основі цих обговорень приймаються рішення щодо змін, які можуть покращити процес, і, за погодженням усіх учасників, відбувається відповідне коригування.

2.3.6. Відкритий вихідний код

Первинно відкритий вихідний код (Open Source) означав не стільки спосіб розробки, скільки тип програмного забезпечення. Проте практика роботи спільноти розробників може бути корисною і для закритих проектів – зокрема для співпраці між командами, які фізично віддалені одна від одної. Це особливо важливо, адже більшість адаптивних методологій припускає спільне фізичне перебування учасників розробки.

У більшості проектів з відкритим вихідним кодом є один або кілька координаторів – лідерів проекту, які мають право безпосередньо змінювати основну гілку коду. Інші розробники Інші учасники надсилають свої зміни координатору у вигляді патч-файлів, які після перевірки можуть бути інтегровані до основного репозиторію. Таким чином, координатор контролює зміни та забезпечує їх відповідність загальній архітектурі та плану розвитку проекту.

Особливістю розробки проектів з відкритим вихідним кодом є те, що налагодження програми може вестися у вигляді паралельного процесу, до якого можуть долучатися численні користувачі. Знайшовши в програмі помилку, вони можуть надіслати виправлення координатору проекту. Це дозволяє ефективно

розподіляти ресурси, особливо залучаючи тих, хто не спеціалізується на проектуванні, але має змогу допомогти з відлагодженням [11].

2.3.7. ASD - Адаптивна методологія

Адаптивна методологія розробки ПЗ (Adaptive Software Development) – одна з сучасних гнучких методологій, розроблена як альтернатива традиційним процесно-орієнтованим підходам. Основна увага приділяється людському фактору, результатам розробки та мінімізації бюрократичних процедур, водночас посилюючи комунікацію між учасниками процесу.

ASD базується на принципі безперервної адаптації, завдяки якій виникає інший життєвий цикл проекту. Постійні зміни в проекті стають нормою.

У ASD звичайний статичний життєвий цикл "Планування - Проектування - Розробка" замінений на динамічний "Обмірковування - Взаємодія - Навчання".

Цей підхід спрямований на безперервне навчання. Він пов'язаний з постійними змінами, повторними оцінками і спробами передбачити, невідоме на поточний момент, майбутнє проекту і вимагає тісної взаємодій між розробниками, тестувальники та замовниками.

Методологія ASD базується на концептуальних засадах теорії складних адаптивних систем і орієнтована на застосування в умовах екстремальних проєктів, яким притаманні високий темп розробки, суттєва непередбачуваність перебігу подій та часті зміни вимог. Попри те що окремі проєкти не підпадають під категорію екстремальних, для переважної більшості інших випадків ASD є значно ефективнішим рішенням порівняно з будь-яким традиційним підходом до розробки програмного забезпечення [12].

2.3.8. Функціонально-орієнтована розробка (FDD)

У методології FDD (Feature Driven Development) центральним поняттям є функція або властивість системи (feature) — відносно самостійний елемент системи, реалізація якого, як правило, не перевищує двох тижнів.

Процес розробки за FDD охоплює п'ять етапів, два останні з яких відтворюються циклічно для кожної окремої функції:

1) розроблення загальної моделі системи;
2) формування переліку необхідних функцій;
3) планування роботи над кожною функцією;
4) проектування функції;
5) реалізація функції.

Учасники проєкту в рамках FDD розподіляються на дві ролі: «власників класів» і «головних програмістів». Головні програмісти залучають власників відповідних класів до реалізації конкретних функцій. На відміну від методології XP, де будь-який розробник має право вносити зміни до довільного класу, у FDD кожен учасник несе персональну відповідальність за чітко визначену зону компетенції.

Робота над проєктом організована у вигляді коротких ітерацій із регулярними зустрічами команди, і кожна ітерація передбачає реалізацію певного набору запланованих функцій.

2.3.9. Метод розробки динамічних систем

DSDM (Dynamic System Development Method або DSDM) – методологія, створена Дженніфером Сталетоном, що дозволяє розробляти програмне забезпечення у стислі строки при обмежених ресурсах. DSDM мінімізує рівні управління та комунікаційні бар’єри, щоб забезпечити більш ефективні процес розробки програмного забезпечення. Фіксуючи час розробки (зазвичай 6 місяців) і встановлюючи обмеження на використовувані ресурси, набагато простіше налагодити процес розробки, що відповідає вимогам замовників.

DSDM заснована на безперервному залученні користувача в ітераційний процес розробки, що є стійким до змін вимог і сумісним із формальними системами управління проектами

DSDM допомагає уникнути двох основних проблем, характерних для традиційних проектів. По-перше, специфікації часто відображають технічно здійсненні рішення з позиції розробників або замовників, замість того щоб відповідати реальним бізнес-потребам. Це означає, що програмісти або навіть самі замовники мислять з точки зору технічної реалізованості, а не з урахуванням фактичної цінності для бізнесу. По-друге, замовники змушені чекати на завершення повного циклу розробки, тестування та документування всієї системи, перш ніж отримають доступ до критично важливих функціональних компонентів програмного забезпечення.

DSDM непридатна якщо:

1) проект не передбачає ітеративної розробки додатку, де функціональність стає очевидною для замовника через інтерфейс користувача;

2) відсутня чітко визначена група кінцевих користувачів;

3) проект значною мірою залежить від складних внутрішніх обчислень.

Основні принципи DSDM:

1. Активне втручання користувача (в ідеалі, користувачі та розробники поділяють робочий простір, тому рішення можуть прийматися на місці).

2. Команда повинна мати можливість ухвалювати рішення без очікування підтвердження вищими рівнями (команда складається як з розробників, так і із замовників).

3. Вимоги замовника - перш за все.

4. Команда зосереджується на результаті проекту, а не на формальному виконанні визначений дій. Це дозволяє обирати технології, які найбільш ефективно сприяють створенню кінцевого продукту у встановлені терміни.

5. Розробка є ітеративною та керується зворотним зв’язком від користувачів. Ітеративність є характерною ознакою всього процесу розробки ПЗ, дозволяючи розробникам постійно вдосконалювати систему на основі отриманого відгуку.

Всі зміни є зворотними, що є ключовою характеристикою DSDM.

Тестування інтегровано в усі етапи життєвого циклу розробки, а не здійснюється безпосередньо перед випуском, коли внесення змін без значних втрат стає неможливим. Часові рамки мають бути обмеженими, з орієнтацією на швидкий випуск продукту. Оцінка програмного забезпечення повинна базуватися на необхідній функціональності, а не на обсязі коду. Аналіз ризиків має фокусуватися на функціональних вимогах до продукту, а не на процесі його побудови.

Контрольні питання

1. Як класифікуються методології розробки інформаційних систем?
2. Охарактеризуйте переваги адаптивних методологій у порівнянні з класичними підходами.
3. Що являє собою методологія швидкої розробки додатків (RAD)?
4. Поясніть, як SCRUM сприяє реалізації IoT-проєктів.
5. Назвіть методології, які найкраще підходять для швидкої розробки IoT-рішень.

ТЕМА 3. ІСНУЮЧІ ТЕХНОЛОГІЇ РОЗРОБКИ ІОТ ПРОМИСЛОВИХ СИСТЕМ

Ключовим змістом технології проектування є технологічні інструкції, що включають опис послідовності виконання технологічних операцій, умов, за яких реалізується та чи інша операція, а також детальну характеристику самих операцій.

3.1. Загальні вимоги до технологій проектування

Визначаються загальні вимоги, яким мають відповідати технології проєктування, розробки та супроводу інформаційних систем[7,15]:

1) підтримка всіх етапів життєвого циклу інформаційної системи;

2) забезпечення досягнення поставлених цілей розробки із заданим рівнем якості та у визначені терміни;

3) можливість декомпозиції великих проєктів на окремі підсистеми – поділ на компоненти, які розробляються невеликими групами з подальшою їх інтеграцією;

4) забезпечення ефективної роботи над окремими підсистемами малими командами (3–7 осіб);

5) скорочення часу до отримання працездатного продукту;

6) підтримка керування конфігурацією, версіонування проєкту та його складових, автоматизоване формування документації і узгодження її версій із версіями системи;

7) незалежність проєктних рішень від конкретних засобів реалізації – СУБД, операційних систем, мов програмування та середовищ розробки.

Технології розглядаються у двох вимірах: вертикальному (процесному) та горизонтальному (часовому).

Процес – це сукупність взаємопов’язаних дій, які перетворюють вхідні дані у результати. Він складається з дій, а кожна дія, у свою чергу, включає окремі

завдання. Вертикальний вимір описує структуру процесу та охоплює такі елементи, як процеси, дії, завдання, результати та виконавці.

Стадія – це впорядкована сукупність дій зі створення програмного продукту, обмежена у часі та спрямована на отримання конкретного результату відповідно до визначених вимог. Стадії поділяються на етапи, які зазвичай мають ітераційний характер. Іноді кілька стадій об’єднуються у більші часові відрізки – фази. Горизонтальний вимір відображає часову послідовність і динаміку процесу розробки, використовуючи такі поняття, як фази, стадії, етапи, ітерації та контрольні точки.

Особливості поєднання процесів і стадій, з урахуванням типу програмного забезпечення та специфіки команди розробників, формують відповідний технологічний підхід.

3.2. Технологічні підходи при розробці програмного забезпечення

Існують декілька різноманітних технологічних підходів.

Підходи зі слабкою формалізацією не використовують явних технологій і їх можна застосовувати тільки для дуже маленьких проектів, які, як правило, завершуються створенням демонстраційного прототипу. До підходів зі слабкою формалізацією відносяться так звані ранні технологічні підходи, наприклад підхід "кодування і виправлення".

Суворі (класичні, жорсткі, передбачувані) підходи застосовують для середніх, великомасштабних і надвеликих проектів із відносно чіткими вимогами до системи і більш-менш фіксованим обсягом робіт. Одна з основних вимог до таких проектів є максимально можлива передбачуваність процесу розробки. До цієї групи належать підходи, що наведені нижче.

3.2.1. Каскадні технологічні підходи

Каскадні підходи можуть реалізовуватися в таких варіантах:

1) Класичний каскадний підхід – перехід до наступного процесу здійснюється лише після завершення поточного; повернення до попередніх етапів не передбачені.

2) Каскадно-зворотний підхід – дозволені повернення до попередніх стадій і перегляд або уточнення раніше прийнятих рішень.

3) Каскадно-ітераційний підхід – передбачає послідовні ітерації кожного процесу до тих пір, поки не буде досягнуто бажаного результату.

4) Каскадний підхід з процесами, що перекриваються, передбачає наявність спеціалізованих команд, які дозволяють скоротити обсяг переданої документації. Наступний процес розпочинається до завершення поточного, при чому кілька процесів можуть виконуватися паралельно.

5) Каскадний підхід з підпроцесами дуже близький підходу з процесами, що перекриваються. Особливість в тому, що проект може бути розділений на підпроекти, які можуть розроблятися індивідуально.

6) Спіральна модель оперує поняттям прототипу — програми, що реалізує часткову функціональність майбутнього програмного продукту. Характерною рисою цієї моделі є ітеративний характер розробки, за якого кожен наступний прототип відрізняється розширеним набором функцій порівняно з попередньою версією.

3.2.2. Каркасні підходи

Раціональний уніфікований процес увібрав у себе краще з технологічних підходів каскадної групи. Він включає в себе наступні фази: початок (визначення цілей проекту), дослідження (розробка плану і архітектури проекту), побудова (поступове створення системи), впровадження (поставка системи кінцевим користувачам). Особливості - ітеративний контроль якості, можливість виявити

помилки на ранніх стадіях, перевага віддається моделям, а не паперовим документам, конфігурування, налаштування, масштабування.

Модель процесів Microsoft Solution Framework (MSF) є технологічним підходом, що ґрунтується на сукупності моделей, правил і специфікацій, які застосовуються під час створення та розповсюдження програмних продуктів і забезпечують ефективніше використання технологій для розв'язання бізнес-проблем. Вона дозволяє кількісно оцінити ступінь завершеності роботи над проектом і адаптувати методи управління відповідно до поточних потреб.

Формальні підходи передбачають наявність суворих вимог до процесу розробки програмного забезпечення. Наприклад, генетичні підходи вимагають формалізації, що пов'язана з походженням програмного продукту та дотриманням чіткої дисципліни його створення.

3.2.3. Генетичні підходи

До генетичний підходів належать такі різновиди:

- Синтезуюче програмування передбачає автоматичний синтез програмного забезпечення на основі його специфікації. Документ, формалізований мовою специфікацій, слугує базисом для подальшої реалізації.
- Складальне (розширюване) програмування – базується на концепції побудови програм шляхом повторного застосування існуючих програмних компонентів.
- Конкретизувальне програмування – передбачає створення спеціалізованих програм на основі універсальної, загальної програми.

3.2.4. Підходи на основі формальних перетворень

Технологія розробки програмного забезпечення, що базується на застосуванні формальних методів, включає такі етапи:

1) розробка функціональних і користувальницьких специфікацій;
2) планування розробки;

3) формальна верифікація;

4) статичне тестування.

Формальні генетичні підходи мають такі особливості:

1. Формальне синтезувальне програмування використовує математичну специфікацію як сукупність логічних формул;

2. Формальне складальне програмування використовує специфікацію у формі композиції вже відомих фрагментів;

3. Формальне конкретизувальне програмування застосовує змішані обчислення і конкретизацію на основі анотацій.

Гнучкі (адаптивні, легкі) підходи застосовують для невеликих або середніх проектів за умов невизначеності або динамічної зміни вимог до системи. Команда розробників повинна бути відповідальною та кваліфікованою, а замовники мають бути готові долучатися до процесу розробки. До цієї групи належать такі підходи, як ранні технологічні підходи швидкої розробки, адаптивні підходи та підходи дослідного програмування.

3.2.5. Ранні технологічні підходи швидкої розробки

До ранніх технологічних підходів швидкої розробки можна віднести:

- Еволюційне прототипування – перший прототип включає створення розвиненого користувальницького інтерфейсу.
- Ітеративна розробка – перший прототип вже повинен включати завершене ядро системи.
- Постстадійна розробка спрямована на усунення недоліку попередніх двох підходів, а саме – неможливості визначення термінів завершення проектів.

3.2.6. Адаптивні підходи

Екстремальне програмування (eXtreme Programming або XP) характеризується відмовою від ретельного попереднього проектування програмного забезпечення. Замість цього акцент робиться, з одного боку, на

постійній присутності представника замовника у команді, готового надавати відповіді на питання та оцінювати прототипи, а з іншого боку – на регулярному рефакторингу коду. Ретельно прокоментований код розглядається як основна проектна документація. Значна увага у методології приділяється тестуванню. Як правило, перед розробкою кожного нового методу спочатку пишеться тест, а потім розробляється відповідний код методу до моменту успішного проходження тесту. Ці тести об’єднуються у набори, які автоматично виконуються після кожної зміни коду [10].

Адаптивна розробка ґрунтується на трьох основних стадіях: обмірковування, співробітництво і навчання. Результати планування у межах даного підходу завжди не передбачувані. На відміну від традиційного планування, де відхилення розцінюються як помилки, у адаптивній розробці відхилення можуть сприяти прийняттю оптимальних рішень. Зобов'язання та плани розробників і замовників підлягають періодичному перегляду протягом усього циклу розробки.

Сімейство технологічних підходів Crystal передбачає застосування індивідуалізованого технологічного підходу для кожного конкретного проекту, залежно від таких факторів, як складність, терміновість та кількість залучених розробників.

3.2.7. Підходи дослідного програмування

Комп'ютерний дарвінізм заснований на принципі висхідної розробки, коли система будується навколо ключових компонентів і програм, які створюються на ранніх стадіях розробки, являє собою метод проб і помилок, заснований на інтенсивному тестуванні, причому на будь-якому етапі система повинна працювати.

Фрагментарне програмування полягає в тому, що спочатку створюється шаблон програм з працюючими шматочками (фрагментами). Далі виконується

поступове наближення до кінцевої мети. Застосовується в тому випадку, коли не можуть бути точно сформульовані вимоги до більшої частини завдання.

3.3. Державні та міжнародні стандарти в області розробки програмного забезпечення

3.3.1. Міжнародний стандарт ISO / IEC 12207: 1995-08-01

Перша редакція стандарту ISO 12207 була підготовлена у 1995 році спільним технічним комітетом ISO/IEC. Цей стандарт визначається як основоположний у галузі процесів життєвого циклу програмного забезпечення та розрахований на застосування у різних типах ПЗ і проєктах автоматизованих систем. Необхідність узгодженого використання стандартів для інформаційних систем і програмного забезпечення обґрунтовується положенням ISO 12207, відповідно до якого процеси життєвого циклу ПЗ мають бути узгоджені з процесами життєвого циклу автоматизованих систем.

Згідно з ISO 12207, система розглядається як інтегрована сукупність одного або кількох процесів, апаратних і програмних засобів, обладнання та персоналу, що спільно забезпечують задоволення визначених потреб.

Стандарт не регламентує жорсткої структури етапів або стадій життєвого циклу інформаційної системи, натомість визначаючи перелік процесів — зокрема придбання, постачання, розробки та інших.

Кожен процес у межах стандарту деталізується через окремі дії, а ті, своєю чергою, — через конкретні завдання. Принциповою особливістю ISO 12207 є відсутність жорстко заданої послідовності процесів: будь-який із них може ініціювати інший за відповідної потреби.

Особливості стандарту ISO 12207:

Стандарт вирізняється динамічним характером: порядок виконання процесів і завдань визначається гнучко, оскільки один процес може викликати інший або його окрему частину залежно від конкретних обставин.

Він також відзначається високим рівнем адаптивності: процеси, дії та завдання можуть змінюватися відповідно до специфіки конкретного проєкту. Це досягається шляхом виключення тих елементів, які не є актуальними для певної системи.

ISO 12207 принципово не містить описів конкретних методів реалізації, готових рішень чи шаблонів документації. Також не регламентуються назви, формати або точний зміст документів – ці аспекти визначаються сторонами, що застосовують стандарт.

Забезпечення якості реалізації процесів передбачає обов’язкову незалежність контролюючого персоналу від об’єктів перевірки. Контроль здійснюється вже на ранніх етапах розробки, зокрема під час аналізу вимог на відповідність потребам.

Після прийняття організацією рішення використовувати ISO 12207 у комерційній діяльності, вона зобов’язується дотримуватися мінімального набору процесів і завдань, необхідних для відповідності стандарту.

Стандарт містить обмежену кількість положень щодо проєктування баз даних, що є виправданим, оскільки різні системи можуть використовувати специфічні типи баз даних або взагалі обходитися без них.

Цінність ISO 12207 полягає у наявності узагальнених наборів завдань, характеристик якості та критеріїв оцінювання, які забезпечують комплексне охоплення різноманітних ситуацій у процесі розробки.

3.3.2. Стандарти комплексу ДСТУ 1934

Наприкінці 90-х років було ініційовано ДСТУ 1934 як інтегрований комплекс взаємопов'язаних міжгалузевих документів. Об'єктами стандартизації є автоматизовані системи різного призначення. До них входять всі складові, включаючи програмне забезпечення та бази даних.

Комплекс орієнтований на погодження між замовником і користувачем. Стандарт передбачає можливість розробки автоматизованої системи самим

замовником, подібно до ISO 12207. Наприклад, шляхом створення відповідного спеціалізованого підрозділу. Порівняно з ISO 12207 формулювання ДСТУ 1934 менш явно та симетрично відображають дії обох сторін. Розподіл відповідальності між сторонами, як правило, визначається на основі цього змісту Тому що ДСТУ 1934 переважно зосереджений на змісті проектної документації,.

Особливий практичний інтерес становить група документів розділу 0 «Загальні положення» та пункту 6 «Створення, функціонування та розвиток автоматизованої системи». Серед актуальних нині стандартів виділяють ДСТУ 34.601-90, що регламентує стадії створення автоматизованої системи, ДСТУ 34.602-89, який визначає вимоги до технічного завдання на створення автоматизованої системи, а також методичні вказівки РД 50-34.698-90, що встановлюють вимоги до змісту документів. Зазначені стандарти визначають стадії та етапи виконання робіт зі створення автоматизованої системи, однак не передбачають наскрізних процесів у явному вигляді.

Відповідно до ДСТУ 34 створення автоматизованої системи поділяється на такі стадії та етапи:

1. Формулювання вимог до автоматизованої системи. Обєднується в наступні стадії:

- обгрунтованість необхідності розробки автоматизованої системи;
- формулювання вимог споживача до проектованої системи;
- укладання звіту про виконання робіт
- заявка на розробку технічного замовлення.

2. Концептуалізація:

- дослідження параметрів;
- науковий пошук;
- безпосередня концептуалізація;
- макет, який задовольняє вимогам замовника;
- звітування про виконану роботу.

3. Формалізація технічного завдання на розробку автоматизованої комплексу.

4. Ескізування проекту:

- генезис рішень всієї системі та окремих складових;
- розробка пакету документів.

5. Проектування:

- апаратне вирішення структурної побудови;
- укладання документації;
- формалізація рішень на постачання складових для комплектування
- формалізація технічних вимог на розробку;
- постановка завдань завданьта проектування суміжних частин пристрою автоматизації.

6. Технічне документування:

- документування системи і її частин;
- оформлення та / або відлагодження програмного забезпечення.
- випробування:
- підготовчі роботи;
- навчання або тренування персоналу;
- комплектація автоматизованої системи запасними інструментами та приналежностями;
- збиральні роботи;
- пуск та налагодження;
- пілотні випробування;
- експлуатування;
- прийомка.

7. Сервіс та обслуговування:

- гарантійне супроводження;
- післягарантійне супроводження.

Опис змісту документів, що розробляються на кожному з етапів, наведено у сімействі стандартів ДСТУ 34.

Особливості стандарту ДСТУ 1934

Головною метою розроблення комплексу нормативних документів ДСТУ 34 є усунення суперечностей, що виникають у процесі інтеграції систем внаслідок неузгодженості нормативно-технічної документації. Комплекс стандартів ДСТУ 34 за своєю структурою є ближчим до конкретних методичних схем, ніж стандарти типу ISO 12207.

Адаптивність стандарту ДСТУ 34 забезпечується такими можливостями:

1) відмова від етапу ескізного проектування та об'єднання етапів розроблення технічного проєкту і розроблення робочої документації;

2) виключення окремих стадій розробки, а також консолідація більшості документів та їхніх розділів;

3) введення додаткових документів, розділів документів і робіт;

4) динамічне формування часткових технічних завдань, що забезпечує гнучке конструювання життєвого циклу автоматизованої системи.

Стадії та етапи, що виконуються організаціями-учасниками робіт зі створення автоматизованої системи, визначаються у договорах і технічному завданні, що узгоджується з підходом, передбаченим стандартом ISO 12207.

Документи ДСТУ 34 встановлюють єдину термінологію та класифікують роботи зі створення автоматизованої системи, а також документацію, що формується в процесі їх виконання. Застосування ДСТУ 34 спрощує інтеграцію різнорідних систем і сприяє підвищенню якості інтегрованих рішень.

Забезпечення якості відповідно до ДСТУ 34 визначається у технічному завданні на автоматизовану систему та здійснюється на всіх подальших етапах із різним ступенем незалежності експертизи. У загальній послідовності розробки відповідні експертизи проводяться на дещо пізніших стадіях порівняно з тим, що передбачено ISO 12207.

Щодо обов'язковості: повна обов'язковість застосування ДСТУ 34 відсутня — матеріали стандарту мають статус методичної підтримки, значною мірою орієнтованої на замовника. Зокрема, стандарт містить вимоги до змісту технічного завдання та порядку проведення випробувань розробленої системи.

Ключовим документом взаємодії між сторонами є технічне завдання (ТЗ) на створення автоматизованої системи. ТЗ виступає основним вихідним документом для розроблення та приймання системи й визначає найважливіші точки взаємодії між замовником і виконавцем.

Відповідно до ДСТУ, автоматизована система складається з програмно-технічних і програмно-методичних комплексів, а також окремих компонентів організаційного, технічного, програмного та інформаційного забезпечення.

Основною метою створення комплексу нормативних документів ДСТУ 1934 є усунення протиріч, що виникають при інтеграції систем через невідповідність нормативно-технічної документації. На відміну від стандартів типу ISO 12207, комплекс стандартів ДСТУ 34 більше відповідає конкретним методикам, ніж абстрактним процесним моделям.

Ступінь адаптивності ДСТУ 34 визначається такими можливостями:

1. відмова від етапу ескізного проєктування та об’єднання етапів розробки технічного проєкту і робочої документації;
2. пропуск окремих стадій розробки та об’єднання більшості документів і їх розділів;
3. введення додаткових документів, розділів чи робіт;
4. динамічне формування часткових технічних завдань, що забезпечує гнучке формування життєвого циклу автоматизованої системи.

Стадії та етапи, які виконують організації-учасники проєкту автоматизованої системи, визначаються договорами та технічним завданням і узгоджуються з підходом, передбаченим стандартом ISO 12207.

Документи ДСТУ 34 забезпечують уніфіковану термінологію, класифікують роботи зі створення автоматизованої системи та визначають типи

документації, що розробляються під час виконання цих робіт. Використання ДСТУ 1934 спрощує інтеграцію різних систем і підвищує якість інтегрованих рішень.

Забезпечення якості згідно з ДСТУ 34 визначається в технічному завданні на автоматизовану систему та реалізується на всіх етапах розробки з різним рівнем незалежності експертизи. При цьому перевірки здійснюються дещо пізніше у порівнянні з підходом ISO 12207.

Ступінь обов'язковості ДСТУ 34: повна обов'язковість відсутня, а матеріали стандарту є методичною підтримкою. Ця підтримка орієнтована на замовника та включає вимоги до змісту технічного завдання і проведення випробувань готової системи.

Ключовим документом для взаємодії сторін є технічне завдання (ТЗ) на створення автоматизованої системи. Воно слугує основним вихідним документом для розробки та приймання системи і визначає критичні точки взаємодії замовника та розробника.

Відповідно до ДСТУ, автоматизована система складається з програмно-технічних і програмно-методичних комплексів, а також окремих компонентів організаційного, технічного, програмного та інформаційного забезпечення.

3.3.3. Стандарти комплексу ЄСПД - ДСТУ 3974-2000 (ДСТУ 19)

ДСТУ 3974-2000 є комплексним нормативним документом, що визначає призначення, межі застосування, класифікаційну структуру та порядок позначення стандартів, які входять до складу Єдиної системи програмної документації (ЄСПД).

Єдина система програмної документації являє собою сукупність державних стандартів, що закріплюють узгоджені між собою правила щодо розроблення, оформлення та обсягу програм і супровідної програмної документації.

Стандарти ЄСПД формують вимоги, що регулюють процеси створення, підтримки, виготовлення та експлуатації програмних засобів. Їх дотримання забезпечує:

1. уніфікацію програмних продуктів, необхідну для обміну програмами між розробниками та повторного використання раніше створених рішень у нових проєктах;
2. зниження трудових витрат і підвищення результативності на всіх етапах життєвого циклу програмних виробів – від розроблення до експлуатації;
3. автоматизацію процесів формування та зберігання програмної документації.

Супровід програмного забезпечення охоплює моніторинг його функціонування, подальший розвиток і вдосконалення, а також внесення необхідних виправлень для усунення виявлених помилок.

Вимоги та положення стандартів ЄСПД поширюються на програми і програмну документацію для обчислювальних машин, комплексів і систем будь-якого призначення та галузі застосування.

До структури ЄСПД входять такі категорії стандартів:

- основні та організаційно-методичні стандарти;
- стандарти, що що регламентують форму і зміст програмних документів, які використовуються в процесі обробки даних;
- стандарти, спрямовані на забезпечення автоматизації розроблення програмної документації.

Формування організаційно-методичної документації, що визначає і регламентує діяльність організацій у сфері розроблення, супроводу та експлуатації програмного забезпечення, має здійснюватися відповідно до вимог стандартів ЄСПД.

Контрольні питання

1. Визначте основні вимоги до технологій проектування IoT.

2. Охарактеризуйте відмінності між каскадними та адаптивними підходами.

3. Поясніть роль формальних перетворень у розробці IoT.

4. Дайте визначення генетичних підходів до створення програмного забезпечення.

5. Назвіть стандарти, що регулюють процеси розробки IoT-технологій.

ТЕМА 4. АНАЛІЗ ОСОБЛИВОСТЕЙ ІНФОРМАЦІЙНИХ СЕРВІСІВ ЗБОРУ ТА ОБРОБКИ ДАНИХ У СЕРЕДОВИЩІ «SMART CITY» ТА ІНШИХ ПРОМИСЛОВИХ СИСТЕМАХ

Провідні компанії пропонують рішення, спрямовані на інтелектуалізацію міського середовища, концентруючись на визначеному спектрі галузей. Наприклад, IBM зосереджується на сервісах для громадян, бізнесу, транспорту, комунікацій, водопостачання та енергетики; ORACLE – на державному управлінні, освіті, електронній охороні здоров'я, безпеці, енергетиці, транспорті та утилізації відходів; CISCO акцентує увагу на взаємодії людей, речей і даних. Більшість зазначених рішень використовують багаторівневу архітектуру, що варіюється від трьох до шести рівнів. Кількість рівнів частково визначається процесом трансформації даних у комерційно цінну інформацію, яка надалі використовується для надання сервісів міським користувачам, а також для створення можливостей підприємствам та міським операторам, зацікавленим у розробці інноваційних та ефективних сервісів на основі міських даних та інформації.

Основні технічні труднощі щодо розумних міських рішень пов'язані з доступом до даних, агрегацій, обґрунтуванням, доступом і наданням послуг через Smart City (інтерфейси програмного застосунку) API. Фінальною метою є служба мешканцям міста більш розумним і більш ефективним способом, стимулювати їх участь у міських стратегіях. Тому зібрані і вироблені дані використовуються для спрощення створення розумних і ефективних сервісів, що використовують міські дані та інформацію.

Розумні сервіси повинні бути розроблені і повинні управлятися підприємствами і міськими операторами, а не муніципалітетом. З іншого боку, муніципалітет повинен забезпечити гнучкий доступ до даних і сервісів. Тобто надавати ефективний і зручний доступ до даних з їх семантикою, сервісною інформацією, з можливістю управління інструментальними панелями і

функціональною сумісністю з будь-якими іншими розумними системами управління, активними у місті. У сучасному світі муніципалітети / міста і державні установи публікують величезну суму відкритих даних. Ці дані можуть бути грубо агреговані для інтеграції за допомогою рішень, таких як CKAN, OpenDataSoft, ArcGIS та OpenData. У більшості випадків ці рішення для відкритих даних підходять для збору відкритих файлів даних і їх індексації на основі відповідних описових метаданих. Відкриті дані можуть бути завантажені шляхом надання файлів в різних форматах: CSV, XLS, XML, SHP тощо. У деяких випадках вони забезпечують доступ до ефективних наборів даних, які, за допомогою деяких інструментів інтеграції даних і візуалізації, забезпечують можливість складання графічних таблиць на основі значень, що містилися в наборі даних. В крайньому випадку вони також забезпечують доступ до таких наборів даних як Сполучені дані (Linked Data, LD), відкриті дані (Linked Open Data, LOD), кодуючи інформацію у вигляді RDF-трійок. Доступ до сховищ RDF для перегляду даних може бути виконаний за допомогою візуальних браузерів.

У більшості випадків ефективність системи надання послуг для розумного міста визначається доступністю приватних даних, керованих міськими операторами, що охоплюють певні домени: оператори мобільності, енергетичні провайдери, підприємства, що надають послуги (у сферах охорони здоров’я, водопостачання), оператори зв'язку, туристичні оператори, університети тощо. Зазначені суб’єкти є постачальниками первинних даних. Наприклад, у місті може функціонувати декілька операторів громадського транспорту з тисячами транспортних засобів (автобусів тощо), а також оператори зв’язку, що розгорнули у межах міста від десятків тисяч до мільйонів станцій зв'язку. Ці організації використовують різноманітні методи збору та надання доступу до даних, такі як публікація відкритих наборів даних та статистичної інформації. Крім цього, вони оприлюднюють дані в режимі реального часу для подальшого використання, включаючи потоки мобільності, дані про енергоспоживання, метеорологічні дані тощо.

Дані в режимі реального часу надаються міськими операторами через певні API у вигляді REST-викликів або веб-сервісів. API для надання даних агрегатору міста можуть відповідати багатокомпонентним стандартам. Водночас деякі периферійні типи зібраних даних не підтримуються жодними стандартами. У цифровій екосистемі E015 для опрацювання значної кількості міських API розроблено середовище для збору документації про наявні дані через сервіси, API та інтерфейси у вигляді веб-порталу. Однак цей підхід не розв'язує проблему відсутності даних або сумісності сервісів, оскільки кожному розробнику мобільного чи веб-додатку необхідно інтегрувати безліч наборів даних та забезпечити доступ до них за допомогою багатокомпонентних протоколів провайдерів. У багатьох випадках також виникає потреба в укладанні кількох угод, оскільки кожен наданий API може мати власну модель ліцензування. Отже, розробники можуть отримувати дані, які потребують подальшої агрегації для досягнення семантичної уніфікації.

Ефективне розгортання розумних сервісів для міських користувачів часто можливе лише за умови семантичної інтеграції різнорідних даних, зокрема відкритих, приватних і даних реального часу, що надходять від органів адміністрування та міських операторів. Це потребує впровадження процесів узгодження та прийняття єдиної моделі даних і онтологій, як, наприклад, у проекті Km4City.

Семантична агрегація даних із кількох доменів є неможливою без загальної онтології, оскільки дані збираються різними установами й компаніями з використанням різних форматів, цілей, географічних прив'язок, а також різних стандартів ідентифікації, що змінювались у часі. Унаслідок цього набори даних рідко є семантично сумісними між собою, адже вони були створені в різний час, різними системами та різними виконавцями. Крім того, дані можуть мати різні моделі ліцензування: деякі з них є відкритими, інші – приватними і належать міським операторам, які не зацікавлені у втраті контролю над ними через

публікацію в нерегульованому середовищі, або можуть встановлювати певні обмеження (наприклад, дозвіл використання лише в некомерційних цілях).

Значуща інформація (фактична, прогнозована чи виявлена) може надаватися операторам і мешканцям міста через персоналізовані сервіси, орієнтовані на профіль та роль користувача. Наприклад, щоб надати інформацію про об'єкти чи події, що знаходяться поблизу поточної GPS-позиції користувача, необхідна інтеграція географічних даних та сервісів. Водночас об’єднання сервісів геолокації з аналізом типових людських потоків дозволяє місту покращувати соціальні послуги та транспорт, формувати персоналізовані рекомендації і планувати міські зміни.

4.1. Аналіз і порівняння архітектур Smart City

Всі розумні міські рішення повинні впоратися з обсягом великих даних, різноманітністю і достовірністю. Відкриті статистичні дані не є основним джерелом інформації в місті. Більшість проблем, пов'язана з даними реального часу такими як транспортна і людська мобільність в місті, споживання енергії, охорона здоров'я та інтернет речей. Архітектура Smart City повинна бути здатна використовувати в своїх інтересах величезну кількість даних, які прибувають з кількох джерел, з різною швидкістю обробки і аналізу для обчислень інтегрованої і багатодоменної інформації, роблячи прогнози, виявляючи аномалії для деталізації і для створення пропозицій і рекомендацій міським користувачам і операторам .

В останні роки було запропоновано багато архітектурних рішень, спрямованих на забезпечення доступності, агрегованості, застосовності та придатності даних до використання. Очевидно, що хмарні й розподілені системи становлять фундаментальну основу рішень у сфері Big Data, які є критично важливими для функціонування розумного міста. У свою чергу, рішення на базі IoT слугують основою для збору даних від міських датчиків та пристроїв. Водночас міська інфраструктура є значно складнішою, і зосередження уваги

лише на окремих перелічених аспектах може створити обмеження, неприйнятні для міських операторів, що, у свою чергу, стримуватиме розвиток сервісів Smart City. Більшість архітектур Smart City реалізують програми та сервіси Smart City шляхом підготовки і забезпечення різних видів API. Основні відмінності між архітектурами полягають у вибраних стратегіях перетворення даних і сервісів (зокрема бізнес-даних), які лежать в основі функціонування систем розумного міста.

4.1.1. Інформаційний інтегратор

Інформаційний інтегратор (архітектура моделі представлена на Рис. 4.1) збирає інформацію про API, надану різними постачальниками даних і послуг (включаючи їхню автентифікацію та ліцензування), і формує загальний огляд для розробників та міських операторів з метою перегляду та вивчення способів доступу до API наданих сервісів. Постачальниками можуть бути міські оператори, такі як оператори мобільності, енергетичні компанії, постачальники послуг з водопостачання та поводження з відходами тощо. Вони можуть надавати як відкриті, так і приватні дані, у вигляді статичних або даних реального часу.

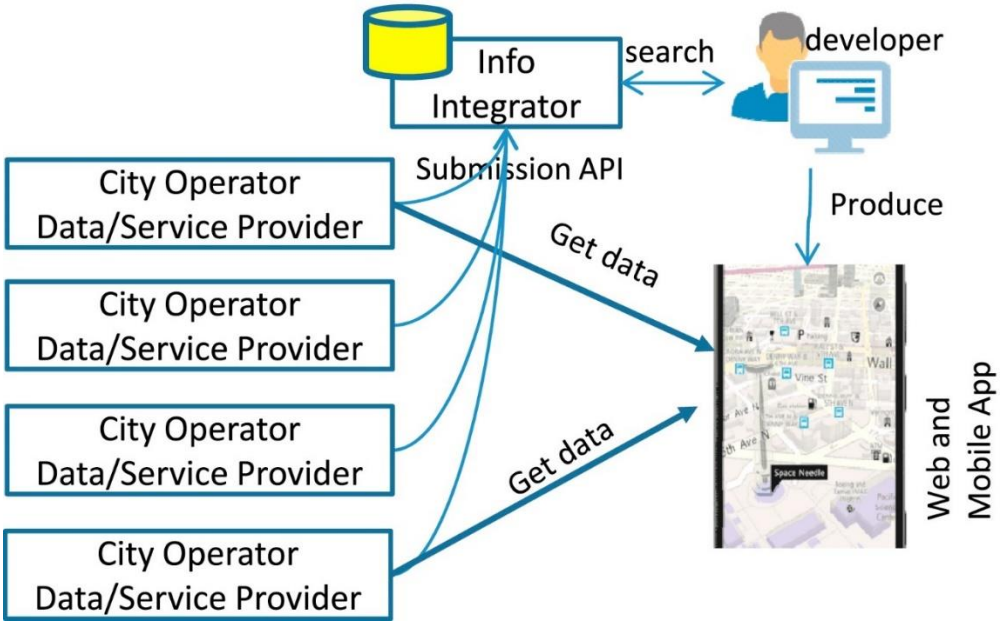

Рисунок 4.1 – Архітектура моделі інформаційного інтегратора

У моделі інформаційного інтегратора дані та послуги не інтегровані між собою: кожен набір API забезпечує доступ лише до даних або сервісів конкретного оператора. Отже, API та дані не є семантично сумісними, а завдання інтеграції лягає на розробників. Їм доводиться працювати з різними, неузгодженими API, протоколами автентифікації, моделями доступу тощо. Розробники додатків мають щоразу самостійно обирати необхідні дані, налаштовувати рішення, отримувати та інтегрувати їх у разі змін. Це також означає, що, ймовірно, доведеться укладати окремі договори та угоди з кожним із постачальників даних і послуг.

Інформаційний інтегратор повністю реалізує підхід «від даних до сервісів», оскільки дані й API кількох міських операторів є несумісними, а отримання ліцензій потребує окремих угод між кожним розробником і відповідними провайдерами даних, які надають доступ до своїх ресурсів через власні протоколи або API. Тому можливість прямого використання таких даних в інструментах візуалізації, бізнес-аналітики або контрольних панелях для моніторингу стану міста й аналізу взаємозв'язків між даними з різних джерел – досі залишається малореалізованою.

4.1.2. Агрегатор даних і метаданих

Агрегатор даних і метаданих (архітектура представлена на Рис. 4.2) збирає дані та інформацію про метадані (переважно відкриті дані) з метою їх індексації й агрегування в єдину модель. Такий підхід здебільшого орієнтований на збір даних, що отримуються за запитом від постачальників послуг або міських операторів (наприклад, через REST або Web Services-запити). Агреговані дані згодом надаються веб- та мобільним додаткам через автоматично згенеровані API, при цьому ігнорується семантика та можливі дублікати об'єктів, зібраних із різних джерел, які можуть описувати одні й ті самі елементи міської інфраструктури.

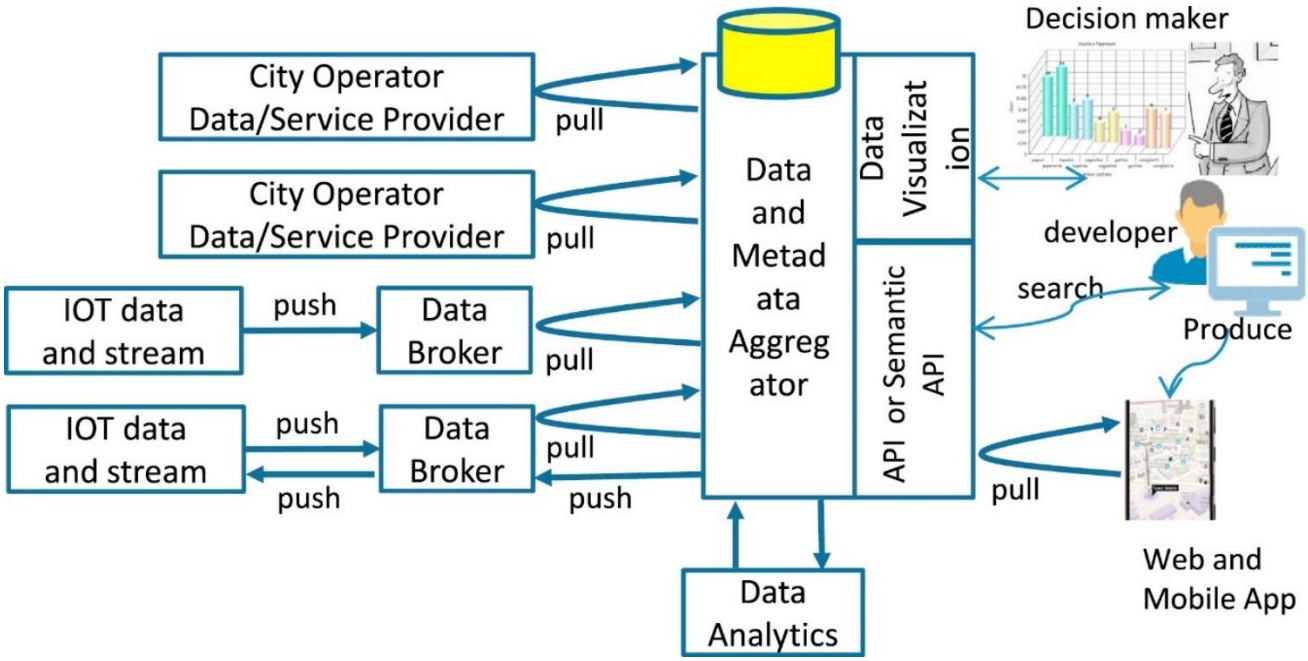

Рисунок 4.2 – Архітектура моделі агрегатора даних і метаданих

Проблеми відсутності вивірки виникають, коли набори даних містять велику кількість координат, дат, часових відміток, а також неточні ідентифікатори міських об'єктів. Дані не доповнюють одне одного, і якість джерел не забезпечує автоматичного відновлення відсутньої інформації. Інтеграція таких даних зазвичай базується лише на синтаксичній або лексичній подібності назв та значень. Для таких випадків часто використовуються таблиці з традиційними SQL-запитами. Для вирішення зазначених проблем застосовуються стандартні підходи до створення сховищ даних, зокрема ETL-процеси для встановлення зв'язків між даними та підвищення їх якості. Інколи візуалізація здійснюється через спеціалізовані інструменти, але семантичні розбіжності все одно залишаються.

Таким чином, надані API зазвичай не підтримують онтології, орієнтовані на предметну область. У деяких рішеннях семантичну модель додають лише як маркетинговий компонент, наприклад, реалізуючи SPARQL API. Проте така онтологія зазвичай є прямим відображенням таблиць і не забезпечує повноцінну дедуктивну підтримку. Усе це створює часові та просторові обмеження в роботі з даними.

До цієї категорії належать рішення на базі CKAN, ArcGIS OpenData, OpenDataSoft, SOCRATA (усі – на основі ArcGIS). ArcGIS-орієнтовані платформи пропонують більше можливостей для геопросторових запитів. Модель агрегатора даних і метаданих є базовим прикладом функціональної сумісності даних, яка може бути ефективною лише за умови використання переважно відкритих даних, без залучення приватних або даних у реальному часі. Дані реального часу можуть бути доступними для отримання за запитом або надаватися у вигляді потоку даних через рішення на основі IOT. Відповідно, відкриті дані повинні бути інтегровані з приватними даними та даними реального часу, як це спостерігається у випадках міських послуг мобільного зв'язку. Рішення повинні забезпечувати ефективне управління численними специфічними випадками, які вимагають окремого розгляду для інтеграції з іншими даними, оскільки це може значно знизити ефективність інтеграції. Дані, що надходять у вигляді потоку, можуть бути зібрані окремими брокерами даних, які, у свою чергу, можуть бути викликаними агрегатором за визначеними умовами.

Рішення, сумісні з моделлю агрегатора даних і метаданих, не забезпечують повне охоплення усіх підцілей міських платформ, оскільки дані не є повністю узгодженими. Такі рішенні було прийнято для реалізації стандарту Smart City, коли під управлінням перебуває більше ніж 15000 датчиків, розподілених по 1200 вузлах. Одним із прикладів є рішення, впроваджене в місті Сантандер, яке використовувало розгалужену мережу брокерів даних для збору інформації. Таким чином, дані агрегувалися через систему Big Data. Крім того, це рішення передбачає використання сервера CityModel, який надає послуги через API міської моделі, що дозволяє додаткам здійснювати прості та складні запити (включаючи статистичний аналіз), а також підписуватися на потоки даних (через push-механізми) для прямого доступу до інформаційних потоків. Однак переважно дані доступні лише періодично, що ускладнює роботу через відсутність семантичної підтримки при отриманні доступу.

Ще одним прикладом такої архітектури є ініціатива в місті Софія, орієнтована переважно на збирання IoT-даних. Проте використання комбінацій даних у прикладних сценаріях тут обмежене, адже семантична модель зосереджена на подіях, а не на створенні зв'язних потоків. Потокові підходи демонструють значні труднощі з обробкою багатодоменних зв'язків між даними. Крім того, IoT-рішення вимагають введення дозволів на виконання дій у системі Smart City (наприклад, для регулювання вуличного освітлення), що потребує доступу до брокерів даних.

У випадках великомасштабних потоків даних агрегатор повинен забезпечувати сервіси для обробки запитів API у реальному часі. Це особливо актуально для нетривіальних сценаріїв, коли дані з багатьох доменів, включно зі статичними, контекстними та потоковими, обробляються одночасно, сприяючи прийняттю обґрунтованих рішень. У таких ситуаціях застосування семантично інтегрованої моделі стає обов'язковим.

Рішенням зазначених проблем є підхід, заснований на моделі семантичного агрегатора і мірника.

4.1.3. Семантичний агрегатор і мірник

Модель семантичного агрегатора і мірника (архітектура представлена на Рис. 4.3) передбачає збирання даних і сервісів від міських операторів з метою їх агрегування та інтеграції в уніфіковану, семантично узгоджену багатодоменну модель на основі онтологій. Ключова роль онтології в процесі агрегації даних полягає у класифікації сенсорів та актуаторів IoT, які підключаються до брокера IoT через відповідні протоколи виявлення, зокрема реалізовані у рамках FIWARE. На відміну від моделі агрегатора даних і метаданих, семантичний агрегатор застосовує онтологію не лише до метаданих і таблиць, а й до міських об'єктів, предметних областей і взаємозв’язків між ними.

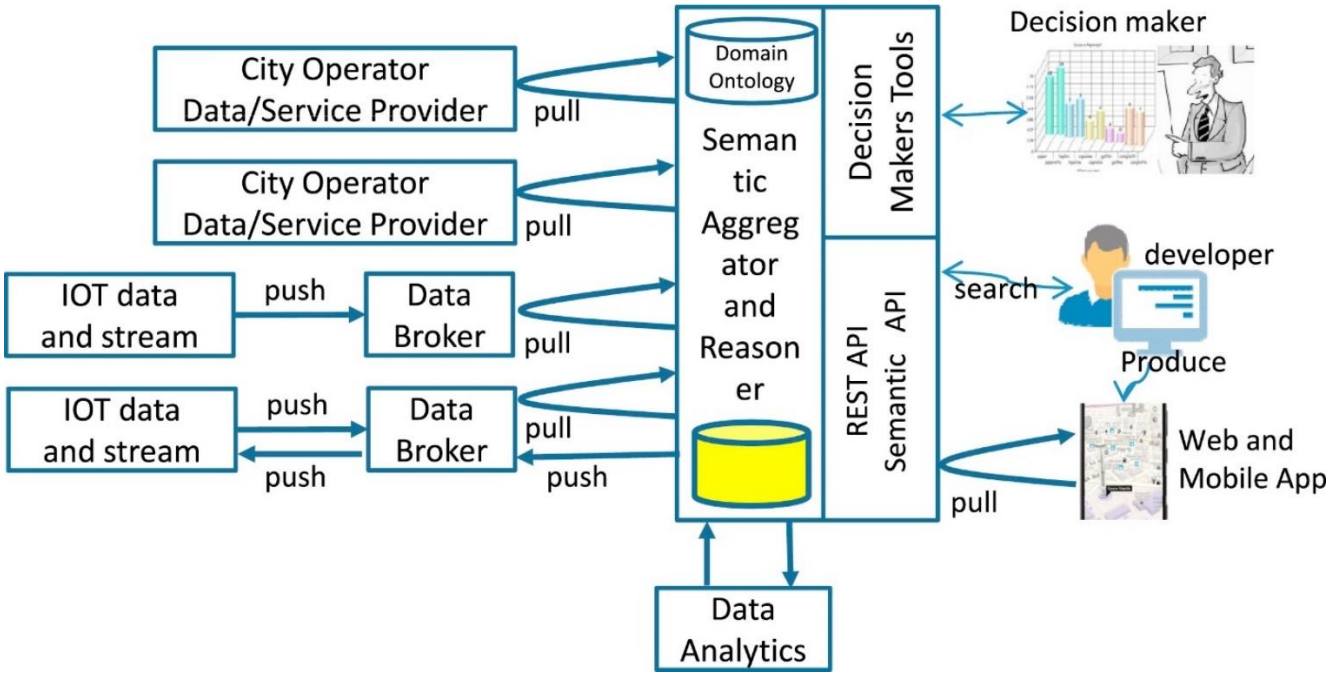

Рисунок 4.3 – Архітектура моделі семантичного агрегатора і мірника

Перевага такого підходу полягає у використанні онтології як основи для побудови бази знань, яка дозволяє повторно поєднувати дані з різних джерел, підвищуючи узгодженість та точність шляхом об'єднання інформації, що описує однакові сутності. Багатодоменна онтологія підтримує представлення зв’язку спеціалізації, агрегації, асоціації та подібності між класами, а також забезпечує дедуктивну обробку даних у форматі RDF. Створена база знань може бути використана для покращення якості даних, підтримки алгоритмів обґрунтування та надання послуг у межах кількох доменів (наприклад, прогнозування, раннє попередження), а також для реалізації просторово-часового аналізу.

Покращена експлуатація полягає в використанні онтології як основи для створення бази знань, щоб повторно поєднати дані для створення когерентної моделі, таким чином зменшуючи помилки, інтегруючи дані, що представляють ті ж поняття і прибувають з різних структур, операторів і джерел.

Цей підхід також забезпечує можливість доступу до узгодженого представлення даних через API, що є особливо важливим для осіб, відповідальних за ухвалення рішень. Завдяки наповненню онтологій даними та

реалізації механізмів логічного виведення, створена база знань може ефективно використовуватись для розробки інтелектуальних сервісів – зокрема, контекстної маршрутизації, персоналізованих рекомендацій, мультимодальних транспортних послуг і цифрових асистентів.

Прикладами реалізації цієї моделі є CitySDK (частково реалізує зазначені функції, розроблений у межах проекту ЄС і охоплює великі міста, пропонуючи REST API відповідно до принципів OASC), а також Km4City, який використовується у рамках проектів Sii-Mobility, RESOLUTE H2020 та REPLICATE H2020.

Модель семантичного агрегатора і мірника суттєво відрізняється від моделі агрегатора даних і метаданих наявністю комплексної онтології, створеної експертами, замість спрощених онтологій, згенерованих із таблиць даних. Крім того, вона включає інструменти для виявлення потреб у даних і забезпечує контроль якості, як-от у рішеннях DataLift і Km4City, де застосовуються словники, алгоритми та спеціалізовані мови, наприклад SILK. Варто зазначити, що модель семантичного агрегатора може бути побудована поверх рішення, реалізованого за моделлю агрегатора метаданих, шляхом використання її API для заповнення семантичної моделі.

Модель семантичного агрегатора і мірника краще відповідає вимогам досягнення підцілей міських платформ, пов’язаних з узгодженістю даних та побудовою інтелектуальних сервісів. Проте, користувацький досвід і розширені сервіси реалізовані лише в обмеженій кількості таких рішень. Графова база даних, яка зберігає великий обсяг даних щороку, створює передумови для сценаріїв обробки Big Data відповідно до характеристик обсягу, швидкості, достовірності та різноманітності. Ефективна семантична інтеграція дозволяє реалізовувати інтелектуальні системи підтримки прийняття рішень, використовуючи запити на декількох доменах, принципи ймовірнісного обґрунтування, а також алгоритми персоналізованої маршрутизації і цифрових

помічників. Водночас у деяких випадках застосування графових баз даних може бути недостатньо ефективним з точки зору продуктивності.

4.1.4. Порівняння архітектур

Таблиця 4.1 підсумовує проведений порівняльний аналіз існуючих архітектур і API-рішень для концепції Smart City. Основним критерієм оцінювання є здатність таких рішень спрощувати міські послуги та трансформувати дані служб у корисну інформацію для додатків, орієнтованих на міських користувачів та осіб, відповідальних за ухвалення рішень.

В аналізі розглянуто ключові відмінності, що сприймаються цільовими користувачами, включаючи кількість інтелектуальних та міждоменних сервісів, які підтримуються у рамках моделі семантичного агрегатору. Ця модель репрезентує нове покоління підходів у сфері розумного міста та становить перспективну тему для подальших досліджень. Рішення на її основі мають потенціал охопити ключові аспекти міських платформ, хоча рівень ефективного покриття залежить від якості інструментальної підтримки. Серед найважливіших інструментів вирізняються системи аналітики даних, API розумного міста та модулі підтримки ухвалення рішень – кожен з них відкриває окремі можливості для побудови інтелектуальної міської інфраструктури.

У більшості рішень концепції Smart City дані трансформуються у сервіси, призначені для міських користувачів та операторів, із використанням засобів аналітики. Такі сервіси, як правило, базуються на поєднанні відкритих і приватних джерел, включаючи як статичні, так і динамічні дані, що надходять від міської адміністрації та приватних постачальників. Функціональні можливості API розумного міста залежать від обраних архітектурних рішень і можуть по-різному реалізовувати процес трансформації даних у сервіси. Залежно від підходу, API може обмежувати або, навпаки, сприяти використанню узгоджених і агрегованих даних, які можуть бути повторно використані та обґрунтовані за допомогою спеціалізованих алгоритмів.

Таблиця 4.1

Порівняння розглянутих архітектур

Дія, процес	**Інформаційний агрегатор**	**Агрегатор даних і метаданих**	**Семантичний агрегатор і мірник**
Звернення до відкритих даних	Так	Так	Так
Звернення до приватних даних	Так	Так	Так
Отримання даних у режимі реального часу	Так	Так	Так
Вирішення взаємодії служб	(Так)	Так	Так
Збір даних	Ні	Так	Так
Пошук даних	Ні	Так	Так
Пошук метаданих	Так	Так	Так
Просторове логічне виведення	Ні	(Так)	Так
Часове логічне виведення	Ні	(Так)	Так
Інтеграція аутентифікованого доступу до даних	Тільки метадані	Так	Так
Надання синтаксичних взаємосумісних даних	Ні	Так	Так
Надання семантичних взаємосумісних даних	Ні	Ні	Так
API, яке адаптується до змін у моделі даних	Ні	Ні	Так
Надання доступу через REST API	(Так)	Так	Так
Надання доступу через SPARQL API	Ні	(Так)	Так
Надання підтримки вихідних даних	Ні	Ні	Так
Візуалізації даних для аналітики	Ні	(Так)	Так

(Так) означає «так» з обмеженими можливостями через обмежену сумісність даних між міськими організаціями: час, простір, кілька доменів (семантично взаємосумісні), структури, послуги та відносини.

У таблиці проведено порівняння кількох сучасних архітектурних рішень для агрегації даних у розумних містах та забезпечення Smart City API, з акцентом на покриття вимог та гнучкість. Оцінено вплив інтеграції семантичних обчислень у архітектуру розумного міста для надання інтелектуальних послуг. Аналіз підкреслює переваги використання семантично сумісних агрегованих даних для виконання просторового, тимчасового та концептуального обґрунтування через API розумного міста [19].

4.2. Принципи роботи Smart City та ключові об’єкти для моніторингу

Відповідно до аналітичних матеріалів консалтингової компанії Navigant Research, сучасна концепція Smart City охоплює такі ключові напрямки:

• Smart Energy – сукупність рішень у сфері енергопостачання та енергозбереження, що включає програми регулювання споживчого попиту, підвищення енергоефективності та інтеграції відновлюваних джерел енергії до загальної енергосистеми;

• Smart Water – комплексне управління водними ресурсами міста, що передбачає модернізацію водогосподарської інфраструктури, диференційований моніторинг водоспоживання за секторами, а також функціонування систем екологічного контролю та протиповеневого захисту;

• Smart Buildings – зведення або дообладнання будівель, у яких усі інженерні та інформаційні підсистеми об'єднані в єдину систему управління будівлею (BMS – building management system). Така інтеграція дозволяє, зокрема, гнучко регулювати режим опалення залежно від фактичної кількості людей у приміщеннях, керувати продуктивністю вентиляційних установок і якістю повітря, а також автоматично активувати режим енергозбереження за відсутності людей;

• Smart Transportation – розбудова інтелектуальних транспортних і логістичних систем, що забезпечують моніторинг і регулювання дорожнього руху, контроль справляння дорожніх зборів, оперативне реагування на надзвичайні події та автоматизоване керування світлофорними об'єктами. До цього ж напрямку зазвичай відносять системи інтелектуального паркування та інформаційного оповіщення на зупинках громадського транспорту;

• Smart Government – використання інформаційних технологій для надання державних і муніципальних послуг широкому колу громадян, що дає змогу оптимізувати діяльність різних відомств та підвищити ефективність публічного управління [20].

Серед об’єктів і складових, які потребують контролю та управління можна назвати такі:

- Інтелектуальна енергетика (вітрова чи біоенергетика).
- Збір та переробка побутових та виробничих відходів.
- Інтелектуальна медицина.
- Освіта.
- Інтелектуальне середовище Smart City.
- Інтелектуальні будівлі.
- Розумний дім.
- Інтелектуальна парковка.
- Публічна безпека.
- Аналіз настрою людини.
- Інтелектуальна система виявлення шахрайства з даними.
- Моніторинг транспортного трафіку (потоків).
- Інтелектуальна система освітлення вулиць.
- Середовище IoT (класи речей, інтерфейси, протоколи, концентратори і сервери даних, рівні інтеграції і комунікацій, тощо).
- Відкриті дані та Великі дані (Big Data).

- Служба контролю забруднення повітря.
- Служба контролю електромагнітного випромінювання.
- Служба контролю якості води.
- Служба контролю виявлення витоку води та газу.
- Мережа зарядки електромобілів.
- Система сервісів «Розумні покупки».
- Інтелектуальні сервіси (доставка їжі, ліків, пошти, інше).
- Середовище роботизованих систем та пристроїв.
- Кібербезпека у Розумному місті.
- Розподілена мережа системи обробки подій у Smart City.
- Загальноміська інформаційна система SEA Smart City.
- «Розумний пил» (мережа мініатюрних сенсорів, які збирають і передають дані про навколишнє середовище бездротовими технологіями).

4.3. Центр моніторингу та управління

«Розумне місто» являє собою міське середовище, в якому традиційна інфраструктура функціонує з вищою ефективністю завдяки впровадженню інформаційно-комунікаційних технологій. Застосування таких технологій дає змогу задовольняти незмінний обсяг суспільних потреб при зниженні споживання енергетичних ресурсів та скороченні обсягів викидів парникових газів. Практично це передбачає розбудову інтелектуальних транспортних мереж, модернізацію систем водопостачання та поводження з відходами, а також підвищення енергетичної ефективності систем опалення й охолодження будівель.

При цьому всі зазначені системи мають бути взаємно інтегровані та діяти як єдиний узгоджений організм. Поряд із інформаційно-комунікаційними технологіями важливу роль відіграє людський і соціальний капітал, що

забезпечує підвищення рівня безпеки у громадських просторах та формування комфортного середовища для мешканців.

Отже, концепція «розумного міста» орієнтована на створення відчутних переваг для повсякденного життя громадян та ділової діяльності в контексті принципів сталого розвитку [20,21]. Smart City ґрунтується на цілісній концепції інтелектуальної інтеграції інформаційних і комунікаційних технологій з метою моніторингу та управління міською інфраструктурою.

Кінцевою метою таких заходів є підвищення якості життя населення через зростання рівня комфорту та безпеки, вдосконалення якості й ефективності надання послуг у різних секторах, а також скорочення витрат на ресурси з інтенсивним характером споживання.

Інфраструктура Smart City охоплює широкий спектр різноманітних рішень, що реалізуються шляхом впровадження інтелектуальних технологій. Серед них альтернативні підходи до енергопостачання та водозабезпечення, технології опріснення морської води, сучасні системи сортування й переробки твердих побутових відходів, запровадження немоторизованих транспортних засобів, розгортання розгалуженої мережі відеоспостереження та відеоаналітики, а також моніторинг якості атмосферного повітря.

Концепція «розумного міста» включає шість ключових складових, п'ять із яких полягають у такому:

− «розумна економіка» (smart economy) – охоплює електронний бізнес та електронну комерцію, зростання продуктивності праці, інноваційне виробництво товарів та технологічно орієнтовану доставку послуг;

− «розумне переміщення» (smart mobility) – транспортні та логістичні рішення на базі інформаційно-комунікаційних технологій, що забезпечують можливість дістатися будь-якої точки міста одним або двома видами транспорту;

− «розумні люди» (smart people) – формування цифрових компетентностей, підвищення освітнього рівня населення, розвиток професійних навичок, стимулювання творчого мислення та інноваційної активності;

− «розумне життя» (smart living) – впровадження моделей поведінки та споживання, що спираються на інформаційно-комунікаційні технології, покращення стану здоров'я громадян та культурний розвиток суспільства;

− «розумне врядування» (smart governance) – інтерактивне місцеве самоврядування, що забезпечує ефективне та всебічне управління містом [16].

Шостою складовою концепції виступає «розумне довкілля» (smart environment), нерозривно пов'язане з енергетичною сферою. Головний акцент тут робиться на впровадженні принципів енергоефективності та скороченні емісії парникових газів. У межах цього напряму передбачається формування «розумної енергетики» шляхом розбудови замкнених енергетичних мереж, систем контролю й моніторингу рівня забруднення довкілля, реконструкції та будівництва енергоефективних будівель, а також підвищення коефіцієнта корисної дії процесів когенерації.

У посиланні [16] обговорюються умови сталого розвитку Smart City та надано приклад структури міського диспетчерського Центру, зображеного на Рис. 4.4.

Рисунок 4.4 – Варіант структури міського диспетчерського Центру моніторингу та управління Smart City

Типова система повинна включати такі особливості, які можуть змінюватися в залежності від зони застосування:

1) Система має виконувати наступні функції:
   - збір статистичних даних;
   - оновлення знань;
   - управління агентом;
   - досягання цілей;
   - використання метазнань;
   - адаптація поведінки агента до змінюваних умов;
   - взаємодія з іншими агентами;
2) Сенсори системи.

Між розробкою мультиагентних систем (МАС) та створенням оболонок експертних систем із наповненням їх експертними знаннями існує певна паралель. Як і оболонку експертної системи, МАС здатні розробляти лише міждисциплінарні команди фахівців.

У зв'язку з цим доцільно розглянути основні сенсорні компоненти, з якими стикаються розробники МАС. Обізнаність із цими сенсорами, а також із доступними методами та технологіями, слугуватиме практичною основою для тих, хто створюватиме власні мультиагентні моделі.

До ключових сенсорних компонентів сучасних автоматизованих систем моніторингу належать:

– вимірювачі параметрів довкілля – температури, вмісту солей у воді, інтенсивності сонячного випромінювання, іонного складу, концентрації важких металів у водному середовищі, рівнів основних забруднювачів атмосфери та водойм тощо;

– вимірювачі біологічних показників – темпів приросту деревини, проективного покриття рослинного покриву, вмісту гумусу в ґрунті та інші;

– засоби автономного електроживлення на базі сучасних акумуляторних батарей або фотовольтаїчних панелей;

– малогабаритні приймально-передавальні радіосистеми ближньої дії з радіусом дії до 10–15 км;

– компактні радіостанції для передачі даних на відстані в сотні та тисячі кілометрів;

– системи супутникового зв'язку, як правило інтегровані із засобами глобального позиціонування (зокрема GPS);

– сучасні обчислювальні засоби, включно з мобільними пристроями;

– спеціалізоване програмне забезпечення.

Глобальна система спостережень охоплює дві підсистеми:

– супутникову;

– наземну.

Синоптична мережа Всесвітньої метеорологічної організації включає три типи станцій спостереження:

– наземних станцій;

– аерологічних станцій;

– морські – розташовані в акваторіях Світового океану.

Дані зі стаціонарних пунктів спостереження передаються до регіональних центрів, а звідти – до центрів збору та аналізу інформації. У світі функціонує близько дев'яти таких центрів, між якими налагоджено постійний інформаційний обмін.

Після визначення об'єктів і параметрів моніторингу необхідно встановити кількість і розміщення пунктів спостереження, а також режим їх роботи. По можливості слід урахувати всю сукупність факторів, що можуть впливати на якість отримуваних даних [21].

Канали передачі даних та протоколи:

Зібрані відомості вносяться до відповідних полів атрибутивних таблиць і можуть використовуватися для подальшої автоматизованої обробки. Серед основних підходів до формування таких таблиць у геоінформаційних системах виділяють [4,14]:

– пошук і систематизацію вже наявних масивів даних;

– цілеспрямоване накопичення даних у ході самостійних досліджень – у цьому випадку формат їх подання слід узгоджувати заздалегідь, особливо для характеристик із недостатньо формалізованими описами;

– автоматизований збір даних із застосуванням серверних механізмів – такий спосіб потребує розробки спеціалізованого програмного забезпечення для перетворення даних у формат, сумісний із геоінформаційними системами.

Вибір формату подання даних набуває особливого значення в системах, що об'єднують різнорідні відомості з кількох джерел. У перших двох випадках раціональним є використання стандартних додатків пакету MS Office: Access – для ведення персональних баз даних, Excel – для роботи з електронними таблицями. Дані в цих форматах легко інтегруються до атрибутивних таблиць відповідних шарів геоінформаційних систем, зокрема можуть бути розпізнані та використані в середовищі ArcGIS.

Найвищу пропускну здатність при безперервному відборі даних із прийнятним рівнем надійності забезпечують технології автоматизованої реєстрації. Зокрема, щогодинно оновлюваний супутниковий знімок Європи доступний у мережі Інтернет. Його отримання автоматизовано за допомогою програми, що запускається системним планувальником cron; на підставі цих знімків відстежується динаміка повітряних мас – зокрема для аналізу процесів перенесення забруднювачів і прогнозування міграційних маршрутів птахів. В автоматичному режимі також фіксуються результати вимірювань характеристик ґрунтового покриву.

Дані у текстовому форматі щогодини передаються на виділений FTP-сервер засобами протоколу GPRS (рис. 4.5) з використанням послуг оператора мобільного зв'язку.

Впровадження такої системи моніторингу потребує формування відповідної організаційної структури, забезпеченої просторовими інформаційними ресурсами, програмними інструментами та кваліфікованим

персоналом. Отримані результати засвідчують наявність низки специфічних рис, притаманних аерокосмічному моніторингу як інструменту оцінювання та прогнозування стану навколишнього середовища.

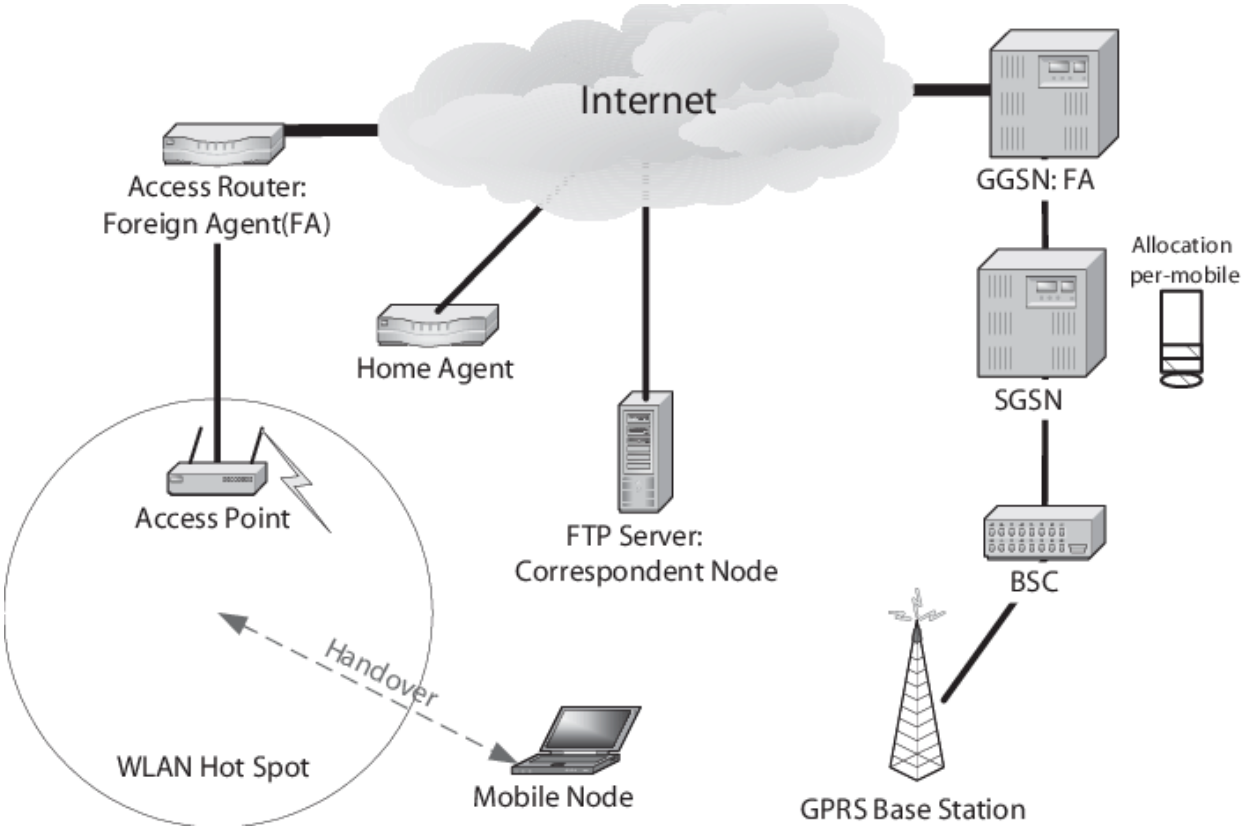

Рисунок 4.5 – Принцип застосування GPRS

Ці особливості здебільшого визначаються потребою у відображенні різнорідних екологічних показників, для кожного з яких застосовуються специфічні методи реєстрації, відбору, класифікації та представлення даних. Топографічну основу слід формувати на базі кількох взаємодоповнювальних джерел – топографічних карт, космічних знімків, планів земельних ділянок та матеріалів польових досліджень, які у сукупності дають повнішу картину об'єктів екологічного моніторингу.

Для прив'язки об'єктів до місцевості за допомогою автономного GPS-приймача виконуються вимірювання координат характерних орієнтирів – перехресть доріг, мостів тощо, які впевнено ідентифікуються на растрових

зображеннях. Такий підхід дозволяє максимально точно відтворити особливості досліджуваної території, а також виявити розбіжності в наявних даних.

Суттєвим інформаційним ресурсом для оцінки стану екосистем є матеріали дистанційного зондування Землі. Результати тематичної класифікації космічних знімків, виконаної у програмних середовищах ENVI та ERDAS IMAGINE, використовуються для уточнення геометричних параметрів об'єктів екологічного моніторингу.

Невід'ємним елементом системи є цифрові картографічні матеріали, доповнені атрибутивними даними з таксаційними описами лісових кварталів. Структурування цих описів і їх просторова прив'язка до об'єктів геоінформаційних систем відкривають можливості для вирішення широкого кола практичних завдань господарського та екологічного характеру [26].

Контрольні питання

1. Перерахуйте ключові архітектури, що використовуються у середовищі «Smart City».

2. Охарактеризуйте функціонування інформаційних інтеграторів у міських IoT-системах.

3. Поясніть функції семантичного агрегатора даних.

4. Назвіть об'єкти, критичні для моніторингу у «Smart City».

5. Опишіть способи інтеграції інформаційних сервісів у промислових системах.

ТЕМА 5. ТЕХНОЛОГІЇ IOT ТА BIG DATA В РОБОТИЗОВАНІЙ ІНТЕРАКТИВНІЙ ІНФРАСТРУКТУРІ «РОЗУМНОГО МІСТА»

Люди продовжують стікатися в міста з кількох причин, таких як можливості працевлаштування, стиль життя і багатьох інших.

Останні дані Бюро перепису населення США показали, що в 2024 році всі, крім одного з 20 найбільших міст США, мали приріст населення. Подібна тенденція розповсюджується і на весь світ. Так у середньому по світу 60% людей проживають у містах (більш детально на Рис. 5.1), та при цьому міста сьогодні створюють до 70% валового світового продукту, тобто, іншими словами, до 70% світової економіки знаходиться у містах. Ефективно управляти таким величезним господарством без використання високотехнологічних інструментів навряд чи можливо.

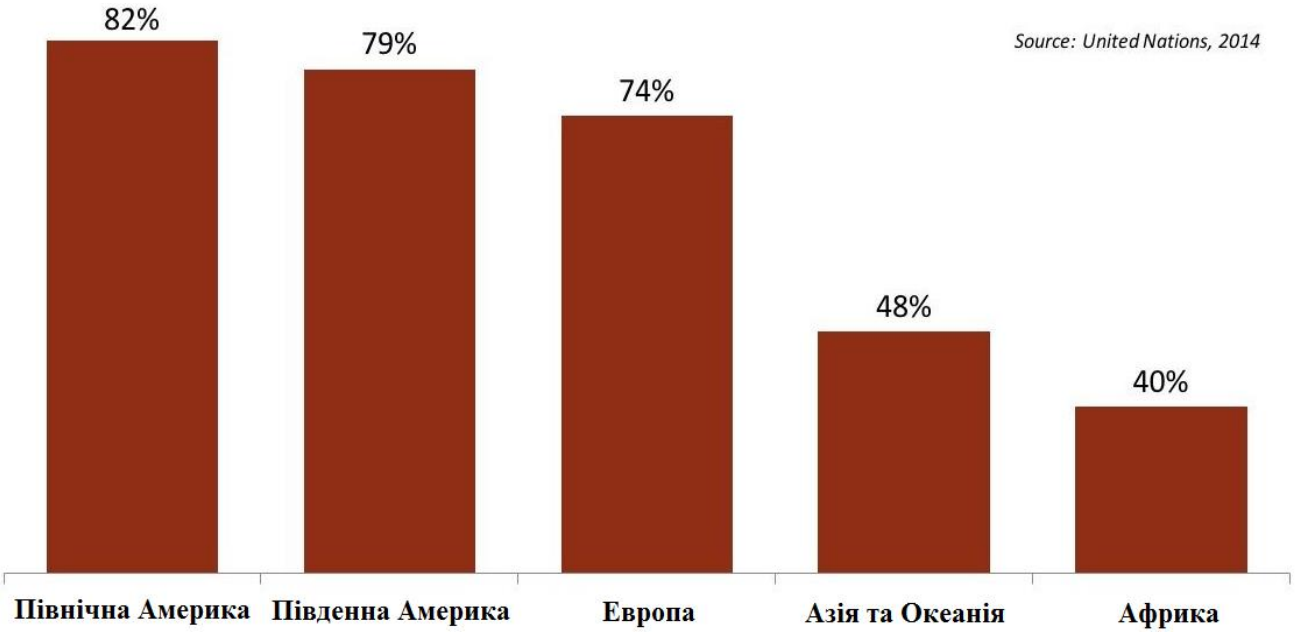

Рисунок 5.1 – Відсоток населення, що проживає у містах в 2024 році

Якщо урбанізація продовжиться, містам необхідно буде стати більш ефективними, щоб відповідати зростаючому населенню. Таким чином, Smart City почне ставати нормою у великих мегаполісах світу.

5.1. Визначення «Smart City»

Якщо звернутися до «офіційного» визначення, то Вікіпедія надає таке формулювання: інфокомунікаційні технології (ІКТ) дозволяють міській владі безпосередньо взаємодіяти з громадою та міською інфраструктурою, а також стежити за розвитком міста і шукати способи поліпшення якості життя. Завдяки використанню датчиків, інтегрованих у режимі реального часу, та накопиченню даних від мешканців і пристроїв, які обробляються та аналізуються, зібрана інформація стає ключем до вирішення проблем неефективності. Потенціал розумних міст практично безмежний, і у найближчі роки зростання таких міст повинно прискоритися. Для розвитку цієї сфери активно використовуються та будуть використовуватися сервіси технологій IoT та Big Data, які не тільки змінюватимуться самі, а й впливатимуть на розвиток суспільства.

5.2. Призначення та компоненти «Smart City»

Що являє собою «розумне місто» у своїй сутності? Це універсальна інтегрована система інформаційної підтримки, що виконує дві ключові функції:

- забезпечує органи міської виконавчої влади всіх рівнів необхідною інформацією;
- надає інструменти зворотного зв'язку, за допомогою яких виконавча влада може здійснювати цілеспрямований вплив на певні сфери міського життя в межах своєї компетенції.

У чому полягає практична цінність такої системи?

По-перше, реалізація цієї концепції однозначно підвищує якість життя населення регіону, або, як зараз ще називають life quality experience.

По-друге, впровадження концепції «розумного міста» забезпечує скорочення витрат на утримання міської інфраструктури завдяки автоматизації рутинних управлінських процесів у сфері міського господарства та запровадженню інструментів об'єктивного контролю за діяльністю комунальних служб. В умовах нинішніх економічних обмежень, що супроводжуються

дефіцитом фінансових ресурсів, зокрема у житлово-комунальній галузі, оптимізація поточних операційних процесів залишається одним із найбільш затребуваних і практично доцільних рішень.

Розглянемо, що визначає і що приховано під етикеткою «Розумне місто»: які компоненти та технології складають цю концепцію (див. Рис. 5.2.).

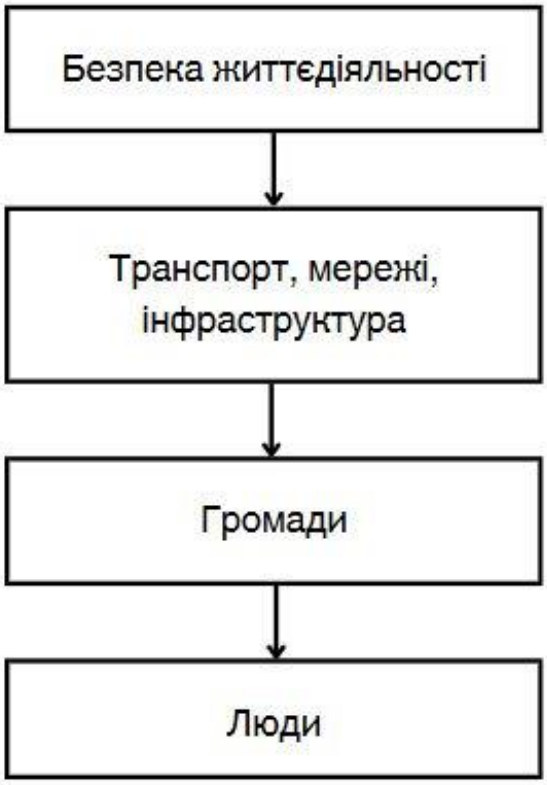

Рисунок 5.2 – Концептуальна структура компонентів «Smart City»

Ключовою сполучною ланкою «розумного міста» є деякий операційний центр, який акумулює у собі інформацію від нижчих систем і є високорівневим джерелом керуючих впливів.

Наступний рівень за центральною консоллю – рівень конкретних областей міського господарства, у кожній з яких стоять свої завдання і свої показники ефективності зі своєю специфікою. І в зв'язку з цим кожна така область управляється окремим додатком. Наприклад, додаток по «розумному управлінню міським освітленням», який дозволяє задавати з центральної консолі

загальну політику управління світлом у місті і оптимізувати витрати на електрику і операційну підтримку освітлення. Або система «розумного управління житлово-комунальним господарством», яка стежить за станом різних ділянок ЖКГ і може ефективно реагувати на збої в окремих своїх частинах, а також прогнозувати потенційні проблеми на основі аналізу зібраних з датчиків даних (наприклад, десь на магістралі спостерігається падіння тиску: найімовірніше, на якийсь з ділянок протікання, і система управління покликана допомогти локалізувати і усунути пошкодження, поки це не привело до виникнення аварії). Усі перелічені автономні підсистеми – «розумне освітлення», «розумне ЖКГ», «розумне управління трафіком», «розумні будинки», «розумна медицина» та інші – інтегруються в єдину централізовану платформу управління містом, що виконує роль своєрідного «міського мозку». Ця платформа здійснює комплексну обробку вхідних даних: їх фільтрацію, систематизацію, агрегування та аналіз. Результати обробки відображаються на загальноміській інформаційній панелі, що забезпечує візуалізацію поточного стану всіх міських систем і слугує повноцінним аналітичним інструментом для прийняття управлінських рішень на рівні міста.

Якщо на центральній панелі спостерігаються якісь проблеми, то відповідальні особи можуть перейти у конкретну систему управління областю міського господарства прямо з панелі і швидко розібратися, що відбувається та вжити заходів.

5.4. Технологічний базис системи «Smart City»

Практична реалізація концепції «Smart City» потребує сукупності технологій та побудованих на їх основі рішень, що забезпечують технічне втілення «розумного міста».

У технологічній архітектурі «розумного міста» можна виокремити чотири базові складові (див. Рис. 5.3.):

– інтернет речей, технологічна концепція якого забезпечує збір необхідних даних від об'єктів міської інфраструктури та організацію зворотного зв'язку з ними;

– інфраструктуру передачі даних, що забезпечує інтеграцію прикладних додатків з об'єктами міського середовища;

– системи аналізу даних, що дають змогу видобувати корисну інформацію з великих масивів неструктурованих даних;

– систему агрегації та уніфікації даних, призначену для впорядкування і синхронізації значних потоків різнорідної інформації.

Рисунок 5.3 – Технологічна основа технологій «Smart City»

З точки зору функціональних можливостей інтернету речей та інфраструктури передачі даних питання є досить зрозумілими, однак необхідно окремо акцентувати увагу на значенні платформ для обробки даних у рамках зазначеного технологічного контексту. Інформаційний потік у системах «Smart City» є надзвичайно великим, а значна частина даних часто є дубльованою або зайвою. У зв'язку з цим, системи обробки даних виконують надзвичайно важливу роль, оскільки вони забезпечують ефективну фільтрацію, кластеризацію та

аналіз даних. Виявлення прихованих закономірностей є необхідною умовою для забезпечення коректності прогнозів і точності реакцій на виникаючі події.

Саме ці системи відіграють роль інтелектуального ядра розумного міста, виконуючи функції його аналітичного центру. Водночас ефективне функціонування такого міста є неможливим без участі кваліфікованих фахівців, які володіють необхідними знаннями та навичками для роботи з відповідними технологіями.

5.5. Опис роботизованої інтерактивної інфраструктури

Під роботизованою інтерактивною інфраструктурою розуміється вся сукупність наявних у її складі сервісів і систем, пристроїв та обладнання, програмно-технічних засобів і компонентів, мереж і центрів обробки даних, а також відповідних підприємств і організацій, що мають інтерактивну складову. Ця інфраструктура забезпечує організаційну, технічну, технологічну, економічну та нормативно-правову функціональність роботизованих систем і пристроїв інженерної, транспортної та сервісної груп у межах мегаполісу, які взаємодіють між собою з метою формування більш комфортного середовища життєдіяльності людини.

Між окремими елементами інфраструктури існує розгалужена система взаємозв'язків: один і той самий процес може підтримуватися кількома інтелектуальними системами, сервісами або пристроями одночасно; системи можуть здійснювати взаємний обмін даними; системи нижчого рівня ієрархії виступають механізмами реалізації систем вищого рівня тощо. При цьому такі взаємозв'язки можуть бути як прямими та очевидними, так і опосередкованими, однак у жодному разі не менш значущими для функціонування інфраструктури в цілому.

Роботизовану інтерактивну інфраструктуру складають:

1) Центр інтелектуального управління, моніторингу та комунікацій.
2) Роботизовані пристрої та системи.

3) Сервісні центри інфраструктури.

4) Розумні – будинок, ліфти, парадні сходи.

5) Компанії провайдерів пристроїв, сервісів і систем.

6) Системи та пристрої забезпечення технічного зв’язку пристроїв, систем і компонентів середовища.

7) Інтерактивні компоненти та засоби забезпечення взаємодії.

8) Білінговий центр роботизованої інтерактивної інфраструктури мегаполісу.

9) Типові компоненти систем моніторингу.

10) Програмно-технічні системи формування даних про події та їх реєстрації.

11) Програмно-технічні засоби оцінювання поточного стану роботизованої системи або пристрою.

12) Програмно-технічні засоби дистанційного керування роботизованими пристроями та системами.

13) Системи перевірки та тестування.

14) Запобіжні засоби безпечного використання робототехнічних пристроїв.

15) Обладнання систем організації стоянок, паркування та зберігання роботизованих пристроїв.

16) Засоби та системи навігації роботизованих пристроїв.

17) Логістичні центри запасних частин.

18) Станції зарядки акумуляторів та технічного обслуговування.

19) Служби евакуації роботизованих пристроїв.

20) Система комплексного управління ресурсами мегаполісу.

21) Роботизований центр капітального ремонту та утилізації роботизованих пристроїв і систем.

5.6. Інфраструктура комунікацій Smart City

1. Мережі, що використовують для передачі повідомлень.

Перші примітивні експерименти з мережевою автомобільною взаємодією проводилися ще в 1989 році. Більш систематичні дослідження у цій області почалися у 2000-х. Незабаром з'ясувалося, що існуючі технології Wi-Fi не відповідають поставленим завданням. Для вирішення цих проблем було створено нове доповнення до стандарту Wi-Fi – IEEE 802.11p. Новий протокол базується на технології DSRC (Dedicated short range communication), що служить для взаємодії на коротких дистанціях. Технологія наступного покоління називається WAVE (Wireless Access to Vehicular Environment) і надає високошвидкісну передачу даних. Вона дозволяє обмін даними в радіусі 1000 м між об'єктами, що рухаються зі швидкістю до 110 км/год на частоті 10 або 20 МГц. Сервіси, розроблені на даний момент, включають системи кооперативного оповіщення про зіткнення, системи детектування зіткнень, кооперативні системи безпеки перехресть, оповіщення про наближення автомобілів екстрених служб або областей з дорожніми роботами і ін.

Стільниковий зв'язок такий як 4G та LTE / 5G вже є доступним. Декілька комунікаційних технологій малої дальності, такі як WiFi, Bluetooth Low Energy (BLE), 6LoWPAN, BroadBand Global Area Network (BGAN) та Near Field Communication (NFC) включені для полегшення взаємодії та зменшення відсутності рівня зв'язку між найближчими робототехнічними системами одна з одною. Технології комунікації середнього та далекого радіусу дії, наприклад, Worldwide Interoperability for Microwave Access (WiMAX), Z-Wave, ZigBee, та Low Power Wide Area Network (LoRA) були введені для передачі інформації через інфраструктуру робототехнічної мережі, розташованої на більших відстанях [23].

Інтернет-з’єднання є ключовим елементом у системах взаємодії в архітектурі робототехнічних систем. Завдяки своїй доступності, специфічні протоколи комунікації IoT були інтегровані у цей шар для досягнення

енергоефективності, обмеження ресурсів і обробки даних невеликого об’єму у робототехнічних системах. До таких протоколів належать MQTT, CoAP, XMPP, IPv6, UDP, uIP, DTLS, AMQP, LLAP та DDS. Кожен з цих протоколів виконує певні завдання, зокрема: публікацію та підписку на повідомлення, підтримку багатоадресної передачі, обмін миттєвими повідомленнями у реальному часі, організацію мережі з комутацією пакетів, альтернативу TCP, розповсюдження вбудованої мережі системи, забезпечення конфіденційності даних у протоколі датаграм, черги повідомлень для середовища проміжного програмного забезпечення, легку локальну автоматизацію та безпосередню адресацію для комунікацій у реальному часі через публікацію та підписку у вбудованих системах.

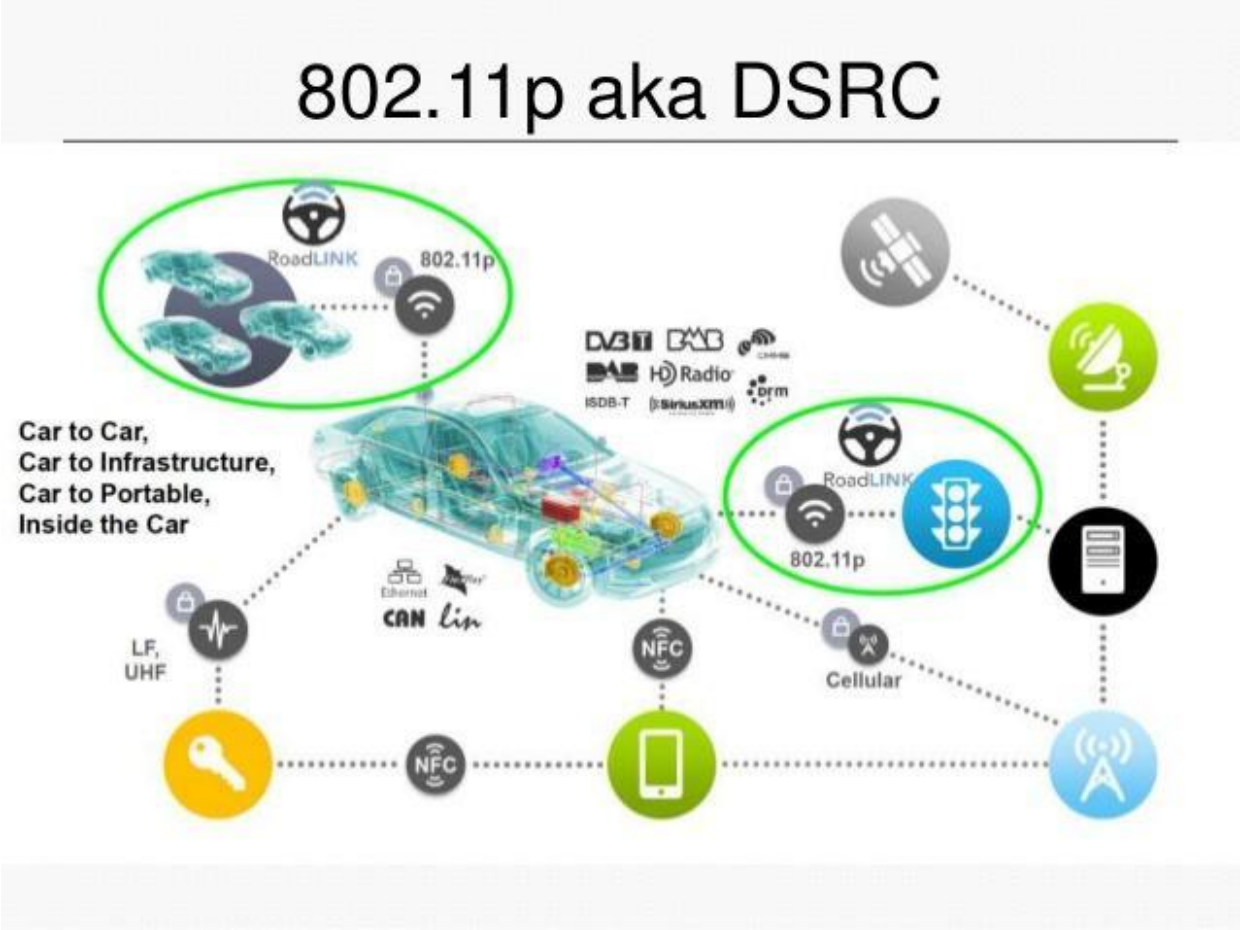

Рисунок 5.4 – Технології зв’язку в робототехнічних системах: від IoT до DSRC

- Рівні інтерактивної взаємодії.

- Аварійні лінії зв’язку.

Система використовує декілька каналів зв’язку і за відсутності одного може використовувати інший.

- Шлюзи даних інфраструктури.

Оскільки запропонована архітектура IoRT використовує інтерфейс Web Service Description Language (WSDL), він прагне стандартизувати кілька комунікаційних інтерфейсів, які розгорнуті для архітектури IoRT. WSDL включено для полегшення загальної комунікації між окремими роботами (або робототехнічними системами) і з іншими сегментами IoRT. Каталог сервісів повинен зберігати інформацію про всі розгорнуті сервіси для роботизованих систем. Всі послуги опубліковані як веб-служби, що робить IoRT простішим для складання складних програм, використовуючи базові веб компоненти [24].

- Топологія комунікацій.

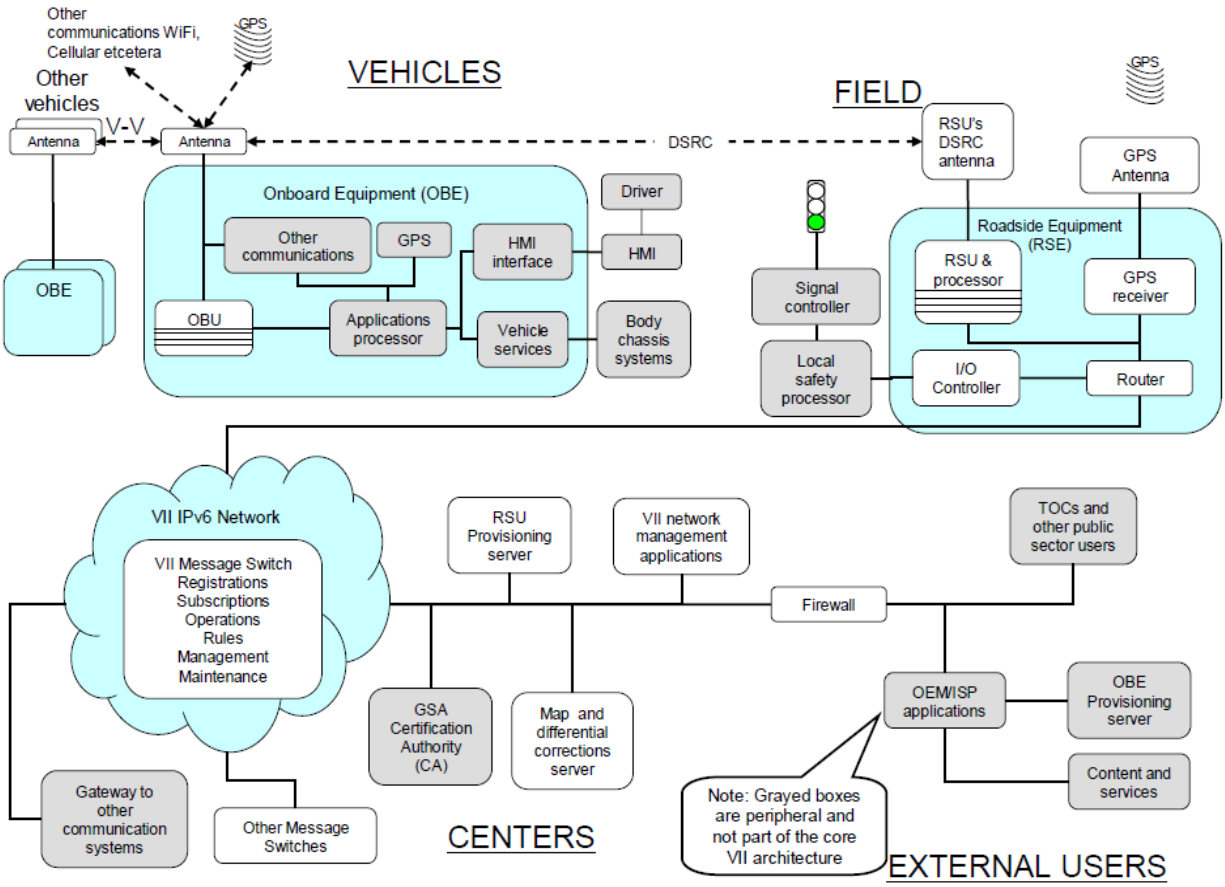

Рисунок 5.5 – Схема інтеграції IoT та V2I у транспортних системах

- Пункти редукції потоків даних IoT.
- Сервери зберігання та аналізу даних.

В останні роки, багато архітектурних рішень було запропоновано з метою створення доступних даних, агрегованих, і які можна застосувати, та придатних для використання тощо. Очевидно, що хмарні і розподілені системи- це основа для рішень для Big Data, які необхідні розумному місту, а рішення IOT – основа для збору даних від датчиків і пристроїв у місті.

Семантичний агрегатор і мірник

При застосуванні моделі семантичного агрегатора та мірника, здійснюється збір даних та сервісів від міських операторів з метою їхньої агрегації та інтеграції в уніфіковане, семантично узгоджене середовище, що базується на багатодоменній онтологічній моделі. Основне застосування онтології у контексті агрегації даних полягає у класифікації сенсорів та актуаторів Інтернету речей (IoT), підключених до брокера IoT за допомогою протоколу виявлення ресурсів у FIWARE. Онтологія може бути використана семантичним агрегатором для моделювання міських об'єктів, доменів та їхньої взаємодії шляхом наповнення таблиць. Подальше застосування полягає у використанні онтології як основи для створення бази знань з метою повторного поєднання даних для формування когерентної моделі, що сприяє зменшенню помилок та інтеграції даних, які представляють ідентичні поняття, але надходять з різних структур, від різних операторів та джерел.

Використання багатодоменної онтології дозволяє створювати моделі, що відображають зв’язок спеціалізації між класами, агрегацію, асоціацію та подібність, які включають дедуктивні процеси при наповнені бази даних RDF. Таким чином, отримана база знань може використовуватися для розробки стратегій поліпшення якості даних, а також підготовки алгоритмів та обґрунтування їх аспектів. Крім цього, така база даних сприяє наданню послуг, які охоплюють багато доменів, а також здійсненню геопросторового та

тимчасового обґрунтування. Ці переваги також є важливими для розробки API, що дозволяють інтегрувати дані та представлення для осіб, які приймають рішення. Завдяки заповненню онтології даними, отримана база знань може бути зручно використана для створення розумних послуг, таких як контекстна маршрутизація, багатомодульна контекстна маршрутизація, пропозиції на вимогу, особисті помічники та інші інтелектуальні сервіси.

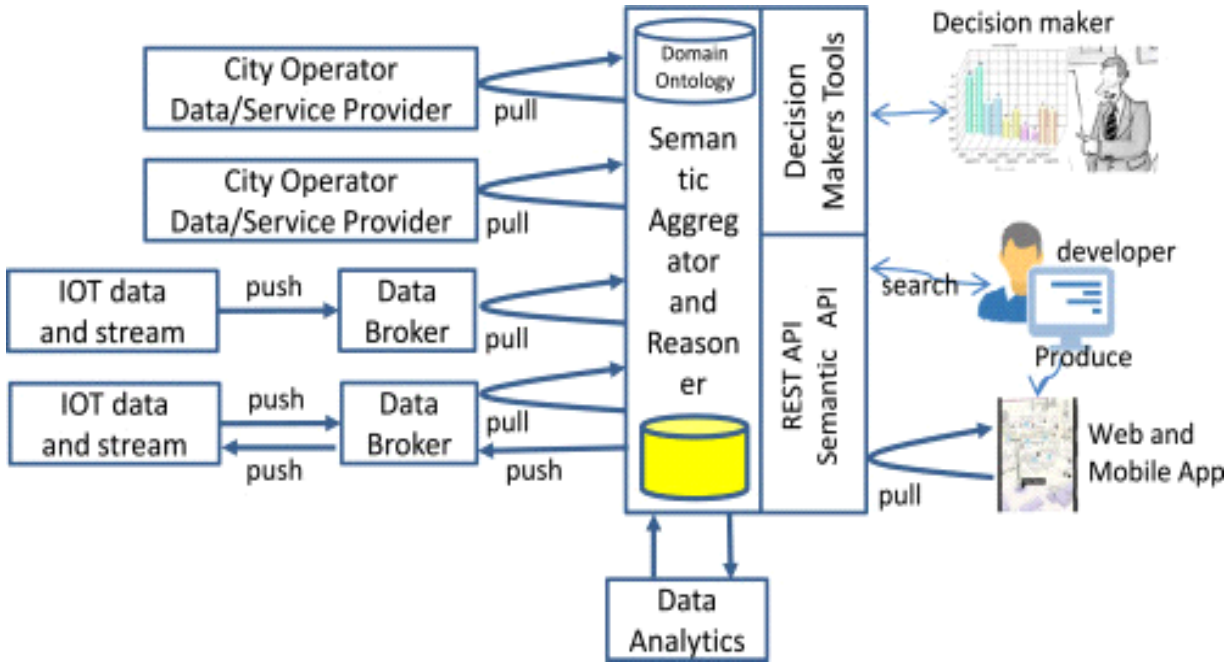

Рисунок 5.6 – Архітектура моделі семантичного агрегатора і мірника

Серед наведених раніше рішень слід відзначити CitySDK, що частково охоплює визначені функції та був розроблений у рамках проекту ЄС. CitySDK включає найбільші міста та надає REST API, що базується на стандартах OASC (Open & Agile Smart Cities). Іншим поширеним рішенням є Km4City, яке також використовувалося у проекті Sii-mobility Smart City, проекті RESOLUTE H2020 та проекті REPLICATE H2020 Європейської комісії.

Крім того, зазначені рішення включають інструменти для відображення потреби в даних щодо онтології та підтримують узгодження даних, як це реалізовано в DataLift та Km4City. В обох випадках як інструменти узгодження використовувалися словники, алгоритми та спеціалізовані мови, зокрема SILK.

Фактично, рішення можуть надавати сервіси, що каталогізують набори даних, оновлюють інформацію про їхнє отримання та забезпечують доступ через автоматично згенерований табличний API, який може бути використаний для оперування семантичною моделлю. Ефективність рішень оцінюється за підцілями Міських Платформ, а саме: надання узгоджених даних та розробка інтелектуальних сервісів. Варто зазначити, що реалізація користувацького досвіду для додаткових сервісів доступна лише у деяких із них. Вищезгадані рішення мають справу з графовими базами даних, які щорічно акумулюють значні обсяги даних, що призводить до сценаріїв використання Big Data, які інтерпретують релевантні характеристики даних, такі як різноманітність, швидкість, достовірність, обсяг тощо.

Ефективна інтеграція даних на семантичному рівні передбачає створення інтелектуальних систем підтримки прийняття рішень, які використовують можливість формування семантичних запитів до множинних доменів для здійснення розподілу обґрунтувань на основі баєсової підтримки прийняття рішень та впровадження алгоритмів для персоналізованої маршрутизації особистих помічників у місті. У певних випадках використання графових баз даних для зберігання та отримання даних розумного міста може бути неефективним рішенням з точки зору продуктивності

Хмарна платформа M2M2A (Machine-to-Machine-to-Actuator) є парадигмою, що підходить для складних роботизованих систем, які розглядаються як критично важливі машини, що сприятимуть розвитку системи IoRT (Internet of Robotic Things). Концепція Machine-to-Machine (M2M) являє собою систему, яку можна інтерпретувати як сукупність взаємопов'язаних машин, що обмінюються інформацією через мережу без безпосередньої участі людини, забезпечуючи автоматизоване оптимальне управління. Система M2M2A призначена для реалізації практичних рішень, де інтеграція різноманітних сенсорних і робототехнічних технологій спрямована на об'єднання фізичного та віртуального просторів. У таких рішеннях сервіси

візуалізації інформації, отриманої від сенсорів, є взаємозалежними, формуючи відповідний ланцюг дій та реакцій, що виконуються роботами. Серед численних функцій найважливішими є збір даних, їх аналіз, керування пристроями, координація даних та акумулювання даних від сенсорів.

- Інфраструктура обробки Big Data.

Ще один важливий напрямок в інтелектуальних транспортних системах – це передбачення заторів. Ідентифікація та прогнозування дорожніх пробок може бути використано як водіями, які прагнуть уникнути заторів, так і системами управління рухом для вжиття заходів щодо їх запобігання. Останні кілька десятиліть найбільш широко поширені техніки дорожнього прогнозування, що ґрунтувалися на фільтрі Калмана і інтегрованій моделі авторегресії змінного середнього аргументу(ARIMA). В даний час велика увага приділяється методам, здатним виконувати прогнозування на основі декількох ознак, що включають дорожній потік, ступінь зайнятості дороги, швидкості. До таких алгоритмів відносяться метод опорних векторів (SVM), нейронні мережі (NN), системи, засновані на правилах нечіткої логіки (FRBS), генетичні алгоритми (GA). Найбільш ефективні методи на сьогоднішній день можуть передбачити виникнення затору на період від 5 до 30 хвилин з точністю у 95%.

Контрольні питання

1. Дайте визначення терміну «Smart City» та його основним компонентам.

2. Назвіть IoT-технології, що лежать в основі інфраструктури «Розумного міста».

3. Охарактеризуйте роль Big Data у підтримці інтерактивної роботизованої інфраструктури.

4. Визначте ключові вимоги до комунікаційної інфраструктури «Smart City».

5. Поясніть, як аналізуються великі дані для прийняття рішень у «Розумному місті».

ТЕМА 6. ВЕБ-СЕРВІСНИЙ ПІДХІД ДО АРХІТЕКТУРИ СЕРВІСУ IOT

6.1. Веб-сервісний підхід для розробки служб IoT

Різнорідність сфер застосування IoT зумовлює існування рішень, потенційно здатних задовольнити вимоги різноманітних сценаріїв використання. Саме ця складність сприяла поширенню різних, часом несумісних пропозицій щодо практичної реалізації систем IoT. Отже, з системної точки зору, реалізація мережі IoT разом із супутніми сервісними та мережевими службами і пристроями ще не має усталеної оптимальної архітектури через її новизну та багатогранність. Окрім технічних викликів, впровадження парадигми IoT також ускладнюється відсутністю чіткої та загальноприйнятої бізнес-моделі, здатної залучити інвестиції для стимулювання розгортання цих технологій

Міська мережа IoT, дійсно, може принести ряд переваг в управлінні та оптимізації традиційних суспільних послуг, розглянутих у другому пункті. Тому необхідно виділити загальні рекомендації проектування міської мережі IoT та розглянути веб-підходи до розробки служб IoT: відповідні протоколи та технології, обговорити їх придатність для середовища Smart City.

Аналіз послуг, представлений у четвертій темі, демонструє, що більшість сервісів Smart City базуються на централізованій архітектурі. У цій архітектурі застосовується гетерогенний набір периферійних пристроїв, розміщених на території міста, які генерують різноманітні типи даних. Ці дані згодом передаються через відповідні комунікаційні технології до центру управління, де відбувається їхнє зберігання та обробка.

Основна характеристика міської інфраструктури IoT полягає у її здатності інтегрувати різні технології з існуючою комунікаційною інфраструктурою для реалізації нових функцій та послуг. Інший фундаментальний аспект полягає у необхідності надання (частини) даних, що збираються міським IoT, органам

влади та громадянам, для підвищення рівня реагування влади на міські проблеми та усвідомлення і участь громадян у суспільних справах.

У цьому розділі буде представлено підхід веб-сервісів до розробки служб IoT, що передбачає розгортання відповідних рівнів протоколів у різних елементах мережі, відповідно до стеку протоколів, зображеного на Рис. 6.1, а також ключових елементів архітектури. Далі буде стисло розглянуто технології каналів зв'язку, які можуть бути застосовані для взаємодії між різними компонентами IoT. На завершення буде надано опис гетерогенного набору пристроїв, що узгоджується з реалізацією міського IoT.

Незважаючи на те, що в домені IoT багато різних стандартів які намагаються бути орієнтиром і найбільш прийнятними, в цьому розділі ми зосереджуємо увагу саме на стандартах IETF, оскільки вони є відкритими та безоплатними, базуються на найкращих практиках Інтернету та можуть розраховуватися на широке коло громадян.

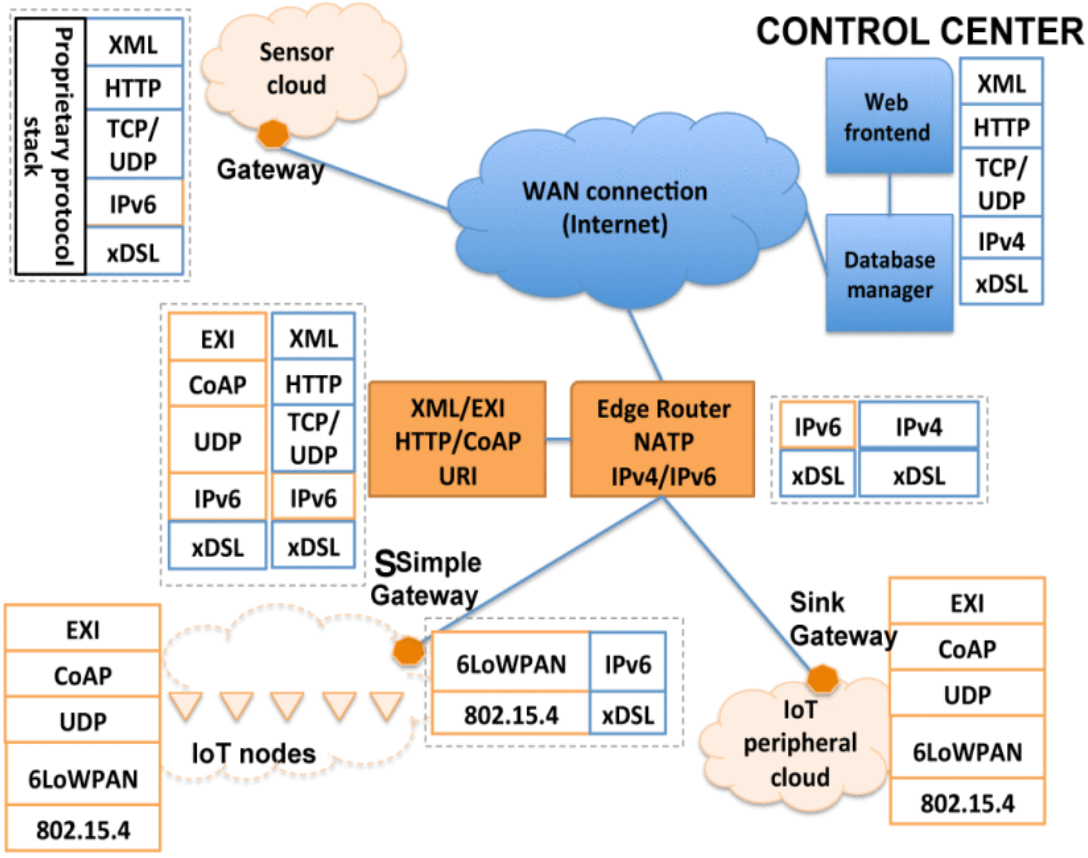

Рисунок 6.1. – Представлення міської мережі IoT на основі веб-сервісного підходу

Серед стандартів IETF для IoT особливе місце посідає архітектура веб-сервісів, яка розглядається в науковій літературі [20,23,24] як один із найбільш перспективних і адаптивних підходів до побудови IoT-систем. В її основі лежить парадигма Representational State Transfer (REST), що забезпечує створення гнучких та інтероперабельних рішень, масштабованих до рівня вузлів IoT. IoT-сервіси, побудовані на принципах REST, структурно близькі до класичних веб-служб, що суттєво знижує поріг входження як для кінцевих користувачів, так і для розробників: накопичені знання у сфері традиційних мереж можуть бути безпосередньо перенесені у проєктування сервісів для мереж із розумними об'єктами. Цей підхід підтримується провідними міжнародними організаціями зі стандартизації – IETF, ETSI та W3C, а також низкою європейських дослідницьких ініціатив у галузі IoT: SENSEI, IoT-A та SmartSantander.

На рис. 6.2. показана архітектура еталонного протоколу для міської системи IoT, що передбачає як незамкнений, так і обмежений стек протоколів. Перша частина складається з протоколів, які у даний час є де-факто стандартами для Інтернет-комунікацій, і часто використовуються звичайними інтернет-хостами, такими як XML, HTTP та IPv4. Ці протоколи відображуються в обмеженому стек протоколі за їхніми аналогами зі слабкою складністю, тобто ефективним обміном XML (EXI), протоколами обмежених програм (COAP) та 6LoWPAN, які підходять навіть для пристроїв з дуже обмеженими можливостями. Операції транскодування між протоколами у лівому та правому стеках на Рис. 6.2. можуть бути виконані стандартним і невисоким рівнем складності, тим самим гарантуючи легкий доступ та взаємосумісність вузлів IoT. Слід зазначити, що системи, які не використовують протоколи EXI / CoAP / 6LoWPAN, все ще можуть бути без проблем включені у міську систему IoT, за умови, що вони здатні взаємодіяти з усіма рівнями лівої частини протоколу в архітектурі на Рис. 6.2.

Протокольна архітектура, представлена на Рис. 6.2, включає три основні функціональні шари: рівень даних, рівень застосунків/транспорту та мережевий

рівень. Для забезпечення ефективної взаємодії між обмеженими та необмеженими форматами й протоколами на цих рівнях можуть бути потрібні спеціалізовані механізми керування та трансляції.

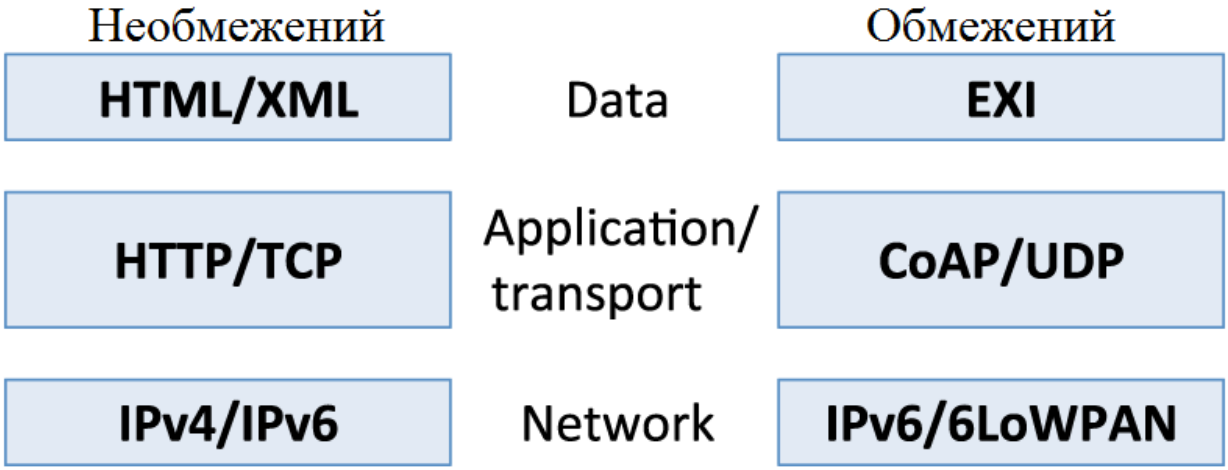

Рисунок 6.2 – Стек протоколів для необмеженого та обмеженого вузлів IoT

6.2. Технології зв’язку для міських систем IoT: класифікація та особливості

Міська система IoT, з огляду на масштаб свого розгортання, потребує комплексу технологій каналів зв'язку, здатних охоплювати широкі географічні зони та одночасно забезпечувати обробку значних обсягів трафіку, що виникає внаслідок агрегації великої кількості дрібних потоків даних. З цих причин технології зв'язку, які забезпечують функціонування міської системи IoT, поділяються на дві категорії: необмежені та обмежені.

До першої групи належать усі традиційні комунікаційні технології стандартів LAN, MAN і WAN — зокрема Ethernet, WiFi, волоконно-оптичні мережі, широкосмугові лінії зв'язку (PLC), а також стільникові технології UMTS та LTE. Вони відзначаються високою надійністю, низькою затримкою та значною пропускною здатністю (порядку Мбіт/с і вище). Водночас через притаманну їм технічну складність та високе енергоспоживання ці технології, як правило, не є оптимальними для периферійних вузлів IoT.

Технології обмеженого фізичного та канального рівня характеризуються низьким енергоспоживанням і відносно невисокою швидкістю передачі даних — зазвичай не більше 1 Мбіт/с. Найбільш поширеними рішеннями цієї категорії є IEEE 802.15.4, Bluetooth і Low Power Bluetooth, IEEE 802.11 Low Power, PLC, NFC та RFID. Такі канали, як правило, мають значні затримки, що зумовлюються двома основними чинниками: по-перше, апріорно низькою швидкістю передачі даних на фізичному рівні; по-друге, політикою енергозбереження, реалізованою у вузлах мережі, яка передбачає переривчастий режим роботи з короткими активними інтервалами.

6.3. Пристрої для мережі IoT

Нарешті, опишемо пристрої, необхідні для реалізації міських IoT, класифікованих за позицією, яку вони займають у потоці комунікації.

Backend сервери

Ядро системи становлять сервери, розміщені у центрі управління, де здійснюється збір, зберігання та обробка даних для формування додаткових сервісів. Попри те що сервери не є обов'язковою умовою функціонування IoT-системи загалом, у контексті міського IoT вони перетворюються на ключовий інфраструктурний елемент: забезпечують доступ до послуг розумного міста та уможливлюють взаємодію з даними через наявну мережеву інфраструктуру. До типових систем, що використовуються для взаємодії з пристроями збору даних IoT, належать такі компоненти:

– С истеми управління базами даних: відповідають за зберігання значних масивів інформації, що генерується периферійними вузлами IoT, зокрема сенсорними пристроями. Залежно від конкретного сценарію застосування, навантаження на ці системи може бути суттєвим.

– Веб-сайти : завдяки широкій поширеності та звичності веб-інтерфейсів для користувачів вони є пріоритетним інструментом взаємодії між IoT-системою

та «споживачами даних» — державними органами, операторами та постачальниками послуг, а також пересічними громадянами.

– Системи планування ресурсів підприємства (ERP): компоненти ERP забезпечують підтримку широкого спектра бізнес-функцій і є важливим інструментом управління інформаційними потоками у складних організаційних структурах, таких як міська адміністрація. Інтеграція взаємопов'язаних модулів ERP із системами управління базами даних, що акумулюють дані, згенеровані IoT-пристроями, сприяє оптимізації роботи з потенційно великими обсягами інформації. Це дає змогу здійснювати сегментацію інформаційних потоків за їх природою та релевантністю з метою спрощення розробки нових послуг.

Шлюзи

На периферії мережі IoT розташовуються шлюзи, основна функція яких полягає в інтеграції кінцевих пристроїв з основною комунікаційною інфраструктурою системи. Згідно з концептуальною протокольною архітектурою, представленою на рис. 6.2, необхідно забезпечити трансляцію протоколів та функціональне відображення між необмеженими протоколами та їхніми обмеженими аналогами, такими як XML-EXI, HTTP-CoAP, IPv4/v6-6LoWPAN.

Хоча зазначені трансляції можуть бути необхідними для забезпечення сумісності з периферійними пристроями IoT та станціями керування, їхня централізація на одному шлюзі не є обов'язковою. Натомість, розподіл функцій трансляції між різними пристроями у мережі є можливим, а у деяких випадках і доцільним. Наприклад, для підтримки кількох маршрутизаторів 6LoWPAN може бути розгорнуто один проксі-сервер HTTP-CoAP.

Пристрої шлюзу також повинні забезпечувати взаємозв'язок між технологіями безперешкодного каналу зв'язку, які переважно використовуються в основній мережі IoT, та обмеженими технологіями, що забезпечують зв'язок між периферійними вузлами IoT.

Периферійні вузли IoT

На периферії IoT-системи розміщуються пристрої, що відповідають за генерування даних, які передаються до центру управління. Вартість таких пристроїв, як правило, є відносно невисокою — від 10 доларів і нижче, залежно від типу та кількості інтегрованих сенсорів і актуаторів. Вузли IoT можуть класифікуватися за різними характеристиками: режимом живлення, мережевою роллю (ретранслятор або кінцевий вузол), наявністю сенсорного чи виконавчого обладнання, а також підтримуваними технологіями каналу зв'язку.

Серед найбільш затребуваних вузлів IoT особливе місце посідають радіочастотні мітки (RF-tags), які, попри обмежені функціональні можливості, здатні відігравати важливу роль у IoT-системах. Це зумовлено їхньою надзвичайно низькою вартістю та пасивним характером комунікаційного обладнання, що не потребує власного джерела живлення. Типовою сферою застосування RF-tags є ідентифікація об'єктів, що знаходить практичне використання в логістиці, технічному обслуговуванні, системах моніторингу та інших галузях.

Контрольні питання

1. Назвіть основні переваги веб-сервісного підходу для розробки IoT-служб.
2. Опишіть функції пристроїв у мережах IoT.
3. Поясніть, як класифікуються технології зв’язку для міських IoT-систем.
4. Як забезпечується сумісність між різними веб-сервісами IoT?

ТЕМА 7. ВИКЛИКИ, ЩО ПОСТАЮТЬ ПРИ ВИКОРИСТАННІ BIG DATA

7.1. Основні виклики при використанні Big Data у розумних містах

Процеси проектування, розробки та розгортання Big Data для розумних міст стикаються з низкою значних проблем. Враховуючи високу динамічність розумних міст як середовищ, особливої ваги набуває мінімізація проблем, пов'язаних зі створенням інтелектуальних застосунків для цієї сфери. Крім того, існують певні дискусії щодо визначення, застосування та переваг Big Data в контексті розумних міст. Ці дискусії стосуються таких аспектів, як наявні інструменти для обробки великих даних, аналітика у реальному часі, точність, представлення, вартість та доступність. Зазначені проблеми можуть негативно впливати на продуктивність інтелектуальних міських застосунків та сервісів, що використовують Big Data. Чи є дані одним із факторів, що спричиняють проблеми? Яким чином? У цьому розділі буде розглянуто ключові виклики, пов'язані з використанням Big Data в розумних містах.

Джерела та характеристики даних: дані створюються з різних джерел у багатьох різних форматах. Спостерігається поява нових форматів даних, значна частина яких є неструктурованою (наприклад, зображення, аудіозаписи, твіти, відеоматеріали, журнали серверів тощо). Ці дані потребують управління та класифікації у структурованому вигляді за допомогою удосконалених систем баз даних. Багатьма дослідниками визначено різні атрибути великих даних, серед яких найбільш важливими є швидкість, обсяг та різноманітність. Додатково було запропоновано такі характеристики, як достовірність, правдивість, мінливість, вартість та варіабельність. Спроба всебічного охоплення цих різноманітних атрибутів призводить до формування складних моделей та підходів, управління якими є проблематичним. Існуючі методології та інструменти вилучення даних часто виявляються недостатніми для опрацювання великих обсягів та складності, що характеризує великі дані, включаючи різноманітність форматів, високу швидкість генерації та значний обсяг. Крім того, у майбутньому

актуальними залишаються питання архітектури систем аналізу великих даних, оцінки ефективності методів обробки та аналізу, розподіленого отримання, еволюціонуючих даних, стиснення, візуалізації та прихованих великих даних. У контексті застосунків розумних міст, що використовують великі дані, виникають значні труднощі, пов'язані як із самими даними, так і з різноманітними супутніми проблемами. З одного боку, збір даних ускладнюється наявністю численних джерел з різними форматами та типами, а також відмінними правилами використання та доступу. З іншого боку, неструктурована природа значної частини даних ускладнює їхню класифікацію, упорядкування та застосування у доступний для додатків спосіб. Ці аспекти вже розглядалися раніше.

Обмін даними та інформацією. Однією з актуальних проблем є організація обміну даними й інформацією між різними міськими департаментами. Кожна державна або муніципальна агенція чи підрозділ, як правило, підтримує власне сховище конфіденційних або загальнодоступних відомостей. Більшість із них часто не схильні ділитися тим, що може вважатися приватними даними. Крім того, певні категорії даних підпадають під дію умов конфіденційності, що додатково ускладнює їх передачу між організаціями. Ключове завдання полягає в тому, щоб не перетнути тонку межу між збором та використанням великих даних і дотриманням права громадян на приватність. Ця проблема є актуальною для будь-якого розумного міста з огляду на різноманіття його секторів і галузей. Відповідні додатки мають шукати шляхи мінімізації або усунення бар'єрів для забезпечення безперешкодного інформаційного обміну між різними структурами. Додаткову складність створює те, що окремі типи даних — зокрема просторово-часові — можуть оновлюватися з високою частотою. Це суттєво ускладнює формування уніфікованого розуміння семантики даних і видобування нових знань на основі даних конкретного циклу та даних у режимі реального часу, що зрештою перешкоджає створенню повноцінної бази знань для розумного міста.

Якість даних. Розглядаючи фундаментальні аспекти великих даних, необхідно виокремити комплекс проблем, пов'язаних із їхньою якістю. Дані, що фіксуються різними суб'єктами у специфічних форматах і зберігаються в різнорідних базах даних, не завжди відповідають стандартним форматам. Унаслідок залучення численних джерел і взаємодії різних провайдерів дані можуть бути позбавлені чіткої структури, що породжує їхню непослідовність, неоднорідність і невідповідність. Таким чином, не існує універсального механізму автоматичного перетворення даних на уніфіковане джерело, придатне для якісного аналізу [25]. Це, своєю чергою, зумовлює додаткові проблеми — невизначеність і ненадійність даних. Наприклад, дані датчиків, зібрані через посередників без централізованого контролю, могли бути сформовані несправними, некоректно відкаліброваними або застарілими пристроями. Проблема може також поширюватися на результати аналізу наявних даних і подальше звітування, якщо кінцеві користувачі не обізнані про існуючі похибки. Тому постійне вдосконалення політики збору, використання та обміну даними між усіма учасниками екосистеми розумного міста, а також забезпечення належного розуміння громадянами відповідних правил, є водночас нагальним і складним завданням.

Безпека та конфіденційність. Серед ключових викликів у контексті розумного міста та використання великих даних особливе місце посідають питання безпеки та захисту персональних даних. Бази даних можуть містити конфіденційну інформацію, що стосується як державних структур, так і приватних осіб, а отже, потребують високого рівня захисту від несанкціонованого доступу та зловмисних атак. Інтегровані між собою додатки також вимагають надійного захисту, оскільки дані циркулюють між різними типами мереж, частина з яких може бути незахищеною. Ситуацію додатково ускладнює те, що провідні технології роботи з великими даними, зокрема Cassandra та Hadoop, на сьогодні мають недостатній рівень вбудованого захисту. Існує також потреба в чіткому визначенні та нормативному захисті права на

приватність як організацій, так і фізичних осіб. Попри те що окремі суб'єкти господарювання можуть формально мати право власності на великі дані, значна їх частина містить персональну інформацію про конкретних людей. Медичні записи, фінансові та банківські дані, історія транзакцій та подібні відомості формують детальний профіль особи, доступ до якого нерідко розглядається як порушення права на приватність. Розробка та належне дотримання суворої політики конфіденційності залишається серйозним викликом для розробників і користувачів додатків на основі великих даних у сфері розумних міст.

Вартість. Це чутливий аспект, що відображає потенційний вплив рішень на основі ІКТ на громадян і державні органи. Наприклад, впровадження системи скорочення енергоспоживання може змусити органи влади встановлювати нові системи, компоненти або функції моніторингу та обліку споживання. Хоча це й сприяє створенню інтелектуальної системи управління енергією, реалізація таких проєктів пов'язана зі значними фінансовими витратами. За умови неналежного планування це може призвести до суттєвого перевищення бюджету та негативних наслідків для міста. Зокрема, тестування інтелектуальних світлофорів і сигнальних систем потребує значних ресурсів і супроводжується не лише високими витратами, а й ускладненням дорожнього руху в процесі фізичного розгортання та випробування. У зв'язку з цим виникає необхідність заміни дорогого обладнання і програмного забезпечення для подальшого розвитку та моніторингу інтелектуальної міської інфраструктури [23].

Населення Smart City. Обсяги великих даних значною мірою визначаються чисельністю міського населення: зі зростанням кількості мешканців масштаби даних, що генеруються, стрімко збільшуються. Це є однією з ключових проблем, оскільки інтенсивне зростання населення загострює труднощі транспортної інфраструктури, погіршує екологічну ситуацію та поглиблює соціальну нерівність. Посилення урбанізації породжує комплекс технічних, соціальних, економічних та організаційних проблем, що становлять загрозу сталому розвитку міст. У цьому контексті інтелектуальні міські додатки

мають оперативно й ефективно розвиватися, щоб справлятися зі зростаючими обсягами та різноманітністю великих даних. Кінцевою метою є створення адаптивних інтелектуальних додатків, здатних забезпечувати ефективне управління динамічним зростанням великих даних задля досягнення оптимальних результатів.

7.2. Проблеми безпеки Big Data

Технології великих даних надають сьогодні істотну цінність для бізнесу, проте одним з обмежень відповідних проектів є ризики інформаційної безпеки. Разом з тим до цього часу немає загальноприйнятої єдиної концепції захисту.

Питанням безпеки систем, що оперують великими даними, часто приділяється недостатньо уваги. Однак, при впровадженні проектів, пов'язаних із великими даними, забезпечення безпеки має бути першочерговим завданням. У іншому випадку, замість отримання бізнес-можливостей, підприємства зіткнуться з додатковими бізнес-ризиками.

Технології великих даних набули сьогодні істотну цінність для бізнесу. Протягом декількох останніх років компанії навипередки запускають проекти, освоюють нові методи виявлення найбільш цінної інформації з доступних їм масивів даних. Збільшення продажів, скорочення витрат, зменшення ризиків, підвищення операційної ефективності - ось лише деякі успіхи, отримані завдяки обробці великих даних при вирішенні бізнес-завдань. Технології обробки великих даних знаходять застосування в найрізноманітніших галузях: телекомунікаційній, фінансовій, роздрібній торгівлі, охороні здоров'я, інформаційних технологіях та багатьох інших. Водночас одним із найсуттєвіших обмежень проєктів у сфері великих даних аналітики вважають ризики інформаційної безпеки.

Безпека в контексті проєктів великих даних — це не лише питання забезпечення доступності інформації. Дані, що слугують джерелом для аналізу, як правило, містять чутливу для бізнесу інформацію: комерційну таємницю та

персональні дані. Порушення конфіденційності під час роботи з такими даними може мати серйозні наслідки — регуляторні штрафи, відтік клієнтів і втрату ринкової капіталізації.

Іншим суттєвим викликом є забезпечення цілісності як вихідних даних, так і результатів їх обробки, що становлять комерційну цінність.

Підстав для занепокоєння чимало. Масштаби витоків вражають: лише у першому півріччі 2024 року в усьому світі, за даними Gemalto, стався витік понад 1,9 млрд записів, а за версією InfoWatch — до 7,78 млрд записів, що у рази перевищує показники попереднього року. За умови недостатньої уваги до питань безпеки проєкти у сфері великих даних здатні збільшити обсяги витоків на порядок.

Наявні підходи до захисту технологій великих даних, як правило, ґрунтуються на розрізнених заходах за відсутності єдиної концепції захисту. Сьогодні бракує чітко сформульованих методів, що описували б систематизовані етапи та дії із захисту як структурованих, так і неструктурованих великих даних, для яких характерні специфічні технологічні особливості збору, агрегування, зберігання та аналізу. Необхідні підходи, орієнтовані на захист критично важливих даних на всіх етапах їх обробки — від збору та передачі до аналізу та розміщення у сховищах.

До роботи зі стандартизації заходів захисту великих даних залучено низку провідних міжнародних інститутів: Міжнародну організацію зі стандартизації та Міжнародну електротехнічну комісію (ISO/IEC), Міжнародний союз електрозв'язку (ITU), Британський інститут стандартів (BSI), а також Національний інститут стандартів і технологій США (NIST). Питанням захисту великих даних приділено особливу увагу і в розділі «Інформаційна безпека» державної програми «Цифрова економіка»: відповідні проєкти національних стандартів мають бути розроблені до кінця 2028 року.

Найбільшого прогресу в цьому напрямі досяг NIST, який запропонував специфікацію Interoperability Framework V1.0 [22], що охоплює документи з

описом усіх аспектів роботи з великими даними: «Definitions», «Taxonomies», «Use Cases and Requirements», «Security and Privacy», «Architecture White Paper Survey», «Reference Architecture» та «Standards Roadmap». Зазначений пакет документів містить методологію, яка також включає питання інформаційної безпеки та представляє концептуальну модель архітектури великих даних, нейтральну щодо постачальників, технологій та інфраструктурних особливостей проєктів. Концептуальна модель NBDRA (NIST Big Data Reference Architecture) визначає систему великих даних як сукупність п'яти логічних функціональних компонентів, пов'язаних між собою через інтерфейси функціональної сумісності.

Контрольні питання

1. Назвіть основні проблеми, які виникають при використанні великих даних у Smart City.

2. Поясніть, як характеристики великих даних впливають на їх обробку.

3. Охарактеризуйте підходи до зменшення проблем, пов’язаних із великими даними.

4. Визначте, які інструменти застосовуються для забезпечення ефективного управління Big Data.

ТЕМА 8. СТРУКТУРА ВЗАЄМОДІЇ BIG DATA ТА SMART CITY

8.1. Обробка великих даних у Smart City: архітектура та технології

У контексті інтелектуальних міських додатків, що генерують безперервні потоки великих даних з гетерогенних джерел, традиційні технології реляційних баз даних виявляються недостатніми для опрацювання таких значних обсягів інформації з огляду на обмежену швидкість обробки та високу вартість масштабування сховища. Для вирішення цієї проблеми були розроблені технології обробки великих даних, що базуються на принципах розподіленого керування даними та паралельної обробки, які забезпечили платформи для зберігання даних, розподіленої обробки та візуалізації інтерактивних даних. Системна архітектура великих даних у розумному місті, зображена на Рис. 8.1, може бути структурована на декілька рівнів з метою забезпечення інтегрованого управління великими даними та технологіями розумного міста. Кожен рівень представляє потенційну функціональність ключових інтелектуальних компонентів обробки даних.

Перший рівень включає множину об’єктів та пристроїв, об'єднаних локальними та/або смуговими мережами. Більшість цих об’єктів і пристроїв безперервно генерують значні обсяги неструктурованих даних кожну секунду.

На другому рівні зібрані неструктуровані дані зберігаються у розподілених відмовостійких базах даних, розміщених або в центрі обробки даних з відповідною мережевою інфраструктурою, або з використанням великих сховищ даних, таких як S3, хмарні сервіси Google і Azure. Для зберігання даних можуть використовуватися різноманітні системи керування великими даними, такі як Cassandra, MangDB, CouchDB, Voldemort, DynamoDB та інші. У межах цього рівня здійснюється зберігання та обробка даних відповідно до вхідних запитів з використанням пакетних модель програмування, наприклад, Framework MapReduce або інших процесорних рушіїв, адаптованих для великих даних. MapReduce забезпечує ефективну парадигму програмування для паралельної та

розподіленої обробки значних обсягів даних на кластері. У випадку обробки потокових даних потрібна швидка обробка для забезпечення можливості реагування компаній та приватних осіб на зміни у реальному часі у середовищі розумного міста. Для обробки потокових неструктурованих даних у реальному часі можуть використовуватися такі технології, як Spark, Storm та S4. Швидкий аналіз може бути реалізований за допомогою масштабованих алгоритмів машинного навчання або інших передових алгоритмів інтелектуального аналізу даних для виявлення закономірностей та отримання знань з великих масивів даних.

Останній рівень представлений службами додатків, через які відбувається безпосередня взаємодія між людьми та машинами для прийняття обґрунтованих рішень. Такі додатки можуть використовуватися для різних цілей, включаючи рекомендаційні системи, виявлення шахрайства, аналіз емоційного забарвлення тексту, інтелектуальне управління дорожнім рухом та аналіз веб-дисплеїв [26].

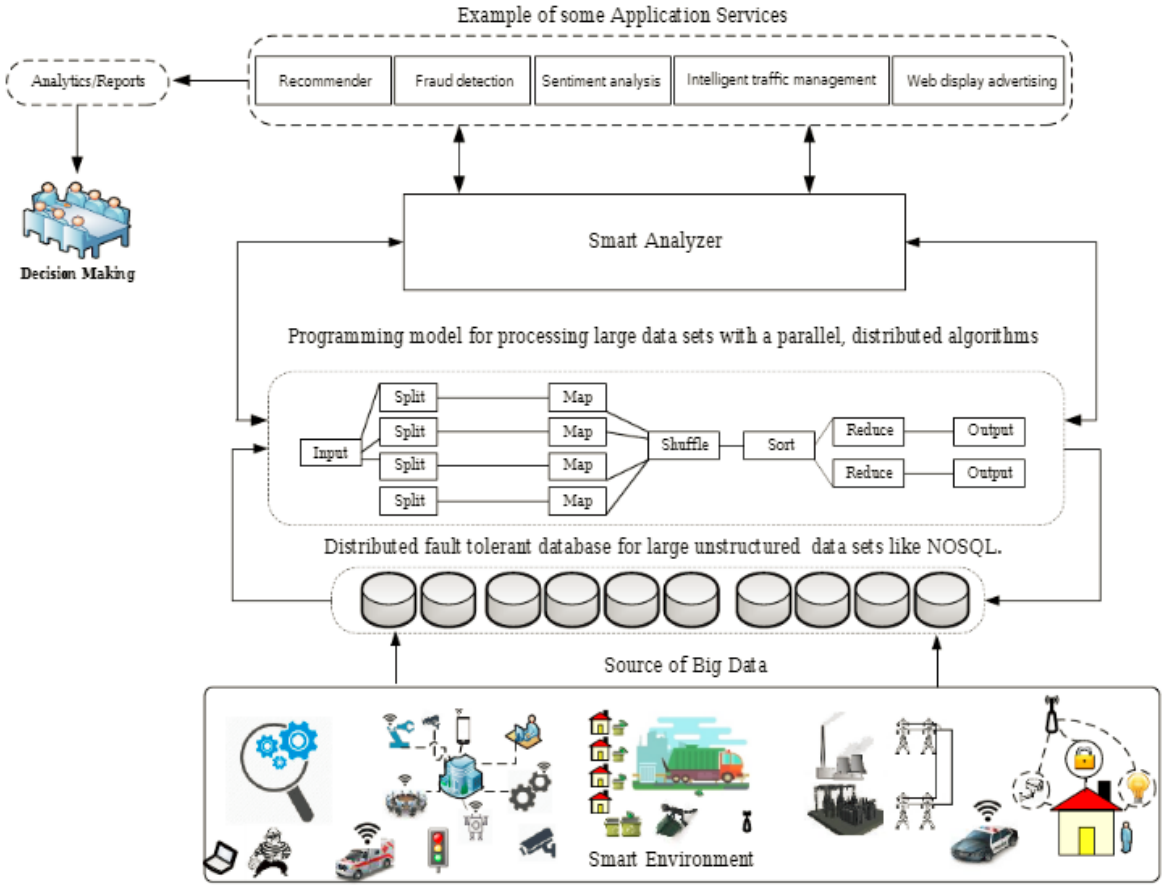

Рисунок 8.1 – Конструкція технологій передачі Big Data для Smart City

8.2. Сервіси Smart City

Незважаючи на те, що ринок розумних міст перебуває на етапі становлення, провідні корпорації, такі як IBM, Microsoft, Huawei та інші (Рис. 8.2), вже активно конкурують за його частку. Це зумовлює залучення науково-дослідного персоналу, який розробляє та впроваджує рішення, що характеризуються значною науковою новизною. Зокрема, інтеграція різнорідних технологій та пошук нових застосувань для існуючих рішень є наслідком реалій, що були розглянуті раніше. Водночас, не обходиться і без розробки принципово нових технологічних рішень.

Розглянемо декілька прикладів сервісів розумного міста.

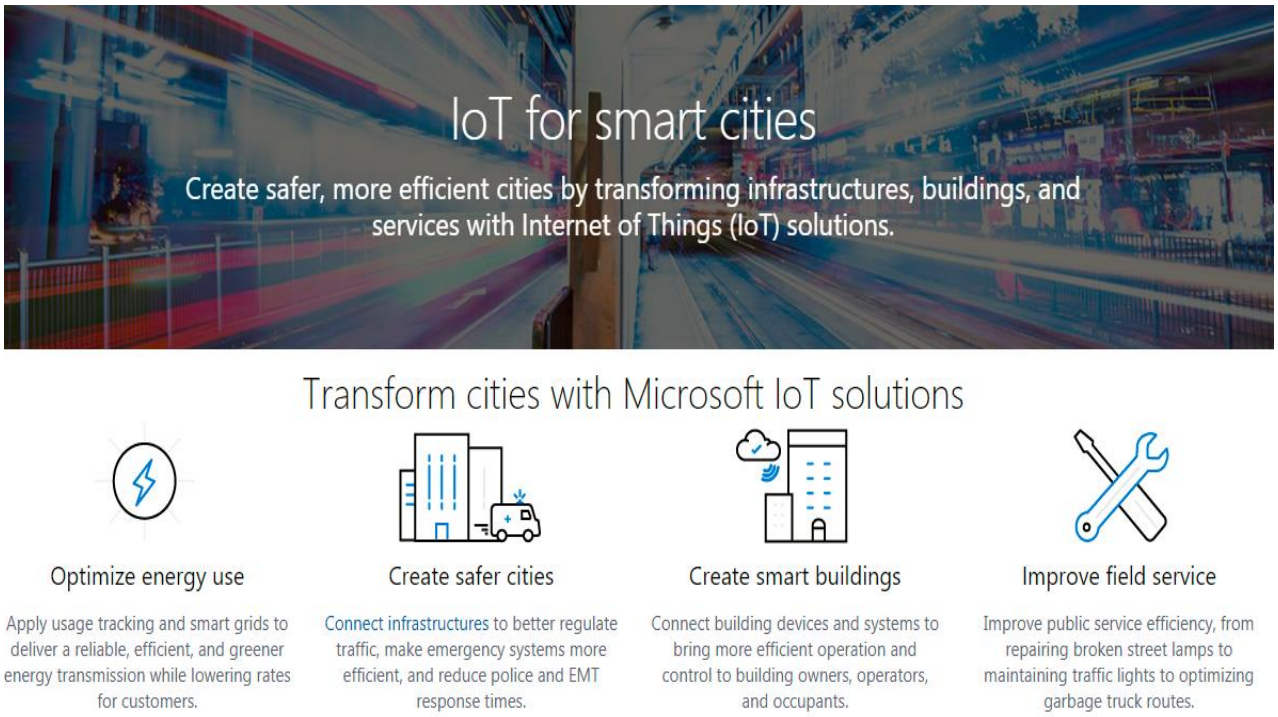

Рисунок 8.2 – Рішення для Smart City представлені на сайті Microsoft

Структурний догляд за будівлями: Належне утримання історичних будівель міста потребує постійного моніторингу фактичного стану кожної споруди та визначення територій, найбільш підданих впливу зовнішніх факторів. Міський IoT може забезпечити розподілену базу даних вимірювань структурної цілісності будівель, отриманий за допомогою відповідних сенсорів, розміщених у спорудах, таких як датчики вібрації та деформації для контролю стійкості

будівлі, датчики атмосферного контролю у прилеглих районах для моніторингу рівня забруднення та температури, а також датчики вологості для повної характеристики умов довкілля. Зазначена база даних повинна зменшити потребу у витратних періодичних структурних інспекціях, що проводяться операторами та сприятиме цілеспрямованому і активному підтриманню та відновленню. Крім того, з'явиться можливість інтегрувати дані вібрації та сейсмічні показники для глибшого вивчення та розуміння впливу слабких землетрусів на міські будівлі. Ця база даних може бути оприлюднена з метою інформування громадян про заходи зі збереження історичної спадщини міста.

Управління відходами: Управління відходами є пріоритетним завданням у багатьох сучасних містах з огляду на витрати на обслуговування та проблему накопичення відходів на звалищах. Однак, більш широке впровадження ІКТ-рішень у цій сфері може призвести до значної економії, а також мати економічні та екологічні переваги. Наприклад, використання інтелектуальних контейнерів для відходів, які визначають рівень заповнення та дозволяють оптимізувати маршрути сміттєвозів, може зменшити витрати на збір сміття та покращити ефективність переробки. Для реалізації управління відходами на основі IoT необхідно підключити кінцеві пристрої, тобто інтелектуальні контейнери для відходів, до центру управління, де програмне забезпечення для оптимізації великих даних обробляє інформацію та визначає оптимальне управління парком сміттєвозів.

Ліфти: це інша потреба, що полягає в більшій ефективності виклику. Шокують результати дослідження IBM, яка зазначила, що в 2020 році люди в Нью-Йорку чекали у цілому 22,5 роки поки спуститься ліфт. Це задача для Big Data. В результаті Allied Market Research очікує, що ринок смарт-ліфтів майже подвоїться з 12 мільярдів доларів у 2015 році до 23 мільярдів доларів в 2022 році.

Можливо, найбільша реалізація інтелектуальної архітектури та інфраструктури – це *інтелектуальні мережі*, які надзвичайно допомагають у збереженні ресурсів та проведенні ремонтних робіт. Європейська комісія очікує,

що до 2025 року 72% споживачів в Європейському Союзі матимуть встановлені інтелектуальні лічильники електроенергії, а 40% – інтелектуальні лічильники газу.

Smart City, Big Data та Internet of Things – сучасні та важливі концепції; отже, багато хто почав інтегрувати їх з метою розробки інтелектуальних міських додатків, які допоможуть досягти кращої стійкості, ефективного управління, підвищення якості життя та інтелектуального управління ресурсами міста.

У цьому пункті було розглянуто низку проблем у зазначеній галузі та виявлено деякі аспекти, що можуть ускладнювати масштабні зусилля з розробки додатків для опрацювання великих даних. Було виокремлено перелік проблемних питань щодо вимог до додатків Big Data для розумних міст. Зазначені вимоги, у поєднанні з мережею IoT, також спрямовані на розв'язання складних завдань та пропонування різних підходів до подолання деяких проблем і досягнення кращих результатів.

Очевидно, що Big Data та IoT можуть зробити величезний внесок у розвиток розумних міст. Тим не менш, виділення концепції Smart City стикається з величезними труднощами, перш ніж справа дійде до реалізації. Схоже, що наявність коштів, конфіденційність даних та соціальні проблеми є найбільшими проблемами. Для того, щоб розумні міста стали глобальним явищем, спочатку потрібно вирішити проблему доступності у країнах третього світу. З огляду на вартість, зрозуміло, що бачення все ще знаходиться на стадії виникнення, і це може зайняти кілька років, перш ніж стане глобальним явищем. Це має бути всеохоплюючою концепцією розвитку міського суспільства.

8.3. Цінність «великих даних» в рамках IoT

Компанії, які хочуть отримати вигоду з «великих даних» (big data), і безпосередньо з промислового «Інтернету речей» (англ. Industrial Internet of Things, IIoT), намагаються тим чи іншим способом визначити цінність даних, зібраних за допомогою цієї технології.

Важливість «великих даних» для розробника промислового обладнання може суттєво відрізнятися від їхньої цінності для виробника будь-якої продукції, орієнтованого на максимізацію прибутку. Отже, для визначення типу даних, необхідних для конкретного додатку IoT, слід розуміти, які бізнес- та виробничі завдання має вирішувати відповідна технологія. Наступним кроком є розробка плану та вибір технологій для належного збору, накопичення, зберігання та аналізу даних. Усі ці етапи необхідні для отримання інформації, яка сприятиме оптимальному використанню наявного обсягу даних у контексті діяльності компанії.

Кінцевий користувач даних у залежності від поставлених цілей може вибрати потрібний тип інформації серед досить широкого спектра, включаючи такі категорії, як прямі і непрямі, або похідні, дані. Прямі, або «сирі» дані, такі як відомості від датчиків віддаленого обладнання, не фільтруються, надходять відразу потоком і часто навіть не перетворюються в інженерні одиниці виміру. Однак непрямі дані, такі як температура двигуна або рівень вібрації, відразу фільтруються по заданих параметрах або критеріям. Дані, що надходять від місць збору даних і подальших розрахунків, можуть включати, наприклад, і показник загальної ефективності обладнання (англ. Overall equipment effectiveness, OEE).

Найчастіше кінцевий користувач прагне оптимізувати пропускну здатність і загальну ефективність промислового обладнання або технологічної установки шляхом порівняння показників від виробничих ліній по всьому підприємству або оптимізації їх функціонування через управління ланцюгами поставок (англ. Supply chain management, SCM). За допомогою цієї інформації компанія може внести у процес позитивні зміни, спрямовані на поліпшення виробничого процесу: наприклад, зрушення виробничих циклів. Альтернативна стратегія заснована на реалізації орієнтованих на майбутнє концепцій, таких як динамічний перерозподіл коштів виробництва та об'єктно-орієнтоване виробництво. При такому підході завод, що підтримує технологію IIoT, може, ґрунтуючись на даних про поточну доступності виробничих ліній, обладнання,

промислових модулів або інших факторах, автономно переміщати етапи виробництва. Таким чином, в індустріальному середовищі «великі дані» використовуються або для підвищення продуктивності машин, або в цілях їх прогностичного технічного обслуговування [15].

Детально проаналізувати функціонування окремої машини можна за допомогою аналітичного програмного забезпечення (ПЗ). Так, інструменти Google Analytics (безкоштовний сервіс, що надається компанією Google для створення статистики) допомагають створювати моделі даних, які у свою чергу дозволяють розробникам виробничого обладнання з'ясувати оптимальні параметри роботи машини і визначити можливості їх поліпшення з точки зору механічних, електронних і програмних компонентів.

8.4. «Великі дані», стандарти IoT і протоколи

Для відповідності вимогам індустріального середовища надзвичайно важливі структури даних і стандарти їх подання, і їх вибір є першим кроком до регульованих методів збору і передачі даних. Оскільки технології IoT і хмарні технології продовжують набирати все більших обертів на промислових ринках, стандартизація даних та протоколів стала значущим чинником, що лежить в основі відповідності даних і взаємодії з ними.

Робочі групи, такі як OPC Foundation, наголошують на необхідності поліпшення збору і передачі даних в системах більш високого рівня при збереженні певних структур даних і прав доступу.

Ряд компаній використовує для передачі даних технічних фахівців і осіб, які приймають рішення, власні протоколи IoT. Ці протоколи призначені для визначення механізму перенесення даних, що являє собою своєрідний канал, через який дані можуть бути переміщені у локальну базу даних або у хмару (як загальнодоступну, так і приватну). При цьому формат запису даних не визначається жорстким протоколом, що дозволяє упаковувати їх в нейтральному форматі – наприклад, у вигляді JavaScript Object Notation (JSON) або у

компактному двійковому коді. Такі формати обміну даними досить прості для розуміння і сприйняття промисловими засобами управління і забезпечують можливість взаємодії різних хмарних платформ, ПЗ середнього рівня і пакетів аналітики.

Впровадження стандартизованих форматів представлення даних є ще одним важливим кроком на шляху інтеграції інформаційних технологій та засобів автоматизації. Це також забезпечує надання промисловими пристроями даних у загальноприйнятому форматі, придатному для аналізу з метою визначення дійсної цінності інформації. Отже, незалежно від пріоритетів розробників технологічного обладнання та компаній, що його використовують, вже існують апаратні та програмні інструменти, які сприяють створенню більш ефективних та цінних даних.

Контрольні питання

1. Назвіть технології, що забезпечують обробку потокових даних у реальному часі.

2. Опишіть, як Big Data та IoT сприяють вдосконаленню управління міськими технологіями.

3. Поясніть функціональність кожного рівня у структурі Big Data для Smart City.

4. Наведіть приклади практичного застосування великих даних у Smart City.

ТЕМА 9. INTERNET OF ROBOTIC THINGS

Робототехнічна система принесла величезні зміни у різні соціально-економічні аспекти людського суспільства у минулому десятилітті. Ці заздалегідь запрограмовані роботи завжди були дуже успішними у своїх досягненнях у ряді структурованих промислових застосувань через їх високу точність, витривалість і швидкість. Робототехнічні технології були інтегровані з існуючими мережевими технологіями щоб розширити діапазон функціональних значень цих роботів, які розгорнуті у неструктурованих середовищах, а також сприяли появі мережевої робототехніки протягом 2000-х років.

IEEE Society of Robotics and Automation's Technical Committee on Networked Robots визначає мережеву робототехнічну систему як сукупність робототехнічних пристроїв, з'єднаних за допомогою дротових та/або бездротових мереж зв'язку. Мережеві робототехнічні додатки можуть бути класифіковані як телекерована робототехніка, тобто віддалено розташовані роботи, що керуються командами, які передаються оператором-людиною через мережу зв'язку, або як системи, що являють собою групу мережевих роботів, розміщених у розподілених локаціях для виконання визначеного завдання шляхом обміну даними з сенсорів та інформацією через окрему мережу зв'язку. Прикладом раннього типу роботизованих систем є марсохід, відправлений для дослідження Марса, тоді як роботи, що грають у футбол, є прикладом сучасного підходу. Мережева робототехніка зазнає впливу властивих фізичних обмежень, таких як низька швидкість виконання бортових інструкцій, обмежений обсяг пам'яті, мережеві затримки, варіативність якості обслуговування, періодичні простої та недостатність даних.

9.1. Хмарна робототехніка

Обмеження дали мотивацію дослідникам розглядати нову форму використання ефективних робототехнічних систем, так звану "Хмарну робототехніку" ("Cloud Robotics"). Хмарна робототехніка може бути описана як

система, яка спирається на інфраструктуру "Хмарних обчислень" для доступу до величезної кількості потужностей обробки та даних для підтримки його роботи. Це означає, що не всі сенсори, обчислення та пам'ять інтегровані в єдину автономну систему, як це було у випадку мережевої робототехніки. Хмарні робототехнічні системи часто включають частину його спроможності для локальної обробки для забезпечення відповідей на запити з низькою затримкою, коли доступ до мережі неможливий або ненадійний, тобто в автономному режимі. Одним із показових прикладів хмарної робототехніки є безпілотні транспортні засоби, розроблені Google. Ці системи використовують індексовані Google Maps, зображення та іншу релевантну інформацію, отриману, зокрема, за допомогою супутників, для забезпечення високоточної локалізації. Хоча хмарна робототехніка користується великою кількістю аналітичних даних, хмарних обчислень, людських обчислень та спільного навчання робота, він страждає від різних факторів, таких як сумісність, неоднорідність, змінний час затримки мережі, безпека, багатопрограмне управління, загальний дизайн інфраструктури, якість обслуговування (QoS) та стандартизація. Відповідно до властивих IoRT переваг якісної обробки згаданих проблем, передбачається, що це дозволить подолати таке обмеження, що веде до більш розумних, спільних, гетерогенних, ефективних, самоадаптивних та ще дешевших робототехнічних мереж.

Основна ідея Internet of Things або IoT не є новою. Ідея IoT була задумана Марком Вайзером в його науковій статті про масове обчислення, названій "Комп'ютер для 21-го століття". Пізніше, в 2009 році, термін Internet of Things був введений Кевіном Ештоном, тодішнім виконавчим директором Auto-ID Center. Згідно Джусто та співавторів, IoT поєднує людей, процес, пристрій і технології з датчиками та виконавчими механізмами. Це загальна інтеграція IoT з людиною включно з комунікаціями, співпрацею та технічною аналітикою дає змогу приймати рішення у реальному часі. Концепція цієї ідеї полягає у взаємодії людини та її соціоекономічного середовища з різними інтелектуальними об'єктами, що включають радіомітки, сенсори, актуатори та розумні пристрої. Ці

об’єкти ідентифікуються за допомогою унікальних схем адресації, використовують захищені канали зв’язку та функціонують у межах стандартизованих архітектурних фреймворків, забезпечуючи взаємодію та сприяючи співпраці з сусідніми об’єктами для досягнення конкретних цілей. Сміт визначає IoT як динамічну глобальну мережеву інфраструктуру з самоконфігурованими параметрами, що базуються на стандартних та взаємосумісних протоколах зв’язку, де фізичні та віртуальні "речі" мають ідентичні фізичні атрибути та віртуальні особистості, використовуючи інтелектуальні інтерфейси та будучи інтегрованими у нерозривну інформаційну мережу; нерідко інформація під час обміну асоціюється з користувачами та їхнім оточенням.

Визначення IoT, що надається Міжнародним союзом електрозв’язку (ITU): глобальна інфраструктура для інформаційного суспільства, яка забезпечує розширені послуги шляхом взаємозв’язку фізичних і віртуальних об’єктів на базі сучасних і перспективних сумісних інформаційно-комунікаційних технологій.

Головною причиною існування різних підходів до сприйняття, розуміння та визначення інтернету речей є те, що цей термін, на відміну від суто технічних понять, не позначає нову технологічну концепцію, а радше нову парадигму формування бізнес-моделі, яка поєднує набір доступних технологічних рішень у взаємопов’язаній та інтегрованій формі. Насправді, більшість технологій, що використовуються в контексті інтернету речей, як-от ідентифікація пристроїв або робота з неоднорідними даними, не є новими. Проте саме їхнє цілеспрямоване використання дає змогу реагувати на актуальні соціальні, технологічні, політичні та економічні запити, що формують попит на нові підходи в інформаційних технологіях.

Інтернет робототехнічних речей, як новітня концепція, потребує чіткого визначення. Згідно з підходом RoboEarth [25], хмарна робототехніка (Cloud Robotics) розглядається як інноваційний напрям, що поєднує можливості робототехнічних систем із хмарними обчисленнями, хмарним зберіганням і

сучасними інтернет-технологіями. Основною метою є використання обчислювальних, комунікаційних і накопичувальних ресурсів центрів обробки даних, з'єднаних через хмарну інфраструктуру, для підвищення ефективності роботів. Це дозволяє мінімізувати проблеми локального обслуговування, оновлення програмного забезпечення та залежності від специфічних платформ, одночасно знижуючи вимоги до енергоспоживання, що, у свою чергу, подовжує час автономної роботи та розширює мобільність. Водночас слід враховувати потенційне зростання експлуатаційних витрат, пов'язаних із передачею даних у хмару, особливо у випадках, що не вимагають обробки в реальному часі.

9.2. Визначення інтернету роботизованих речей

Система підтримки інформаційного суспільства, що забезпечує надання розширених роботизованих послуг шляхом взаємодії між об'єктами робототехнічного інтернету речей, базується на поєднанні наявних та перспективних сумісних інформаційно-комунікаційних технологій, включаючи хмарні обчислення, хмарне зберігання та інші інтернет-технології, які орієнтовані на переваги централізованої хмарної інфраструктури та спільного використання сервісів. Такий підхід дозволяє роботизованим системам ефективно використовувати ресурси сучасних центрів обробки даних – обчислювальні, зберігальні й комунікаційні – водночас знижуючи витрати на технічне обслуговування та оновлення, і підвищуючи автономність користувацьких хмарних рішень, заснованих на універсальних програмних платформах.

Таким чином, Інтернету Роботизованих Речей передбачено бути розташованим на вершині парадигми Хмарної робототехніки, використовуючи певні аспекти хмарних обчислень, таких як технологія віртуалізації та три моделі обслуговування (наприклад, програмне забезпечення, платформу та інфраструктуру), при цьому використовуючи IoT та його технології, що дозволяють розширювати колосальні можливості гнучкості при розробці та

впровадженні нових програм для мережевої робототехніки для досягнення мети забезпечення розподілення обчислювальних ресурсів як основної утиліти. Він поділяє деякі аспекти Хмарної робототехніки та Інтернету речей, однак водночас відрізняється від них у низці аспектів. Завдяки цьому даний підхід відкриває унікальні переваги та формує специфічні виклики, що мають бути враховані для задоволення його вимог.

9.3. Архітектура інтернету роботизованих речей

Архітектуру Інтернету Роботизованих речей можна розділити на 5 рівнів, таких як: (1) рівень обладнання/роботів, (2) мережевий рівень, (3) інтернет-рівень, (4) інфраструктурний рівень, і (5) рівень застосування, як показано на Рис. 9.1.

А. Рівень обладнання

Це найнижчий рівень, що складається з різних роботів і таких речей, як автомобілі, датчики, смартфони, обладнання, датчики погоди, персональне обладнання, побутова техніка та промислові датчики.

Технічно кажучи, фізичні речі (реальні компоненти) охоплюють цей рівень абстракції, забезпечуючи передачу інформації з периферії до вищого, так званого мережевого рівня.

В. Мережевий рівень

У другому нижньому рівні передбачено кілька типів параметрів підключення до мережі. Серед них – стільникові технології, такі як 3G, LTE та 4G [12], які вже широко впроваджені та використовуються. Також застосовуються комунікаційні технології малої дальності, зокрема WiFi, Bluetooth Low Energy (BLE), 6LoWPAN, BroadBand Global Area Network (BGAN) та Near Field Communication (NFC), які сприяють покращенню локальної взаємодії та мінімізації розривів у зв’язку між розташованими поруч робототехнічними пристроями. Для забезпечення стабільної передачі інформації на більші відстані використовуються технології середнього та великого радіуса

дії, такі як Worldwide Interoperability for Microwave Access (WiMAX), Z-Wave, ZigBee та Low Power Wide Area Network (LoRA), що інтегруються в інфраструктуру робототехнічної мережі [25].

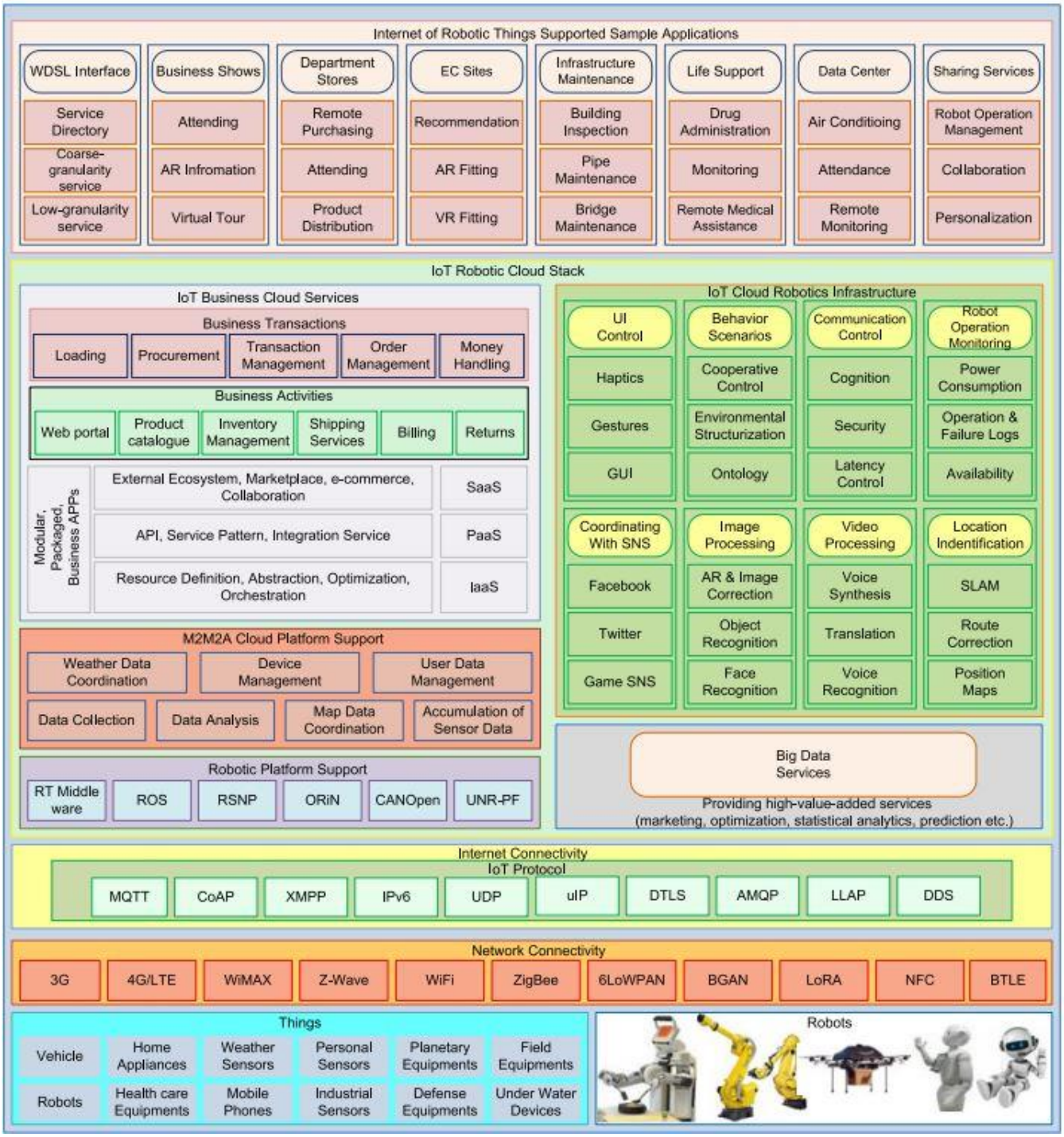

Рисунок 9.1 – Рівні архітектури IoRT

C. Інтернет-рівень

Інтернет-з'єднання становить ключовий елемент комунікаційної архітектури IoRT (див. Рис. 9.1). Завдяки широкій доступності, до цього рівня були цілеспрямовано інтегровані спеціалізовані протоколи зв’язку, притаманні IoT, з урахуванням вимог енергоефективності, обмежених обчислювальних ресурсів і необхідності обробки невеликих обсягів даних у робототехнічних системах. Серед таких протоколів – MQTT, CoAP, XMPP, IPv6, UDP, uIP, DTLS, AMQP, LLAP і DDS, кожен з яких виконує специфічні функції: публікація та підписка на повідомлення, підтримка багатоадресної передачі, обмін миттєвими повідомленнями в режимі реального часу, організація мереж із комутацією пакетів, забезпечення альтернатив TCP-протоколу, поширення вбудованих мереж, захист конфіденційності переданих даних, управління чергами повідомлень у проміжному програмному забезпеченні, локальна автоматизація, а також адресація на основі механізмів публікації та підписки для комунікацій у вбудованих системах реального часу.

D. Рівень інфраструктури

Роботизований хмарний стек, базований на IoT реконструює цю частину архітектури як найцінніший (службові центричні підходи до хмар, проміжне програмне забезпечення, бізнес-процес, і великі дані взагалі) рівень. Цей рівень є конгломератом 5 різних, але споріднених складових, таких як роботизована хмарна платформа, підтримка хмарної платформи M2M2A, хмарні бізнес-сервіси IoT, служби Big Data і IoT хмарної робототехнічнічної інфраструктури.

Підтримка робототехнічної платформи забезпечує роботизовані специфічні сервісні технології, такі як проміжне програмне забезпечення RT (Robot Technology), Robot Operating System (ROS), Robot Service Network Protocol (RSNP), Open Robot/Resource interface for the Network (ORiN), CANOpen, та open source ubiquitous network robot platform (UNR-PF).

Хмарна платформа M2M2A реалізує концепцію «машина–машина–виконавчий механізм» (machine-to-machine-to-actuator), яка орієнтована на

складні робототехнічні системи, здатні виступати як критично важливі елементи архітектури IoRT. Платформа M2M може бути інтерпретована як сукупність взаємопов’язаних машин, об’єднаних у мережу для обміну інформацією без участі людини, з метою забезпечення автоматизованого та оптимізованого управління процесами. Модель M2M2A, у свою чергу, орієнтована на практичне застосування в сценаріях, де необхідна інтеграція різноманітних сенсорних і робототехнічних технологій для синхронізації фізичного та цифрового середовищ. У межах таких систем сервіси візуалізації даних, згенерованих сенсорами, взаємодіють між собою, формуючи логічний ланцюг дій та реакцій, реалізованих роботами. Серед багатьох функцій платформи ключовими є збір даних, їхній аналіз, управління пристроями, координація процесів та агрегація сенсорної інформації.

IoT Business Cloud Services виступають комплексним інструментом для підтримки спеціалізованих сервісів у робототехнічних IoT системах. У межах цієї платформи реалізовано різноманітні бізнес-процеси та транзакції, призначені для обслуговування сервісів моделей SaaS, PaaS та IaaS. Модульні та пакетні бізнес-орієнтовані API забезпечують спрощення і підвищення ефективності операцій, зокрема в контексті електронної комерції. Одночасно з цим виконується управління ресурсами, абстрагування, оптимізація та оркестрація зовнішніх елементів екосистеми. Загалом, бізнес-хмари відіграють ключову роль в екосистемі IoRT, дозволяючи організаціям та виробникам робототехнічних систем зменшити операційні витрати завдяки єдиному багаторівневому підходу, який забезпечує всі необхідні функціональні компоненти підтримки.

IoT Cloud Robotics Infrastructures включається винятково для послуг. Хмара IoT може бути описана як: "Модель, призначена для інформаційного суспільства, що забезпечує надання додаткових послуг через взаємозв'язок (фізичні та віртуальні) речі на основі, існуючих і тих, що розвиваються, взаємодіючих інформаційних та комунікаційних технологій через обладнання

зручної, на вимогу мережі доступної до спільного пулу настроюваних обчислювальних ресурсів (наприклад, мережі, сервери, пам'ять, програми та служби), що можна швидко забезпечити і звільнити з мінімальними зусиллями управління або взаємодії з постачальником послуг, що усуває необхідність і неоднорідні проблеми підключення основних речей у чітко визначені моменти" [24].

У такому контексті хмарні сервіси IoT дозволяють робототехнічним системам бути інтегрованими з численними спеціалізованими службами, серед яких можна виділити обробку зображень, обробку відео, ідентифікацію місцезнаходження, контроль зв'язку, координацію з SNS, управління робототехнічними сценаріями поведінки та керування інтерфейсом користувача. Особливу увагу слід приділяти оптимізації цих служб для забезпечення ефективної взаємодії між компонентами системи.

E. Рівень додатків

Це верхній рівень архітектури IoRT, який призначений для поширення досвіду користувачів через вивчення вибірки програм, що можуть бути виконані за допомогою робототехніки. Роботи, пов'язані з IoT, можуть активно брати участь у вирішенні багатьох проблем на різних рівнях, таких як медичне обслуговування, підтримка інфраструктури, вебсайти, відомчі структури, критичні ситуації, центри обробки даних, бізнес-показники, інтерфейси та багато іншого. Можливості цієї технології є незліченними і постійно зростають, що підкреслює її важливість і необхідність існування.

9.4. Характеристики IoRT архітектури

Інтернет Робототехнічних речей надає кілька основних переваг, які відрізняються від традиційних сервісів робототехніки, таких як хмарна робототехніка і мережева робототехніка, які підсумовані нижче:

1) Комплексність.

Оскільки запропонована архітектура IoRT використовує інтерфейс Web Service Description Language (WSDL), він прагне стандартизувати кілька комунікаційних інтерфейсів, розгорнуті для архітектури IoRT. WSDL включено для полегшення загальної комунікації між окремими роботами (або робототехнічними системами) і з іншими сегментами IoRT. Каталог сервісів повинен зберігати інформацію про всі розгорнуті сервіси для роботизованих систем. Всі послуги опубліковані як веб-служби, що робить IoRT простішим для складання складних програм, використовуючи базові веб-компоненти.

2) Контекст оточення.

На основі даних, отриманих від сенсорів, що фіксують фізичні параметри довкілля, сенсорні вузли, інтегровані у екосистему IoRT, отримують інформацію про навколишній контекст. Рішення, що приймаються робототехнічними системами на основі цих даних, є контекстно-залежними.

3) Віртуалізована диверсифікація.

Запропонована архітектура IoRT використовує спеціальну інфраструктуру компонентів, що містить ідентифікацію місця розташування, базовану на рівні співставлення, відповідальному за відображення об'єктів віртуального робота до фізичних роботів. Таким чином, кінцевий користувач, тобто бізнес, виробник, або особа, запитує лише бажані послуги не враховуючи специфіку фактичного розподілу завдань між роботами для задоволення їхніх потреб. Пропонована архітектура IoRT буде підтримувати та перевіряти гетерогенну робототехніку, де кожен індивідуальний робот (або робототехнічна система) може мати різну апаратну архітектуру та програмне забезпечення. Наприклад, окремі роботизовані системи можуть бути залучені до обслуговування лікарень, інші – ресторанів, ще частина – до виконання розважальних функцій, а також у ролі поліцейських роботів або в рятувальних операціях тощо. Таким чином, архітектура IoRT характеризується значною віртуалізацією та диверсифікацією.

4) Розширюваність.

Підхід до розробки комплексної архітектури IoRT полягає у розширенні наявних робототехнічних сервісів шляхом інтеграції нових типів роботів, наприклад, дронів або сервісних роботів, у модуль хмарної M2M2A, або шляхом оновлення існуючих служб вбудованої системи IoRT. Оновлені сервіси можуть бути легко опубліковані через розроблений веб-інтерфейс.

5) Сумісність.

Пристрої IoT здатні підтримувати множинні взаємопов'язані протоколи комунікації, встановлювати з'єднання з мережею інтернет або відповідними сервісами, а також взаємодіяти з іншими пристроями різного типу та з різною інфраструктурою. Отже, IoRT характеризується сумісністю у власному середовищі.

6) Динамічність та самоадаптація.

Пристрої та системи IoT повинні мати здатність до динамічної адаптації до змінних контекстів та ініціювати дії на основі робочих умов, контексту завдання або сприйнятого середовища. Наприклад, розглянемо систему відеоспостереження, що складається з множини автономних камер, тобто автоматизованих ботів. Ці автоматизовані боти можуть змінювати свої режими функціонування (на нормальний або інфрачервоний нічний режим) залежно від часу доби. Крім того, при виявленні руху автоматизовані боти можуть переходити до режимів підвищеної готовності та сповіщати сусідні автоматизовані боти або інші робототехнічні системи для виконання аналогічних дій. У наведеному прикладі автономні боти демонструють адаптивність, що базується на контексті та динамічних змінах умов.

7) Географічний розподіл та універсальний доступ до мережі.

Хмарні ресурси, як правило, є доступними через мережу Інтернет, яка використовується як основний канал надання послуг. Таким чином, будь-який пристрій з підключенням до інтернету, будь то робототехнічна система, мобільний телефон або інше обладнання, може отримати доступ до розподілених

хмарних сервісів. Крім того, для забезпечення високої продуктивності мережі та локалізації обчислень, багато сучасних робототехнічних систем використовують хмарні центри обробки даних, розміщені в різних віддалених географічних регіонах по всьому світу. Постачальник послуг має можливість використовувати географічну різноманітність для досягнення оптимального рівня обслуговування. Це робить IoRT географічно розподіленим мережевим ресурсом.

Контрольні питання

1. Дайте визначення Інтернету роботизованих речей (IoRT).

2. Назвіть ключові особливості архітектури IoRT.

3. Опишіть, як хмарна робототехніка підтримує розвиток IoRT.

4. Наведіть приклади впливу IoRT на різні галузі, такі як медицина, транспорт чи виробництво.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Internet of Things Smart Cities [Електронний ресурс]. – Режим доступу: http://www.businessinsider.com/internet-of-things-smart-cities-2016-10.

2. Жураковський Б. Ю., Зенів І. О. Технології Інтернету речей : навч. посіб. Київ : КПІ ім. Ігоря Сікорського, 2021. 271 с. URL: https://ela.kpi.ua/items/03f7d3c6-aa4a-4615-b864-0b4a6ab409e4.

3. 1. Wendell Odom, David Hucaby, Jason Gooley. CCNA 200-301 Official Cert Guide Library Premium Edition and Practice Test, 2nd Edition. Published 2024 by Cisco Press. Part of the Official Cert Guide series. ISBN-10: 0-13-822139-1, ISBN-13: 978-0-13-822139-3.

4. Величко В. Ю. Інформаційні системи і технології в економіці : навч. посіб. Київ : Центр навчальної літератури, 2006. 354 с. ISBN 966-364-308-5.

5. Брауде Е. Технологія розробки програмного забезпечення / Е. Брауде. – 2004. – 655 с.

6. Sommerville I. Software Engineering. 10th ed. Pearson, 2015. 816 p. ISBN-13: 978-0-13-394303-0.

7. Fowler M. Continuous Integration // martinfowler.com. 2006. URL: https://martinfowler.com/articles/continuousIntegration.html.

8. 12 кращих методологій розробки програмного забезпечення з перевагами та недоліками [Електронний ресурс]. – Режим доступу: https://www.smart-it.com/uk/2021/08/12-best-software-development-methodologies-with-pros-and-cons/?utm_source=chatgpt.com.

9. Хаф Л. Методологія розробки програмного забезпечення: в 3-х ч. – Ч. 2: Екстремальне програмування [Електронний ресурс]. – Режим доступу: http://www.lib.csu./dl/bases/prg/kompress/articles/2003_10_XP/index.html.

10. Хаф Л. Методологія розробки програмного забезпечення: в 3-х ч. – Ч. 3: Rational Unified Process [Електронний ресурс]. – Режим доступу: http://www.lib.csu.u/dl/bases/prg/kompress/articles/2004_01_upIntro/index.html.

11. What is Adaptive Software Development (ASD)? [Електронний ресурс]. – Режим доступу: https://www.geeksforgeeks.org/adaptive-software-development-asd/.

12. Маніфест гнучкої розробки програмного забезпечення [Електронний ресурс]. – Режим доступу: https://agilemanifesto.org/iso/uk/principles.html.

13. Стандарти з інформаційних технологій [Електронний ресурс]. – Режим доступу: https://nbuv.gov.ua/node/1469.

14. 7. Лосев Ю. І. Комп'ютерні мережі: навчальний посібник / Ю. І. Лосев, К. М. Руккас, С. І. Шматков / За редакцією Ю. І. Лосева. – Х.: ХНУ імені В. Н. Каразіна, 2013. – 248 с .

15. Chen C. P., Zhang C. Y. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data // Future Generation Computer Systems. – 2014. – Vol. 36. – P. 314–347. – DOI: 10.1016/j.future.2013.07.011.

16. Smart city: технології «розумного міста» та їх цільове призначення [Електронний ресурс]. – Режим доступу: https://eukraine.org.ua/ua/news/smart-city-tehnologiyi-rozumnogo-mista-ta-yih-cilove-priznachennya.

17. Wooldridge M. An Introduction to MultiAgent Systems. – WILEY, 2009. – 424 p.

18. Gou Y., Pearce D. A. J., Mitchell P. D. A receiver-based vertical handover mechanism for TCP congestion control // IEEE Transactions on Wireless Communications. – 2006. – Vol. 5, No. 10. – P. 2824–2833. – DOI: 10.1109/TWC.2006.04583.

19. SMART-інфраструктура у сталому розвитку міст: світовий досвід та перспективи України / [керівник проекту та автор – К. Маркевич; наук. консультант – В. Сіденко]. – Київ: Центр Разумкова, 2021. – 367 с.

20. Gáspár P., Szalay Z., Aradi S. Highly Automated Vehicle Systems. – Budapest: BME-MOGI, 2014. – 256 p.

21. The Role of Big Data in Smart City [Електронний ресурс]. – Режим доступу:https://www.researchgate.net/publication/301803005_The_Role_of_Big_Data_in_Smart_City.

22. NIST Big Data interoperability Framework (NBDIF) - Version 1.0 Final [Електронний ресурс]. – Режим доступу: https://www.nist.gov/itl/big-data-nist/documents/nbdif-version-10-final.

23. How Big Data Impacts Smart Cities [Електронний ресурс]. – Режим доступу: https://www.dataversity.net/how-big-data-impacts-smart-cities/.

24. Al Nuaimi, E. Applications of Big Data to Smart Cities / E. Al Nuaimi, F. Al Neyadi, M. Al Mansoori, M. A. M. Zaidi. – Journal of Internet Services and Applications. – 2015. – Vol. 6, Issue 1. – DOI: 10.1186/s13174-015-0041-5.

25. P.P. Ray: Internet of Robotic Things: Concept, Technologies, and Challenges // EEE Access, vol. 4, pp. 9489–9500, Jan. 2017, DOI: 10.1109/ACCESS.2017.2647747.

26. Ricci F., Rokach L., Shapira B. Recommender Systems Handbook. 2nd ed. – Springer, 2015. – 1003 p. – DOI: 10.1007/978-1-4899-7637-6

27. Шипулін В. Д. Основні принципи геоінформаційних систем : навч. посіб. – Харків : ХНАМГ, 2010. – 337 с. – URL: https://eprints.kname.edu.ua/17605/