УДК 004.738.52

ДОСЛІДЖЕННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СИСТЕМИ МИТТЄВОГО ОБМІНУ ПОВІДОМЛЕННЯМИ НА ОСНОВІ ПРОТОКОЛУ WEBSOCKET

Студ. Н.І. Денисова, гр. МгІТ-2-17
Науковий керівник доц. Т.І. Астістова
Київський національний університет технологій та дизайну

Мета і завдання. Дослідити та порівняти існуючі рішення для створення системи миттєвого обміну повідомленнями на основі протоколу WebSocket. Проаналізувати переваги та недоліки існуючих підходів до розробки таких систем. Завдання – розробити нову систему миттєвого обміну повідомленнями на основі протоколу WebSocket з урахуванням попередньо проведених досліджень.

Об'єкт та предмет дослідження. Основним об'єктом дослідження є системи миттєвого обміну повідомленнями на основі протоколу WebSocket. Предмет дослідження - це слабкі та сильні сторони існуючих систем миттєвого обміну повідомленнями.

Методи та засоби дослідження. Одним із основних методів дослідження систем миттєвого обміну повідомленнями є дослідження відкритих фреймворків, архітектур та підходів для створення подібних систем.

Наукова новизна та практичне значення отриманих результатів. Дане програмне забезпечення відрізняється від її аналогів та покращує функціональні показники. Оптимізований інтерфейс полегшує користувачеві роботу у системі, а удосконалене програмне забезпечення покращує якість передачі даних, що дає більш високу швидкість обміну даними.

Результати дослідження. Основна відмінність між двома підходами до організації введення / виведення в тому, що Java IO є потоко-орієнтірованним, а Java NIO - буфер-орієнтованим.

Підхід, на якому заснований Java NIO трохи відрізняється від Java IO. Дані зчитуються в буфер для подальшої обробки. Ви можете рухатися по буферу вперед і назад. Це дає трохи більше гнучкості при обробці даних. У той же час, вам необхідно перевіряти чи містить буфер необхідний для коректної обробки обсяг даних. Також необхідно стежити, щоб при читанні даних з буферу ви не знищили ще не оброблені дані, що знаходяться в буфері.

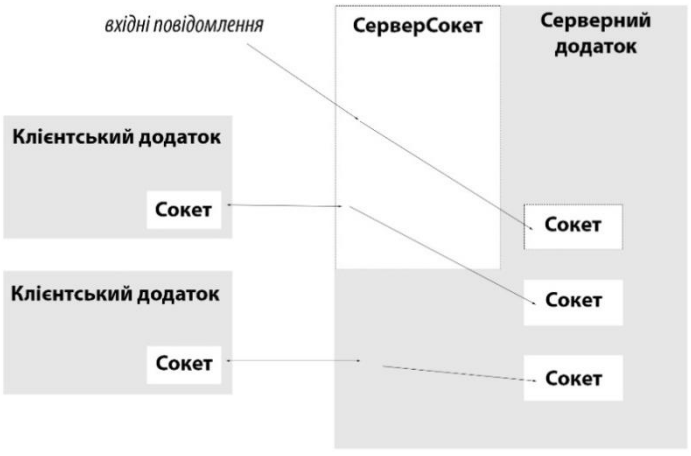

Рисунок 1 − Схема роботи стандартного Java сокету IO

Потоки введення / виведення (streams) в Java IO є блокуючими. Це означає, що коли в потоці виконання (tread) викликається read () або write () метод будь-якого класу з пакета java.io. *, відбувається блокування до тих пір, поки дані не будуть прочитані або записані. Потік виконання в даний момент не може робити нічого іншого.

Неблокуючий режим Java NIO дозволяє запитувати зчитані дані з каналу (channel) і

отримувати тільки те, що є на даний момент, або взагалі нічого, якщо доступних даних поки немає. Замість того, щоб залишатися заблокованим поки дані не стануть доступними для зчитування, потік виконання може зайнятися чимось іншим.

Дане дослідження дало змогу дослідити основні принципи та підходи у розробці систем миттєвого обміну повідомлень. Виявити що реалізація з використанням принципу NIO більш продуктивна та більш економічна ніж підхід IO в мові програмування Java.

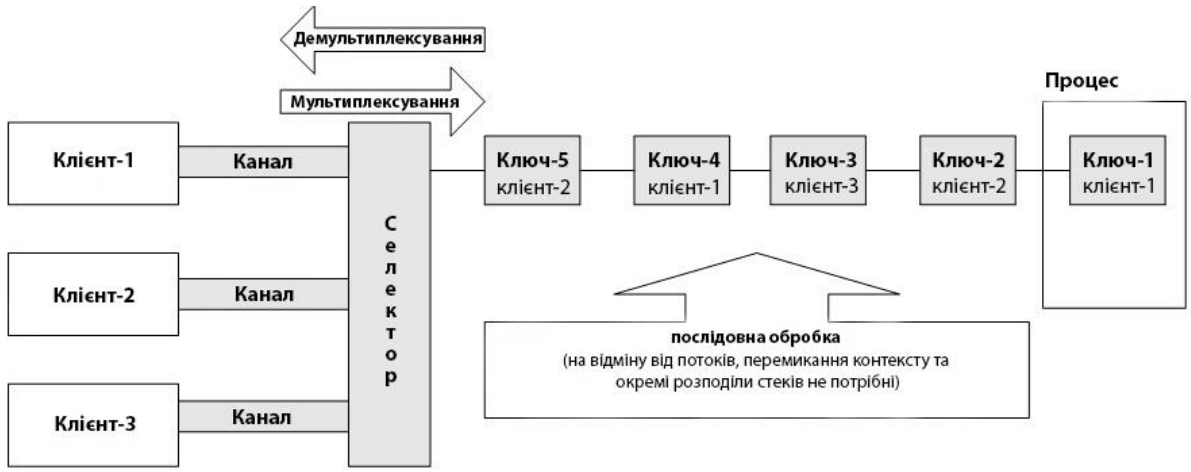

Рисунок 2 − Схема роботи неблокуючого сокет каналу Java.NIO

Висновки . Розроблене програмне забезпечення для системи миттєвого обміну повідомленнями на основі протоколу WebSocket та зручний інтерфейс дадуть змогу поліпшити процес передачі даних.

Ключові слова: *система миттєвого обміну повідомленнями, обмін даними, протокол WebSocket, мова Java.*

ЛІТЕРАТУРА

1. Java NIO vs IO // [Електронний ресурс] – Режим доступу: http://tutorials.jenkov.com/java-nio/nio-vs-io.html#main-differences-between-java-nio-and-io

2. Package java.nio.channels // [Електронний ресурс] – Режим доступу: https://docs.oracle.com/javase/8/docs/api/java/nio/channels/package-summary.html

3. Java SE Net Class Socket // [Електронний ресурс] – Режим доступу: https://docs.oracle.com/javase/7/docs/api/java/net/Socket.html