УДК 004.415.2:654.1

ЗАСТОСУВАННЯ ПЛАТФОРМИ APACHE STORM ДЛЯ ОБРОБКИ ДАНИХ МОБІЛЬНОГО ЗВ'ЯЗКУ

Якушко Максим

Науковий керівник: канд. пед. наук, доцент Лупан І.В.

Центральноукраїнський державний педагогічний університет імені Володимира Винниченка, м. Кропивницький, Україна

Анотація. *У статті розглянуто можливості середовища розробки Apache Storm для обробки великих даних в реальному часі, основні поняття, архітектуру, процес роботи, а також розглянуто вимоги мобільних операторів щодо обробки даних в реальному часі.*
Ключові слова: *великі дані, кортеж, потік, болт, смерч, Apache Storm.*

APPLICATIONS OF APACHE STORM PLATFORM FOR MOBILE DATA PROCESSING

Yakushko Maxim

Scientific supervisor: Candidate of Pedagogical Sciences, Docent Lupan I.V.

Volodymyr Vynnychenko Central Ukrainian State Pedagogical University, Kropyvnytsky, Ukraine

Abstract. *The article considers the possibilities of the Apache Storm development environment for real-time big data processing, basic concepts, architecture, work process, and also considers the requirements of mobile operators for real-time data processing.*

Keywords: *big data, tuple, flow, bolt, tornado, Apache Storm.*

Актуальність теми визначається необхідністю дослідження роботи фреймворку Apache Storm, його використання в різних системах обробки даних та застосування в різноманітних сферах діяльності. На сьогоднішній день все більше інформації потрібно обробляти в реальному часі без затримок, оскільки кожного дня завдяки сучасним інформаційно-комунікаційним технологіям з наших телефонів, комп'ютерів та інших пристроїв накопичується багато структурованих та неструктурованих даних, з яких у разі якісної своєчасної обробки можна добувати цінну інформацію. Одним з відомих інструментів для обробки великих даних в реальному часі є платформа Apache Storm. Основними особливостями цього засобу є простота API, що має всього три абстракції (носик, болт, топологія), простота в програмуванні, вбудовані засоби відмовостійкості та масштабованість. Також Apache Storm гарантує, що кожен кортеж буде обов'язково обробленим. Тому що один із основних механізмів Apache

Storm дозволяє відслідковувати походження кортежу в момент його проходження через топологію дуже ефективним способом. Нижче представлено системи, в яких використовується Apache Storm:

1. Apache Storm глибоко інтегрований з інфраструктурою Twitter'a. *Twitter* використовує Apache Storm для своєї лінійки продуктів Publisher Analytics, які обробляють всі твіти та кліки.
2. *NaviSite* використовує Storm для системи моніторингу та аудиту журналу подій. Всі журнали, згенеровані в системі, проходяти через Storm, який перевіряє повідомлення на відповідність налаштованому набору регулярних виразів, і, якщо збіг знайдено, то таке повідомлення зберігають в базі даних.
3. *Wego* – це механізм метапошуку подорожей, розташований в Сінгапурі. Дані, пов'язані з поїздками, надходять з багатьох джерел по всьому світу з різними термінами. Storm допомагає Wego здійснювати пошук даних в реальному часі, усуває проблеми з паралелізмом і знаходить найкраще відповідність для кінцевого користувача.

Мета дослідження полягає в огляді роботи середовища розробки Apache Storm для обробки великих потоків даних в реальному часі на прикладі даних мобільного зв'язку.

Постановка задачі. Технічне завдання полягало у дослідженні роботи середовища розробки Apache Storm. Дане середовище розробки повинно відповідати таким критеріям:

- бути надійним та зручним для користувача;
- бути відмовостійким, гнучким до підтримання різних мов програмування;
- швидко обробляти великі обсяги даних в реальному часі;
- гарантувати обробку даних при будь-яких несправностях, навіть у разі відмов деяких з вузлів кластеру.

Виклад основного матеріалу роботи. Apache Storm – це розподілене середовище розробки великих даних в реальному часі. Storm призначений для

обробки великої кількості даних в реальному часі, які потрібно безперервно обробляти без будь-яких збоїв та втрати інформації. Це потокова структура даних, яка має високу швидкість прийому.

Основною структурою даних в Storm є кортеж. Кортеж – це список упорядкованих елементів, який за замовченням підтримує всі типи даних. Також він моделюється як набір певних значень, розділених комами, і потім передається до кластера Apache Storm.

Stream (потік) – це невпорядкована послідовність кортежів, яка постійно надходить до компонента Spout.

Spout (джерело даних) – джерело потоків даних, які перетворюються в кортеж і відправляються для подальшої обробки. Як правило, Storm приймає вхідні дані з необроблених джерел, наприклад, таких як Twitter Streaming API, черга Apache Kafka, черга Kestrel і т. д. «ISpout» є основним інтерфейсом для реалізації spouts. Також існує ще декілька інтерфейсів, зокрема IRichSpout, BaseRichSpout, KafkaSpout та інш.

Bolt (болт) – логічна одиниця обробки даних. Spout передає дані до bolt, потім дані обробляють в залежності від топології і створюють новий вихідний потік. Болти можуть виконувати операції фільтрації, агрегації, об'єднання, взаємодії з джерелами даних і базами даних. Болт отримує дані і випускає один або кілька болтів. «IBolt» є основним інтерфейсом для реалізації bolt. Прикладами поширених інтерфейсів є IRichBolt, IBasicBolt і т. п. [1, 3].

Головними перевагами Apache Storm є те, що він відмовостійкий і швидкий. Крім того він не має розподіленого додатку «Single Point of Failure» (SPOF- єдина точка відмови), тобто Apache Storm можна встановити на стільки систем, скільки необхідно для збільшення ємності додатку. На Рис. 1 зображено діаграму, яка ілюструє кластерну архітектуру Apache Storm.

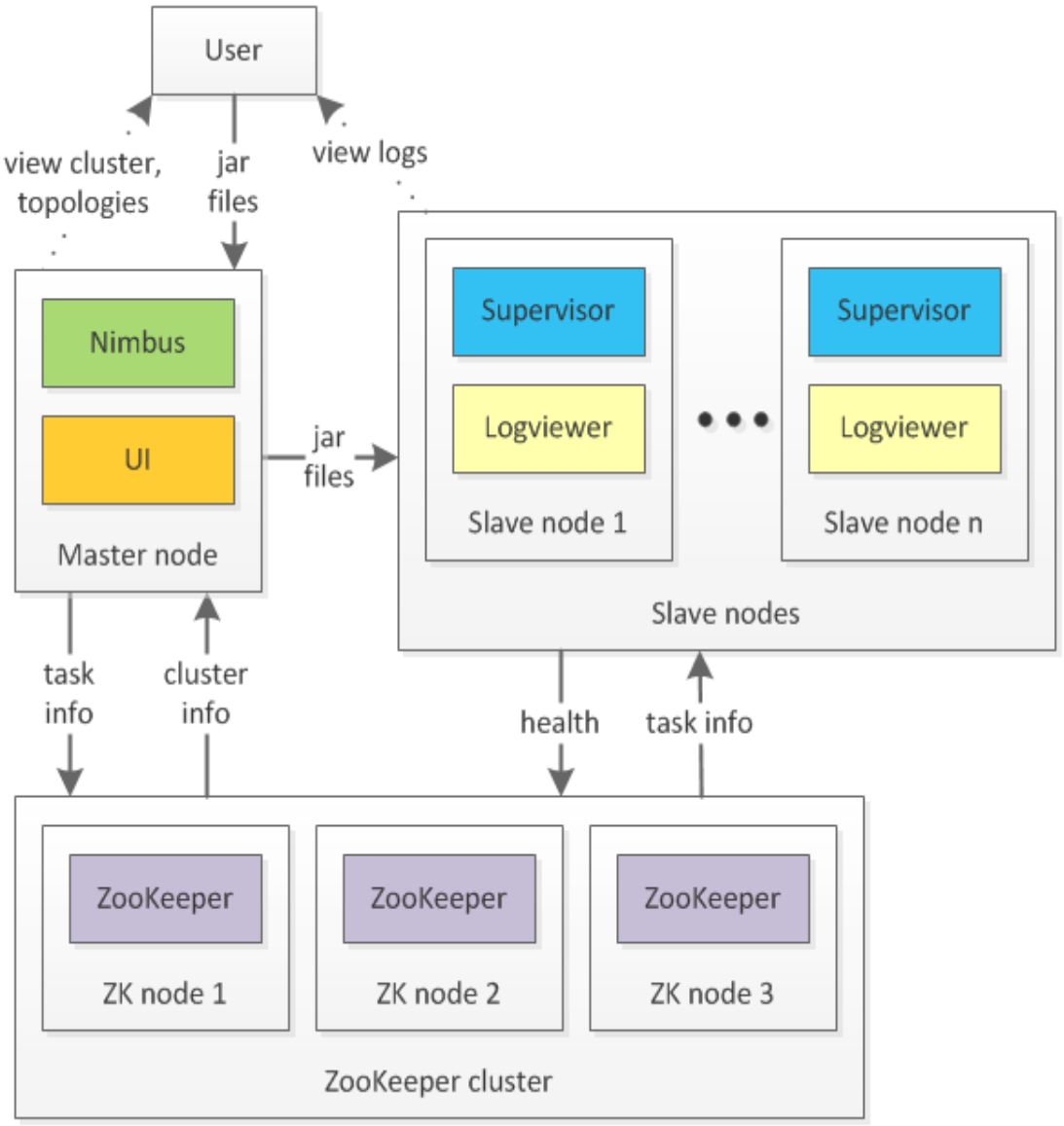

Рис. 1. Архітектура Apache Storm [2].

Apache Storm має два типи вузлів: Nimbus (головний вузол) і Supervisor (робочий вузол). Nimbus – є центральним компонентом Apache Storm. Основною його задачею є запуск топології Storm. Nimbus аналізує топологію і збирає завдання для подальшого виконання обчислень над даними. Потім він поширює завдання серед доступних виконавців. Supervisor має один або кілька робочих процесів. Головний вузол делегує завдання для інших робочих процесів. Робочий процес породжує свої обчислення для стількох виконавців, скільки необхідно для роботи і запускає завдання. Apache Storm використовує

внутрішню розподілену систему обміну повідомленнями між головним вузлом і робочими вузлами для зв'язку між ними [4, 5].

Робочий процес Apache Storm такий:

- спочатку головний вузол чекає, коли йому буде представлена «топологія»; потім аналізує її, збирає завдання, які необхідно виконати, і визначає порядок їх виконання;
- потім головний вузол рівномірно розподіляє завдання між доступними на даний момент часу виконавцями;
- робочі вузли час від часу передають головному вузлу сигнали («сердцебиття») про свою працездатність, а коли вузол «вмирає», тобто перестає передавати сигнали, головний процес передає його завдання іншому робочому вузлу;
- коли «вмирає» головний вузол, то робочі вузли продовжують працювати над вже призначеними завданнями без будь-яких відмов.
- Коли всі завдання виконані, робочі вузли знаходяться у стані очікування нового завдання, а головний вузол, який закінчив роботу, буде автоматично перезапущений за допомогою інструментів моніторингу сервісу.
- Перезапущений головний вузол продовжить роботу з того місця, де він її закінчив. Аналогічно, робочий вузол який закінчив свою роботу, також може бути перезапущений автоматично системою моніторингу. Отже Apache Storm гарантовано виконає всі завдання хоча б один раз.
- Як тільки всі топології будуть оброблені, робочий та головний вузли вузол будуть очікувати надходження нової топології, і нових завдань.

Проблема обробки великих даних [7] є надзвичайно актуальною для операторів мобільного зв’язку: кожного дня зростають обсяги даних, які їм доводиться приймати, аналізувати і обробляти. Для організації робочих процесів в цій сфері потрібні передові технології, щоб забезпечити себе і абонентів від непередбачуваних збоїв і несправностей. Центр обробки даних

(ЦОД) є універсальним очислюючим засобом, який дозволяє вирішувати абсолютно різні задачі, як то:

- збір статистичних даних роботи мережі;
- обробка та моніторинг мережі та абонентів;
- аналіз даних, які надходять від відділів роботи з абонентами;
- оперативне регулювання при збоях в роботі мережі;
- обробка рекомендацій для усунення недоліків і збоїв в роботі мережі;
- обробка рекомендацій для покращення якості роботи мережі;
- прогнозування майбутніх збоїв, які можуть виникнути при роботі мережі(моделювання, навантаження на мережі за даними які наповнюють мережу) [6].

Наприклад Українська компанія Vodafone має свій сервіс обробки великих даних в хмарі. Послуга Data-sa-Servise (BdaaS) роздається з власних ЦОДів компанії. Це допомагає клієнтам компанії працювати з великими даними, навіть якщо в них немає власних необхідних для обчислення потужностей.

Для демонстрування роботи середовища розробки Apache Storm нами був створений аналізатор журналу мобільних викликів.

Мобільний виклик та його тривалість передається в якості даних для Apache Storm, який обробляє та групує виклики між абонентом та отримувачем, що мають спільний сеанс зв’язку, а також аналізує загальну кількість викликів.

Інформація журналу викликів містить в собі номер абонента, який телефонує, номер абонента, який приймає виклик, та триваліть самого виклику. Для демонстрації роботи аналізатора в додатку генеруються підроблені дані про виклики (оскільки у нас немає можливості отримувати дані в реальності про виклики реальних абонентів). Підроблені дані ствоюються за допомогою класу Random.

Потім вхідні данні передаються в компонент Bolt, який обробляє кортежі вхідних даних і формує нові кортежі вихідних даних. Bolt створює нове значення та комбінує номер абонента, який робить виклик и номер отримувача.

Формат кортежу такий: «Номер абонента, що ініціалізує виклик – Номер абонента, який отримує виклик». Далі виконується ініціалізація об’єктів, перевірка кортежів та створення нового значення поля «Виклик».

Після всіх маніпуляцій створюється топологія за допомогою структури Thrift. Thrift – мова опису інтерфейсів, яка використовується для визначення і створення служб для різних мов програмування (є фреймворком для віддаленого виклику процедур).

В цілях розробки був створених локальний кластер, який приймає топологію. Він має клас «Config», який використовуються для налагодження параметрів конфігурації перед відправкою топології. Ця опція конфігурації об’єднана з конфігірацією кластера в ході виконання и відправки всіх задач. Як тільки топологія передасться в кластер, протягом 10 секунд кластер обчислить дану топологію.

Як тільки запуститься додаток, він виведе всю інформацію про процес запуску кластера та обробку інформації. Ось як буде відображатись інформація в консолі:

```
1234123402 - 1234123401 : 78
1234123402 - 1234123404 : 88
1234123402 - 1234123403 : 105
1234123401 - 1234123404 : 74
1234123401 - 1234123403 : 81
1234123401 - 1234123402 : 81
1234123403 - 1234123404 : 86
1234123404 - 1234123401 : 63
1234123404 - 1234123402 : 82
1234123403 - 1234123402 : 83
1234123404 - 1234123403 : 86
```

Висновки та перспективи подальших пошуків у напрямі дослідження. В процесі дослідження було розглянуто основні поняття, кластерну

архітектуру, задачі центру обробки та аналізу великих потоків даних мобільних операторів, а також представлено роботу додатку за допомогою середовища розробки Apache Storm. Було досліджено, що Storm буде відмінним вибором для реалізації високошвидкісної системи обробки подій, що забезпечує інкрементні обчислення для операторів мобільного зв'яку на прикладі журналу викликів. Однак, якщо в рамках Storm необхідно забезпечити збереження стану (stateful) і в точності одноразову доставку повідомлень (exactly once), слід використовувати Trident API, який дозволяє працювати з мікропакетами, як Apache Spark.

Список літератури:

1. Електронний ресурс: https://storm.apache.org/index.html.
2. Електронний ресурс: https://jansipke.nl/storm-in-pictures/
3. Anand Nalya, Ankit Jain Learning Storm 2014. [Електронний ресурс]. Режим доступу : https://www.oreilly.com/library/view/learning-storm/9781783981328/
4. Jonathan Leibiusky, Gabriel Eisbrucb, Dario Simonassi Getting Started with Storm 1 видання 2012.
5. Sean T. Allen, Matthew Jankowski, Peter Pathirana Storm Applied 1 видання 2015. [Електронний ресурс]. Режим доступу : https://github.com/clojurians-org/storm-ebook/blob/master/ Storm%20Applied%20%20Strategies%20for%20real-time%20event%20processing.pdf
6. Максименко В.Н., Филиппов А.А. Центр обработки данных в структуре системы управления качеством оператора сотовой связи. T-Comm #6-2008 С.47-51. [Електронний ресурс]. Режим доступу : https://cyberleninka.ru/article/n/tsentr-obrabotki-dannyh-v-strukture-sistemy-upravleniya-kachestvom-operatora-sotovoy-svyazi/viewer
7. Дэви Силен, Арно Мейсман. Основы Data Science и Big Data. Python и наука о данных. 2017.