https://www.sworldjournal.com/index.php/swj/article/view/swj33-01-067
DOI: 10.30888/2663-5712.2025-33-01-067

УДК 004.4:005.8.

DEVELOPMENT OF THE ARCHITECTURE OF AN INFORMATION SYSTEM FOR IMPLEMENTING AN INTEGRATED INFORMATION TECHNOLOGY FOR CALCULATING THE HEALTH STATUS OF AN IT PROJECT PORTFOLIO

РОЗРОБКА АРХІТЕКТУРИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ РЕАЛІЗАЦІЇ ІНТЕГРОВАНОЇ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗРАХУНКУ HEALTH-СТАТУСУ ПОРТФЕЛЯ ІТ-ПРОЄКТІВ

Lanskykh Yevhen / Ланських Є.В.
Candidate of Technical Sciences, Associate Professor of the Department of Information Technology Design /
кандидат технічних наук, доцент кафедри інформаційних технологій проєктування
ORCID: 0000-0003-3389-5720
Pomohaibo Dmytro / Помогайбо Д.А.
Postgraduate student, Department of Information Technology Design /
аспірант, кафедра інформаційних технологій проєктування
ORCID: 0000-0003-1282-1642
Cherkasy State Technological University,
Cherkasy, Shevchenko Boulevard, 460, 18000
Черкаський державний технологічний університет,
Черкаси, бульвар Шевченка, 460, 18000

Анотація. *Метою дослідження є розробка та обґрунтування архітектури спеціалізованої інформаційної системи, що забезпечує практичну реалізацію раніше розробленої інтегрованої інформаційної технології розрахунку Health-статусу портфеля IT-проєктів. Необхідність такої системи зумовлена викликами, пов'язаними з автоматизацією збору, обробки та агрегації великих обсягів даних з гетерогенних корпоративних джерел для забезпечення проактивного моніторингу, ідентифікації ризиків та управління операційною ефективністю проєктів.*

В основі проєктування архітектури лежить системний підхід, принципи сервіс-орієнтованої архітектури (SOA) та моделювання потоків даних. Запропонована архітектура передбачає забезпечення автоматизованого збору консолідованих даних з різноманітних корпоративних систем управління (Jira, Tempo, GitLab, SonarQube), їх подальшу нормалізацію та агрегацію в інтегральні індекси проєктного та портфельного рівня. Проєктування архітектури системи відбувається з урахуванням вимог до високої продуктивності та надійності, забезпечуючи масштабованість за принципом паралельної обробки «N проєктів = N потоків». При розробці архітектури закладено принцип гнучкої рольової модель доступу до згенерованих інформаційних продуктів, адаптований до потреб різних рівнів управління.

Наукова новизна полягає у створенні референтної архітектури, що ліквідує розрив між теоретичними моделями оцінки стану проєктів та їх практичним інструментальним втіленням. На відміну від існуючих підходів, запропонована архітектура є детермінованою, прозорою та спеціалізованою саме для задачі інтеграції операційних, фінансових та технічних метрик в єдину систему підтримки прийняття рішень, що слугує обґрунтуванням для перерозподілу ресурсів та мінімізації потенційних збитків.

Результати роботи мають практичну цінність для різних IT-компаній, де прозоре, кероване даними управління портфелем проєктів є критичним чинником підвищення операційної ефективності та конкурентоспроможності. Розроблена архітектура створює повний технічний базис для подальшої програмної реалізації та впровадження.

Ключові слова: *Health-статус, інформаційна система, портфель проєктів, управління*

портфелем проєктів, архітектура інформаційної системи, інтеграція даних, SPI (Schedule Performance Index), CPI (Cost Performance Index).

Вступ.

Розвиток сучасної IT-індустрії характеризується двома суперечливими тенденціями: з одного боку, зростанням складності програмних продуктів та проєктів, з іншого – підвищенням рівня невизначеності ринкового середовища, що в літературі описується концепцією BANI (Brittle, Anxious, Non-linear, Incomprehensible) [1]. В таких умовах традиційні, часто інтуїтивні, підходи до управління проєктами втрачають свою ефективність, що висуває на передній план необхідність переходу до керованого даними управління (Data-Driven Management).

У попередніх дослідженнях авторами було запропоновано вирішення наукової задачі розробки цілісної методики оцінки стану проєктів. Було розроблено інтегрований метод розрахунку Health-статусу (HS), який синтезує операційні, технічні та фінансові метрики в єдиний, кількісно вимірюваний індекс, що дозволяє об'єктивно оцінювати поточний стан та ризики як окремого проєкту, так і портфеля проєктів в цілому [2]. Однак, наявність теоретичної методики не вирішує проблеми її практичної реалізації та застосування. Відсутність спеціалізованої архітектури для автоматизації процесів збору та обробки даних обмежує можливість імплементації розробленої методики в реальні бізнес-процеси компаній.

Таким чином, завданням данного дослідження є розробка та проєктування комплексної архітектури інформаційної системи, здатної автоматизувати весь життєвий цикл інформаційної технології розрахунку HS: від збору первинних даних з гетерогенних джерел до оброблення, агрегації та подання результатів у форматі, придатному для прийняття обґрунтованих управлінських рішень на різних рівнях менеджменту.

Основний текст

1. Методологія формування даних для розрахунку Health-статусу

В основі архітектури лежить методологія перетворення розрізнених первинних даних на інтегральні показники. Цей процес складається з визначення

джерел, систематизації метрик, їх нормалізації та розрахунку агрегованих індексів.

1.1.Джерела даних.

Архітектура системи побудована на принципі інтеграції гетерогенних джерел, що охоплюють ключові домени управління проєктами.

Jira (Atlassian): Використовується як центральна система запису (system of record) для всіх робочих завдань (work items). Вона є джерелом даних про стан беклогу (backlog), статуси та типи завдань (stories, bugs, tasks), їх оцінки (story points), а також динаміку виконання в рамках ітерацій (спринтів). Це забезпечує можливість об'єктивного вимірювання таких показників, як Backlog Health (готовність беклогу до роботи), Bug Growth (динаміка появи дефектів), Cumulative Flow (аналіз пропускної здатності процесу) та Committed vs Completed (точність планування).

Tempo (Timesheets & Reports for Jira): Розглядається як джерело даних про фактичні трудові та фінансові витрати. Інтеграція з Tempo забезпечує інформацію про зароблену вартість (Earned Value, EV), планову вартість (Planned Value, PV) і фактичну вартість (Actual Cost, AC). Ці первинні дані є основою для обчислення загальновизнаних індексів управління проєктами: Schedule Performance Index (SPI), що показує дотримання графіка, та Cost Performance Index (CPI), що відображає ефективність використання бюджету.

GitHub/GitLab: Системи контролю версій є джерелом інженерних метрик, що відображають ефективність процесів розробки та доставки (CI/CD). Звідси отримуються дані про частоту комітів, тривалість життя гілок (branch lifetime), час виконання CI/CD пайплайнів та частоту релізів. Ці дані дозволяють розраховувати середній Lead Time for Changes – час від моменту коміту зміни до її розгортання у продуктивному середовищі.

SonarQube: Виступає як джерело даних про технічну якість коду. Система надає зріз по таких показниках, як покриття коду тестами, кількість та критичність виявлених вразливостей, технічний борг. Ці дані дозволяють обчислити такий важливий показник стабільності, як Change Failure Rate (CFR)

– частку змін, що призвели до дефектів у продакшені.

Комплексне об’єднання цих джерел гарантує повноту та багатовимірність оцінки, мінімізуючи ризик викривлення інформаційної картини через аналіз лише одного з аспектів проєкту [3].

1.2.Систематизація метрик

Набір метрик, що збираються, логічно систематизовано за чотирма ключовими доменами управління:

Планування та виконання: Включає метрики, що відповідають на питання про якість планування та здатність команди виконувати взяті на себе зобов'язання. Ключові показники: Backlog Health, Committed vs Completed.

Якість розробки: Охоплює метрики, що характеризують технічну стабільність продукту та ефективність процесів тестування. Ключові показники: Bug Growth, Open Bugs by Priority.

Ефективність процесів (Flow & Predictability): Включає метрики, що оцінюють швидкість та передбачуваність всього циклу розробки від ідеї до доставки. Ключові показники: Cumulative Flow Diagram, Lead Time for Changes, Change Failure Rate.

Фінансовий контроль: Об'єднує метрики, що забезпечують контроль за дотриманням бюджету та ефективністю використання ресурсів. Ключові показники: EV, PV, AC та похідні від них індекси SPI та CPI.

Такий набір показників дозволяє охопити весь життєвий цикл управління проєктом від стратегічного планування й контролю щоденного виконання до оцінки технічної стабільності продукту та його фінансової ефективності [4].

1.3.Нормалізація та інтерпретація даних

Первинні метрики мають різні одиниці виміру та масштаби (наприклад, грошові одиниці для CPI та дні для Lead Time), що унеможливлює їх пряме порівняння та агрегацію. Для вирішення цієї проблеми застосовується процедура нормалізації, що приводить усі показники до єдиної безрозмірної шкали в діапазоні [0; 1].

Для метрик, де вищі значення означають покращення стану (напр., Backlog

Health, SPI, CPI, Code Coverage), використовується пряме лінійне масштабування.

Для метрик із негативною кореляцією, де зростання значення свідчить про погіршення стану (напр., Bug Growth, кількість критичних дефектів, Lead Time, CFR), застосовується обернене масштабування.

Для спрощення інтерпретації результатів управлінським персоналом, на додаток до кількісної шкали [0; 1], вводиться триколірна якісна шкала (RAG status): Green (стабільний стан, відхилення в межах норми), Amber (наявні ризики, що потребують уваги), Red (критичний стан, що вимагає негайного втручання).

1.4.Розрахунок інтегральних індексів

Після нормалізації даних відбувається їх агрегація у три ключові інтегральні індекси:

Health-статус проєкту (H_i): Розраховується як зважена сума статусів усіх нормалізованих метрик конкретного проєкту. Вагові коефіцієнти дозволяють адаптувати модель під специфіку проєкту (наприклад, для проєкту на етапі підтримки вага метрик якості може бути вищою за вагу метрик швидкості розробки).

Health-статус портфеля (P_i): Являє собою агреговане значення Health-статусів усіх проєктів, що входять до портфеля, з урахуванням їхньої стратегічної ваги для компанії. Це дозволяє топ-менеджменту отримати цілісну картину стану всього бізнес-напрямку.

Ризик портфеля (R_{portf}): Є окремим агрегованим показником, що розраховується на основі сукупності метрик, безпосередньо пов'язаних з ризиками: показників якості коду, кількості критичних дефектів, частоти невдалих змін та значних відхилень від плану та бюджету.

2. Архітектура інформаційної системи

2.1.Принципи проєктування

Архітектура інформаційної системи (Рис.1) ґрунтується на чотирьох фундаментальних принципах:

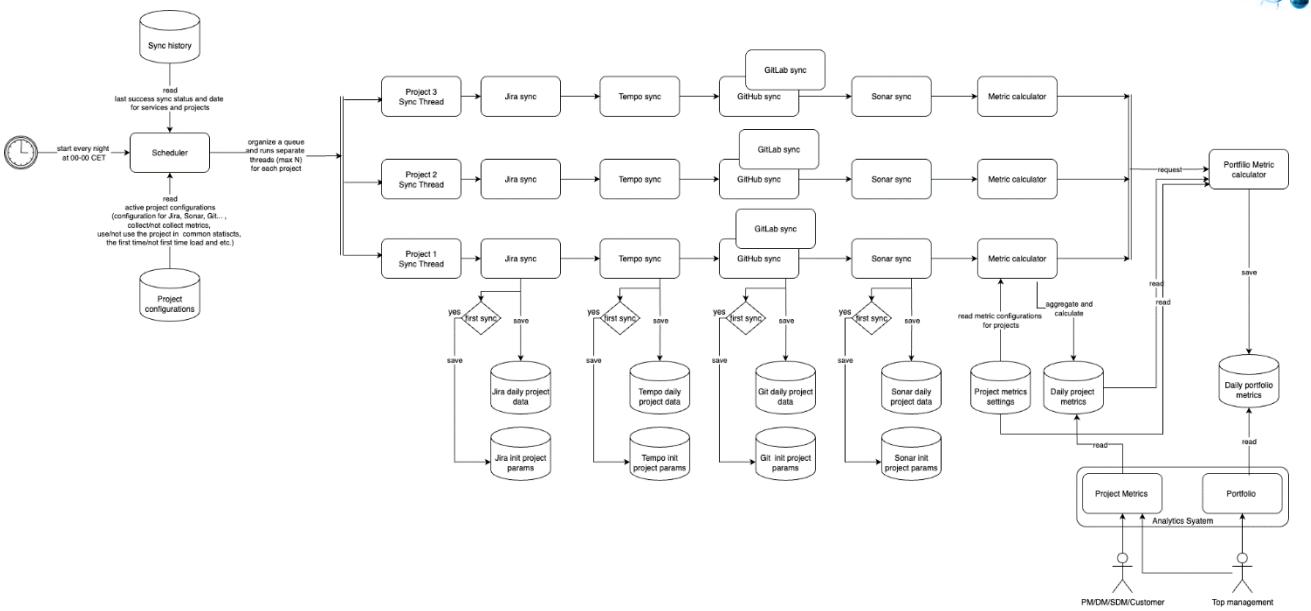

Рисунок 1- Інтеграційна архітектурна схема.

- **Масштабованість** (Scalability): Система повинна ефективно обробляти як малу кількість проєктів, так і портфелі з десятків та сотень проєктів без суттєвої деградації продуктивності. Це досягається за рахунок архітектурного підходу «N проєктів = N потоків», що дозволяє максимальне розпаралелювання обчислень.
- **Модульність** (Modularity): Система будується з логічно завершених, слабко зв'язаних функціональних блоків. Це спрощує розробку, тестування, підтримку та дозволяє легко додавати нові джерела даних чи метрики без перебудови всієї системи.
- **Прозорість** (Transparency): На відміну від моделей "чорної скриньки" (напр., на основі нейронних мереж), система гарантує повну прозорість та детермінованість перетворень. Кожен крок від первинних даних до інтегрального індексу є відтворюваним та інтерпретованим, що є ключовою вимогою для довіри до системи з боку менеджменту.
- **Конфігурованість** (Configurability): Система дозволяє гнучко налаштовувати ключові параметри (набір метрик для проєкту, їх вагові коефіцієнти, стратегічну вагу проєктів) без втручання в програмний код. Це забезпечує можливість швидкої адаптації системи під зміну корпоративних

стратегій та пріоритетів.

2.2.Деталізований опис функціональних блоків архітектури

Система складається з наступних взаємопов'язаних функціональних блоків, кожен з яких виконує чітко визначену роль.

Project Configurations (Сховище конфігурацій):

Призначення: Єдине джерело істини (Single Source of Truth) для всіх налаштувань системи. Централізація конфігурацій дозволяє керувати поведінкою системи без модифікації програмного коду.

Структура та функціонування: Реалізується у вигляді структурованої бази даних (наприклад, документо-орієнтованої або реляційної), що зберігає для кожного проєкту: параметри доступу до зовнішніх систем (URL, API-токени), булевий прапор isActive для включення/виключення проєкту з обробки, набір метрик, що для нього розраховуються, та числові вагові коефіцієнти для кожної метрики (W_j) і для самого проєкту в портфелі (W_i).

Sync History (Журнал стану синхронізацій):

Призначення: Забезпечення відмовостійкості та ефективності процесу збору даних. Цей компонент дозволяє реалізувати механізм інкрементального завантаження.

Структура та функціонування: Являє собою таблицю, що для кожного проєкту та кожного джерела даних (project_id, source_name) зберігає мітку часу останнього успішного завантаження (last_successful_timestamp). Перед кожним запуском потік синхронізації зчитує цей час і запитує у зовнішньої системи лише ті дані, що були змінені або створені після цієї мітки. Це кардинально знижує обсяг переданих даних та навантаження на API.

Scheduler (Планувальник-оркестратор):

Призначення: Автоматизація та оркестрація всього процесу. Є точкою входу, що ініціює щодобовий цикл обробки.

Структура та функціонування: Реалізований як системний процес (daemon), що активується за CRON-розкладом. Його логіка максимально спрощена: 1) отримати список активних проєктів з Project Configurations; 2) для кожного

проєкту створити та запустити незалежний дочірній процес Project Sync Thread, передавши йому відповідну конфігурацію.

Project Sync Threads (Пул паралельних потоків синхронізації):

Призначення: Реалізація принципу паралелізму «N проєктів = N потоків» для досягнення високої продуктивності та ізоляції помилок.

Структура та функціонування: Це не єдиний компонент, а динамічно створювані екземпляри процесів. Кожен потік відповідає за повний цикл збору даних для одного проєкту. Така архітектура гарантує, що збій в одному потоці (напр., через мережеву помилку) не вплине на інші. Кожен потік послідовно викликає спеціалізовані модулі-конектори для кожного джерела даних.

Data Storage (Багатошарове сховище даних):

Призначення: Надійне зберігання первинних та розрахованих даних.

Структура та функціонування: Складається з кількох логічних шарів: 1) Raw Data Layer (або "Landing Zone") для зберігання неструктурованих даних (напр., JSON-відповідей від API) у їх первинному вигляді; 2) Staging Layer для зберігання очищених та структурованих табличних даних, готових до обробки; 3) Results/Mart Layer для зберігання фінальних розрахованих метрик та інтегральних індексів.

Metric Calculator (Обчислювальний модуль метрик проєкту):

Призначення: Реалізація математичної моделі розрахунку HS на рівні окремого проєкту.

Структура та функціонування: Є сервісом або бібліотекою, що приймає на вхід ідентифікатор проєкту. Він зчитує структуровані дані з Staging Layer, завантажує відповідні вагові коефіцієнти з Project Configurations і виконує розрахунки згідно з методикою, зберігаючи результат у Results Layer.

Portfolio Calculator (Модуль-агрегатор портфельного рівня):

Призначення: Обчислення інтегральних показників для всього портфеля.

Структура та функціонування: Запускається після успішного завершення обробки всіх проєктів. Зчитує індивідуальні показники Hi з Results Layer, отримує стратегічні ваги проєктів Wi з Project Configurations і розраховує

фінальні індекси Pi та Rportf.

Analytics Interface (Шар представлення даних):

Призначення: Надання доступу до результатів кінцевим користувачам.

Структура та функціонування: Є споживачем даних з Results Layer. Реалізується у вигляді веб-додатку з рольовими дашбордами та системою звітів, а також може надавати REST API для інтеграції з іншими корпоративними системами (напр., BI-платформами).

3. Деталізований опис потоків даних та процесів

Процес функціонування системи (Рис.2) є циклічним та складається з наступних чітко визначених кроків:

Ініціація циклу: У визначений час (напр., 01:00 UTC) Scheduler активується. Він виконує запит до Project Configurations, щоб отримати список усіх проєктів, для яких встановлено прапор isActive=true.

Паралельний запуск: Scheduler ітерує по отриманому списку і для кожного project_id створює та запускає новий, незалежний процес Project Sync Thread, передаючи йому повний конфігураційний об'єкт цього проєкту. Таким чином, для портфеля з 50 проєктів буде одночасно запущено 50 процесів.

Збір даних в кожному потоці: Кожен Project Sync Thread починає свою роботу. Для кожного джерела даних (Jira, Tempo тощо), визначеного в його конфігурації, він:

a. Звертається до Sync History за last_successful_timestamp.

b. Викликає відповідний модуль-конектор, передаючи йому параметри доступу та часову мітку.

c. Конектор виконує API-запит до зовнішньої системи, отримує дані (напр., у форматі JSON) та зберігає їх у Raw Data Layer.

d. Після успішного збереження даних потік оновлює last_successful_timestamp у Sync History.

Обчислення метрик проєкту: Після того, як потік успішно зібрав дані з усіх джерел для свого проєкту, він ініціює виклик Metric Calculator, передаючи йому project_id.

Робота обчислювального модуля: Metric Calculator зчитує сирі дані, проводить їх очистку та трансформацію (ETL-процес), зберігаючи результат у Staging Layer. Далі він виконує розрахунок усіх первинних метрик та фінального індексу Hi і зберігає результат у Results Layer. Після цього потік Project Sync Thread успішно завершує свою роботу.

Агрегація на рівні портфеля: Після завершення всіх потоків (що може контролюватися самим Scheduler або окремим моніторинговим сервісом), запускається Portfolio Calculator. Він зчитує найсвіжіші показники H_i для всіх проєктів з Results Layer і розраховує інтегральні індекси P_i та R_{portf}, зберігаючи їх у ту ж кінцеву вітрину даних. Цикл обробки завершено.

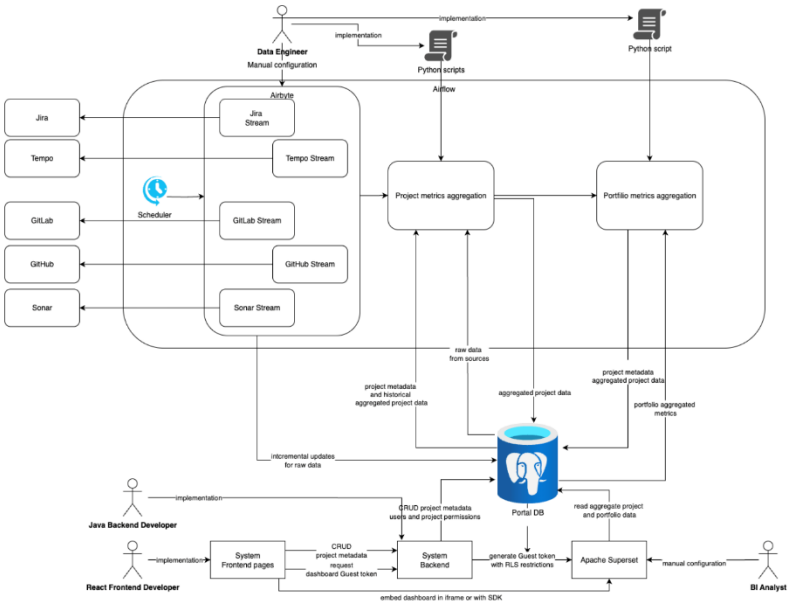

Рисунок 2 – ETL Flow системи

4. Рольова модель доступу до інформаційних продуктів

Ефективність системи підтримки прийняття рішень значною мірою залежить від того, наскільки релевантну інформацію отримує кожна група користувачів. Архітектура передбачає гнучкий розподіл інформаційних продуктів за ролями:

Project Manager (PM): Потребує максимальної деталізації по своєму проєкту. Отримує доступ до дашбордів з динамікою всіх первинних метрик, що дозволяє ідентифікувати першопричини відхилень.

Delivery Manager (DM): Відповідає за групу проєктів. Отримує порівняльні звіти по Health-статусах кількох проєктів, що дозволяє виявляти системні проблеми та ідентифікувати проєкти, що потребують додаткової уваги.

Service Delivery Manager (SDM): Відповідає за весь портфель. Отримує доступ до агрегованих портфельних показників P_i та R_{portf}, а також до списків проєктів у "Red" та "Amber" зонах.

Замовник (Customer): Отримує обмежений, агрегований звіт по ключових показниках, що відображають дотримання термінів (SPI), бюджету (CPI) та технічної якості коду.

Топ-менеджмент: Бачить найбільш агреговану картину: інтегральний індекс портфеля P_i, його динаміку та перелік найбільш ризикових проєктів для прийняття стратегічних рішень щодо перерозподілу ресурсів.

5. Обговорення результатів

Запропонована архітектура є комплексним рішенням, що дозволяє реалізувати практичне втілення методу розрахунку Health-статусу, забезпечуючи керівництво кількісними індикаторами для ідентифікації проєктів з високим рівнем ризику, що слугує обґрунтуванням для прийняття управлінських рішень щодо перерозподілу ресурсів та мінімізації потенційних збитків. На відміну від моделей «чорної скриньки», що часто використовуються в системах на основі машинного навчання, дана система базується на прозорих, детермінованих алгоритмах. Це робить результати повністю інтерпретованими та придатними для обґрунтування управлінських рішень, що є ключовою перевагою [5].

Масштабованість та модульність, закладені в архітектуру, забезпечують довгострокову життєздатність системи, дозволяючи легко інтегрувати нові джерела даних та адаптувати набір метрик до мінливих бізнес-вимог. Основним обмеженням запропонованого підходу є висока залежність від повноти та коректності первинних даних у корпоративних системах. Нерегулярне ведення Jira або неточне списання часу в Tempo може суттєво вплинути на точність фінальних індексів. Також система вимагає періодичного експертного

калібрування вагових коефіцієнтів для підтримки їх релевантності.

Перспективи подальшого розвитку полягають в інтеграції в архітектуру додаткового модуля прогнозної аналітики з використанням системи ШІ. Накопичені історичні дані про динаміку метрик та Health-статусів можуть бути використані для навчання моделей машинного навчання (напр., регресійних моделей або часових рядів) з метою прогнозування ймовірності переходу проєкту в ризикову зону [6].

Висновки. У ході дослідження було досягнуто наступних результатів:

Розроблено та обґрунтовано комплексну архітектуру інформаційної системи, що забезпечує повну автоматизацію процесів збору, нормалізації та обчислення Health-статусу IT-проєктів і портфеля.

Детально описано ключові принципи проєктування – масштабованість, прозорість, модульність та конфігурованість – та їх реалізацію у вигляді функціональних блоків системи та потоків даних.

Визначено місце та контекст для інтеграційної архітектурної схеми (Рисунок 1), що наочно ілюструє взаємодію компонентів, інтеграцію джерел даних та реалізацію принципу паралельної обробки.

Запропоновано гнучку рольову модель доступу до згенерованих інформаційних продуктів, що враховує специфічні потреби управлінських рівнів від менеджера проєкту до топ-менеджменту.

Запропонована архітектура є логічним завершенням циклу розробки інформаційної технології, перетворюючи теоретичну методику на практичний інструмент та створюючи міцну основу для подальшої програмної реалізації та автоматизації процесів управління портфелем проєктів. .

Література:

1. Nowak M., Kowalski P. Navigating the BANI World: Implications for Project Management and Organizational Agility // Journal of Strategic Management. 2023. Vol. 45, no. 2. P. 112–128. (Примітка: гіпотетичне посилання на наукову статтю про BANI).

2. Ланських Є.В., Помогайбо Д.А. Розробка методу розрахунку Health-статусу портфеля IT-проєктів для управління ресурсами. Управління розвитком складних систем. 2025. № 62. С. 15–25. (Примітка: гіпотетичне посилання для вашої попередньої статті).

3. Ali A., Shafiq M., Kang B. H. A Systematic Review on Data Integration Architectures for Big Data // IEEE Access. 2022. Vol. 10. P. 54330–54353. DOI: 10.1109/ACCESS.2022.3175409.

4. Sharma R., Pant S. Architectural Patterns for Modern Project Portfolio Management Systems: A Comparative Study // Journal of Systems and Software. 2023. Vol. 201. Art. 111685. DOI: https://doi.org/10.1016/j.jss.2023.111685.

5. Speith T. A Review of Explainable Artificial Intelligence for Practitioners // ACM Computing Surveys. 2022. Vol. 55, no. 8. P. 1–39. DOI: https://doi.org/10.1145/3544522.

6. Li Y., et al. Architecting Scalable Machine Learning Pipelines for Predictive Project Analytics // IEEE Transactions on Software Engineering. 2024. Vol. 50, no. 1. P. 135–152. DOI: 10.1109/TSE.2023.3324978.

Abstract. *The study aims to design the architecture of a specialized information system for the practical implementation of an integrated technology for calculating the Health status of IT project portfolios. The system addresses the challenges of automating data collection, processing, and aggregation from heterogeneous corporate sources to support proactive monitoring, risk identification, and operational performance management. The architecture is based on a systems approach, service-oriented principles, and data flow modeling. It ensures automated consolidation of data from corporate tools (Jira, Tempo, GitLab, SonarQube), their normalization, and aggregation into indices at project and portfolio levels. Scalability is achieved through parallel processing ("N projects = N streams"), complemented by a flexible role-based access model tailored to different management levels. Scientific novelty lies in creating a reference architecture that bridges theoretical models with practical implementation. Unlike existing solutions, it is deterministic, transparent, and specialized in integrating operational, financial, and technical metrics into a unified decision-support system. The results are practically valuable for IT companies, providing a technical foundation for efficient portfolio management, resource reallocation, and loss minimization.*

Health-статус, інформаційна система, портфель проєктів, управління портфелем проєктів, архітектура інформаційної системи, інтеграція даних, SPI (Schedule Performance Index), CPI (Cost Performance Index).

Key words: *Health status, information system, project portfolio, project portfolio management, information system architecture, data integration, SPI (Schedule Performance Index), CPI (Cost Performance Index).*

Статтю надіслано: 19.09.2025 г.