Специфікація вимог до програмного забезпечення для додатку «Java Frameworks Comparison»

Затверджено версією 1.0

Підготовлено Лаухіним Олександром Андрійовичем

Харківський національний університет радіоелектроніки

18.06.2025

1. Вступ

1.1. Мета

Цей документ визначає функціональні та нефункціональні вимоги до додатку «Java Frameworks Comparison».

1.2. Умовні позначення

Ключові компоненти виділяються жирним, приклади та змінні – курсивом, назви структур, функцій і змінних – у стилі CamelCase.

1.3. Цільова аудиторія

* Java розробнки;
* архітектори програмного забезпечення.

1.4. Область проєкту

Додаток призначено для порівняльного дослідження ефективності Java-фреймворків для обробки великих обсягів даних: **Apache Spark**, **Apache Kafka**, **Apache Storm** та **Spring Batch**. Для кожного фреймворку реалізуються однакові операції обробки даних, що дозволяє об'єктивно порівняти їхню продуктивність, масштабованість та використання ресурсів.

1.5. Посилання

* IEEE Std 830-1998;
* ISO/IEC 25010:2011.

2. Загальний опис

2.1. Перспектива продукту

Додаток є консольною утилітою для вимірювання продуктивності різних Java-фреймворків при обробці великих обсягів даних.

2.2. Основні функції

* операція агрегації: обчислення статистичних показників по групах даних;
* операція з'єднання: об'єднання двох наборів даних за ключем;
* потокова обробка: обробка даних в режимі реального часу з обчисленням ковзних вікон;
* пакетна трансформація: складна ETL-операція з послідовними трансформаціями даних;
* вимірювання метрик: середній час виконання, латентність, пропускна здатність, використання CPU та пам'яті, активність GC;
* масштабування: тестування на різній кількості вузлів кластера;
* генерація звітів: вивід результатів у табличному вигляді та графіки продуктивності.

2.3. Користувачі

Усі користувачі мають однаковий функціонал.

2.4. Оточення

* Java: >=17.0.1;
* Apache Spark: 3.3.2;
* Apache Kafka: 3.4.0;
* Apache Storm: 3.4.1;
* Spring Batch: 5.0.0;
* ОС: Linux, Windows, macOS;
* Кластер: мінімум 2 вузли, рекомендовано 4-16 вузлів.

2.5. Обмеження

* підтримка лише JSON формату вхідних даних;
* мінімальний розмір набору даних 1 ГБ;
* відсутність графічного інтерфейсу (тільки CLI);
* потребує налаштованого кластерного середовища.

2.6. Документація

* README у репозиторії;
* JavaDoc коментарі коду.

2.7. Припущення

* користувач має досвід роботи з Java та розподіленими системами;
* налаштовано кластерне середовище для тестування;
* доступні тестові набори даних.

3. Зовнішні інтерфейси

3.1. Інтерфейс користувача

* CLI-інтерфейс: запуск benchmark-сценаріїв з параметрами;
* конфігураційні файли: налаштування параметрів тестування;
* вивід результатів: консольний вивід у табличному вигляді та збереження у файли.

3.2. Апаратні інтерфейси

* кластер серверів або хмарна інфраструктура;
* мережеве з'єднання між вузлами кластера.

3.3. Програмні інтерфейси

* взаємодія з фреймворками та базою даних (PostgreSQL) через Docker-контейнер;
* файлова система: читання тестових даних та збереження результатів.

3.4. Комунікаційні інтерфейси

* локальне підключення до СУБД PostgreSQL;
* мережеві протоколи: TCP/IP для комунікації між вузлами кластера;
* Storm/Spark/Kafka UI: для моніторингу стану фреймворків.

4. Особливості системи

4.1. Модулі фреймворків

* Apache Spark: реалізація операцій через RDD та Dataset API;
* Apache Kafka: потокова обробка через Kafka Streams;
* Apache Storm: топології для обробки потоків даних;
* Spring Batch: пакетна обробка через Job та Step компоненти.

4.2. Проведення тестів

* запуск кожного фреймворку з однаковими наборами даних;
* фіксація метрик: час виконання, латентність, пропускна здатність, використання ресурсів;
* тестування масштабованості на різній кількості вузлів;
* повторення тестів для статистичної достовірності.

4.3. Збір метрик

* часові метрики: загальний час виконання, латентність операцій;
* ресурсні метрики: використання CPU, пам'яті, активність GC;
* метрики масштабування: ефективність при збільшенні кількості вузлів;
* метрики надійності: стійкість до відмов, час відновлення.

4.4. Налаштування

* можливість налаштувати розмір тестових даних;
* конфігурація параметрів JVM для кожного фреймворку;
* налаштування параметрів кластера;
* вибір типів операцій для тестування.

5. Нефункціональні вимоги

5.1. Продуктивність

* точний вимір часу виконання з мікросекундною точністю;
* мінімальний вплив системи моніторингу на результати тестів;
* підтримка великих обсягів даних (до 200 ГБ).

5.2. Безпека

* відсутність збереження чутливих даних;
* логування без розкриття конфіденційної інформації;
* безпечне управління ресурсами кластера.

5.3. Якість

* відповідність Java Code Style;
* документовані публічні API.

5.4. Надійність

* коректна обробка помилок виконання операцій;
* відновлення після збоїв окремих вузлів;
* валідація вхідних даних та конфігурації.

5.5. Гнучкість

* легке додавання нових типів операцій;
* можливість інтеграції додаткових фреймворків;
* налаштування через конфігураційні файли;
* модульна архітектура для незалежного тестування компонентів.

5.6. Масштабованість

* підтримка горизонтального масштабування;
* ефективна робота на кластерах від 2 до 16+ вузлів;
* адаптація до різних конфігурацій апаратного забезпечення.

Appendix A: Глосарій

* Apache Spark -- фреймворк для обробки великих даних з підтримкою кешування в пам'яті;
* Apache Kafka -- платформа для потокової передачі даних;
* Apache Storm -- система для обробки потоків даних в реальному часі;
* Spring Batch -- фреймворк для пакетної обробки даних;
* RDD -- Resilient Distributed Dataset, основна абстракція даних у Spark;
* JMH -- Java Microbenchmark Harness, інструмент для мікробенчмарків;
* ETL -- Extract, Transform, Load, процес обробки даних.

Appendix B: Моделі аналізу

* діаграма послідовності виконання тестів (див. додаток до дипломної роботи);
* діаграма продуктивності використання наданих ресурсів кожним фреймворком.

Appendix C: TBD

* веб-інтерфейс для візуалізації результатів;
* інтеграція з хмарними платформами (AWS, Azure, GCP);
* додавання підтримки інших Java-фреймворків (Apache Flink, Hazelcast).