ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ СУЧАСНИХ ОРКЕСТРАТОРІВ ЗАВДАНЬ, ДЛЯ РОБОТИ ІЗ ЗАДАЧАМИ ОБРОБКИ ВЕЛИКИХ ДАНИХ

Зініч О. Є.
Науковий керівник – к.т.н., доцент Чуприна А.С.
Харківський національний університет радіоелектроніки
(61166, Харків, пр. Науки, 14 каф. ПІ, тел. (057) 702-14-46)
e-mail: oleksandr.zinich.cpe@nure.ua

The purpose of this work is to overview and analyze main technologies that are used to orchestrate tasks in process of building Big Data processing pipelines, their advantages and disadvantages in order to propose the most efficient and flexible ways to organize them, monitor, measure and make them fault tolerant and idempotent.

У сучасному світі інформації, яка передається в інтернеті стає усе більше. Проходить все більше фінансових транзакцій та здійснюється кількість покупок в онлайн магазинах. За для таргетінгу реклами під кожного окремого користувача збираються усі його дії, а саме: кліки, прокручування екрану, та переходи за посиланнями. Жодна сучасна компанія, яка хоче розвиватись та розширюватись не може обійтись без збору та аналізу даних з яких буде робити висновки.

Для організації роботи з цим незчисленним об’ємом джерел даних, побудови їх обробки та подальшої автоматизації візуалізацій використовуються спеціальні застосунки, які називаються оркестраторами завдань.

Прикладом стандартного процесу, також званого пайплайном, в обробці даних є наступне. До сховища надходить один чи декілька файлів, які необхідно обробити. Як тільки вони надійшли – необхідно перевірити їх цілісність, запустити програму, яка обробить їх та запишіть результат до спеціалізованого сховища, буде це база даних, інше місце у файловій системі, тощо, після чого перевірити що дані записались коректно, кількість співпадає та не було помилок під час читання чи запису. Далі необхідно перенести початкові файли до архіву та надіслати звіт, на який чекають інші системи чи кінцеві користувачі, наприклад у вигляді електронного листа.

На кожному етапі може статись помилка, яка може не залежати від якості коду, як то блокування у базі даних, непрацюючий сервіс відправки електронних листів, тощо. У цьому разі необхідно розуміти який саме етап має помилку та перезапустити виключно його, або навіть пропустити, бо він не є суттєвим на даний час.

Історично для цього використовували самостійно написані скрипти, на мовах shell, python, тощо, які не мали користувацького інтерфейсу, не мали можливості перезапускати окремі частини самостійно, тощо, що було

дуже не зручно для звичайного користувача. Подібні застосунки могли бути написані самостійно, але це вимагало окремої команди розробки та підтримки, тестування цього застосунку, налаштування та інше. Через усі перелічені фактори більшість компаній не могли собі дозволити створення подібних застосунків.

Зараз існує багато варіантів оркестраторів, як з відкритим кодом, які потрібно розгортувати самостійно, так і у вигляді server-less застосунків із закритим кодом. Серед найпопулярніших можна виділити Apache Airflow, застосунок з відкритим кодом, який існує також і в server-less варіанті, Apache Luigi, розробка компанії Spotify, що також має відкритий код та можливість розробки власних модулів, та власні закриті варіанти від основних постачальників хмарних сервісів – Google Workflow, Amazon Step Functions та Azure Data Factory.

Усі вони надають можливість планування запусків чи запуск від подій, запис логів, взаємодію із різними сервісами та можливість створення складних пайплайнів з різними рівнями залежності, помилкостійкості та автоматичних перезапусків.

Суттєво їх можна розділити на дві групи: застосунки із відкритим кодом, та застосунки із закритим кодом.

До перших відносяться Apache Airflow та Apache Luigi. Плюсом є незалежність від постачальника хмарних послуг, можливість створення власних розширень, можливості перевизначать деякі компоненти та через це робити більш гнучкі та складні пайплайни. Мінусом є потрібність в експертизі при роботі з ними, бо є більше шансів зробити щось не так, та у варіанті не server-less – потрібність у людини з навичками DevOps.

Серед другого типу є сервіси від Amazon, Google та Azure. Перевагами цих сервісів є підтримка від великої компанії, відсутність необхідності DevOps або самостійного налаштування, підтримки та контролю серверів, та більш сильна інтеграція з іншими сервісами постачальника. Недоліками є залежність від постачальника та менша гнучкість при розробці.

Список використаних джерел:

1. Braille character recognition based on neural networks / [K. Smelyakov, A. Chuprina, D. Yeremenko та ін.]. // 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). – 2018. – С. 509– 513.

2. Gradational Correction Models Efficiency Analysis of Low-Light Digital Image / [K. Smelyakov, A. Chupryna, M. Hvozdiev та ін.]. // 2019 Open Conference of Electrical, Electronic and Information Sciences (eStream). – 2019. – С. 1–6.