REAL VAQTDA MA'LUMOTLAR OQIMINI VIZUALIZTSIYA QILISH FREYMVORKINI YARATISH

Hotamov F.N.
hotamov_farhod@tuneconsulting.uz
Tune Consulting LLC, Tashkent, Uzbekistan
https://doi.org/10.5281/zenodo.17374463

Annotatsiya

So‘nggi yillarda streaming data analytics sohasida real-time vizual tahlilga bo‘lgan talab keskin oshdi.

Fintech, IoT, transport va kiberxavfsizlik tizimlarida soniyasiga millionlab hodisalar (events) hosil bo‘ladi, lekin ularni vizual shaklda tezkor va intuitiv tarzda ko‘rsatish — arxitekturaviy va algoritmik muammo bo‘lib qolmoqda.

An’anaviy BI vositalar (Tableau, PowerBI) real-time oqimlarni ishlov berishda kechikish (latency) muammosiga duch keladi. Shu bois event-driven visualization framework — ya’ni ma’lumot oqimlariga mos ravishda avtomatik yangilanadigan, past kechikishli va o‘lchamlanadigan tizim yaratish zarur.

Ushbu muammolarni hal etish uchun biz Apache Kafka, ClickHouse va Java asosidagi aggregator, shuningdek React frontendni birlashtirgan reaktiv arxitekturani taqdim etamiz.

Freymvork ikki asosiy yangilikni o‘z ichiga oladi:

1. **Delta-rendering algoritmi** — faqat o‘zgargan ma’lumot segmentlarini yangilaydi;
2. **Voqealarni ustuvorlashtirish modeli (event-prioritization model)** — muhim voqealarni minimal kechikish bilan ekranga chiqarishni ta’minlaydi.

Tajriba natijalari shuni ko‘rsatdiki, taklif etilgan tizim an’anaviy polling asosidagi tizimlarga nisbatan vizualizatsiya kechikishini 65% gacha kamaytiradi va sekundiga 100 000 dan ortiq voqeani qayta ishlash imkonini beradi.

Mazkur yondashuv moliyaviy monitoring, IoT tahlili va xavfsizlik paneli (dashboard) kabi sohalarda qo‘llanishi mumkin bo‘lgan masshtablanadigan, modulli va moslashuvchan vizualizatsiya pipelineni taqdim etadi.

Kalit so‘zlar: Big Data Visualization, Event-Driven Architecture, Streaming Analytics, ClickHouse, Java, React, Real-time Dashboard, Low Latency

Abstract

In recent years, the demand for real-time visual analysis in the field of streaming data analytics has increased sharply. In fintech, IoT, transportation, and cybersecurity systems, millions of events are generated every second; however, displaying them quickly and intuitively in a visual form remains both an architectural and algorithmic challenge. Traditional BI tools (such as Tableau and Power BI) face latency issues when processing real-time data streams. Therefore, it is necessary to develop an event-driven visualization framework — a low-latency and scalable system that automatically updates in accordance with incoming data streams.

To address these issues, we present a reactive architecture combining Apache Kafka, ClickHouse, and a Java-based aggregator with a React-based frontend.

The framework introduces two key innovations:

(1) a delta-rendering algorithm that updates only changed data segments, and

(2) an event-prioritization model ensuring critical events are displayed with minimal delay.

Experimental results demonstrate that the proposed system reduces visualization latency by up to 65% compared to traditional polling-based systems, while sustaining a throughput of over 100,000 events per second. This approach provides a scalable, modular, and adaptive visualization pipeline applicable to financial monitoring, IoT analytics, and security dashboards.

Keywords: Big Data Visualization, Event-Driven Architecture, Streaming Analytics, ClickHouse, Java, React, Real-time Dashboard, Low Latency

Maqsad va Vazifalar

Asosiy maqsad — Event-driven yondashuv asosida real-time big data visualization framework yaratish.

Asosiy vazifalar:

1. Eventlarni oqim tarzida qabul qilish (Apache Kafka).
2. Java (Spring WebFlux) yordamida asinxron event aggregator yaratish.
3. ClickHouse'da ma'lumotlarni real-time saqlash va indekslash.
4. React frontend'da incremental rendering (delta updates) mexanizmini ishlab chiqish.
5. Event-priority modeli orqali muhim hodisalarni ajratib ko'rsatish.

Metodologiya

1. Ma'lumotlar to'plamini tayyorlash

1 – jadval. 1 – jadval. Ma'lumotlar to'plamini tayyorlash.

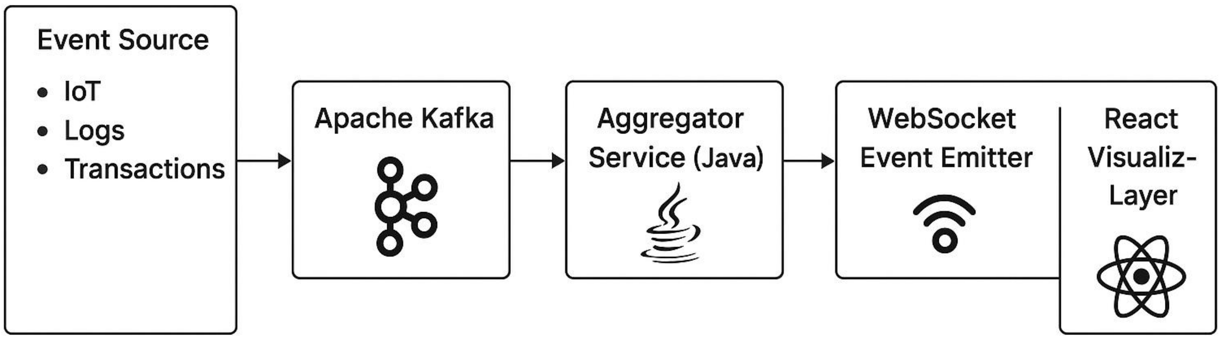
Bosqich	**Amal**	**Texnologiya**
1	Event oqimini yaratish	Apache Kafka
2	Event'larni yig'ish va tahlil qilish	Spring WebFlux (Java Reactive Streams)
3	Ma'lumotlarni saqlash	ClickHouse (Kafka engine, materialized view)
4	Event asosida vizual yangilanish	WebSocket API + React
5	Delta rendering mexanizmi	Custom D3.js components
6	Sinov va benchmarking	JMeter, Grafana

2. Model arxitekturasi.

Tizim komponentlari:

- Event Source (IoT, Logs, Transactions)
- Apache Kafka – event buffering va queue.
- Aggregator Service (Java) – eventlarni birlashtirish, filtrlash, delta hisoblash.
- ClickHouse – OLAP saqlash.
- WebSocket Event Emitter – frontend'ga push.
- React Visualization Layer – interaktiv grafikalar, real-time yangilanish.

1 – rasm. Tizim komponentlari.



3. Asosiy ishlash jarayoni

• **Hodisalarni qabul qilish** — Ma’lumotlar turli manbalardan (transaksiyalar, loglar, IoT signallar va boshqalar) oqim shaklida keladi. Har bir hodisa Apache Kafka orqali real vaqt rejimida qabul qilinadi va tegishli “topic”larga yuboriladi. Kafka bu yerda bufer va xabar vositachisi (message broker) sifatida ishlaydi — u ishlab chiqaruvchi (producer) va iste’molchi (consumer) komponentlarni bir-biridan ajratadi hamda uzatishni ishonchli qiladi.

• **Agregatsiya va transformatsiya** — java asosida ishlab chiqilgan Aggregator Service (Spring WebFlux) Kafka oqimlariga obuna bo‘ladi va ma’lumotlarni reaktiv oqim (non-blocking reactive stream) tarzida qabul qiladi.

Har bir hodisa asinxron tarzda qayta ishlanadi, filtrlab chiqiladi va qo‘shimcha ma’lumot (masalan, vaqt, foydalanuvchi identifikatori, joylashuv) bilan boyitiladi. Agregatsiya algoritmi hodisalarni vaqt oynasi yoki muayyan kalit (masalan, userId, sessionId) asosida guruhlaydi. Shu jarayonda delta ma’lumotlar (ya’ni faqat o‘zgargan yoki yangi kelgan hodisalar) aniqlanadi va bu vizual interfeysni yangilashda ortiqcha resurs sarfini kamaytiradi.

• **Ma’lumotlarni saqlash va materializatsiya** Qayta ishlangan (aggregated) va xom (raw) ma’lumotlar ClickHouse ma’lumotlar bazasiga yoziladi.

ClickHouse’ning Kafka Engine moduli real vaqt rejimida ma’lumotlarni qabul qiladi, Materialized View esa avtomatik agregatsiyani amalga oshiradi. Shu tarzda tayyorlangan ma’lumotlar vizual interfeysga millisekund ichida yetkazilishi mumkin bo‘ladi, hatto katta hajmdagi oqimlarda ham.

• **Hodisalarga asoslangan aloqa qatlami (Event-Driven Communication Layer)** — Agregatsiyadan so‘ng ma’lumotlar **WebSocket API** orqali frontend’ga yuboriladi. Bu tizimda an’anaviy “polling” o‘rniga push-notifikatsiya mexanizmi ishlatiladi, bu esa tarmoq yuklamasini kamaytiradi va kechikishni minimallashtiradi. Muhim hodisalar (masalan, ogohlantirishlar yoki anomaliyalar) prioritet navbat orqali yuqori ustuvorlik bilan yuboriladi.

• **Real-time vizualizatsiya va delta-renderlash** — Frontend tomonda React asosidagi vizual interfeys WebSocket oqimlarini qabul qiladi. Grafik komponentlar Recharts yoki D3.js yordamida quriladi. Delta-rendering algoritmi faqat o‘zgargan elementlarni yangilaydi, butun grafikani qayta chizmaydi. Natijada, interfeys tezkor, silliq va resurs jihatdan tejamkor bo‘ladi.

• **Monitoring va teskari aloqa** — Tizimning asosiy ko‘rsatkichlari (latency, throughput, CPU yuklamasi) Prometheus orqali yig‘iladi va Grafanada vizuallashtiriladi.Monitoring ma’lumotlari asosida tizim avtomatik tarzda agregatsiya oynasi, bufer hajmi yoki prioritet darajasini dinamik o‘zgartiradi. Bu yechim tizimni o‘zgaruvchan yuklamalarda ham barqaror ishlashini ta’minlaydi.

4. Ilmiy yangilik va amaliy ahamiyat

• Event-driven visual architecture — foydalanuvchi interfeysi har bir event asosida o‘zgaradi.

• Delta rendering algoritmi — faqat o‘zgargan elementlar qayta chiziladi.

• Priority-based event queuing — muhim ma’lumotlar ustuvor tarzda uzatiladi.

• Hybrid event aggregation model — ClickHouse materialized views bilan birga ishlovchi real-time aggregator.

Xulosa

Ushbu tadqiqotda real vaqt rejimida oqimli ma’lumotlarni vizuallashtirish muammosi tahlil qilindi va event-driven yondashuvga asoslangan yangi arxitektura taklif etildi. Taklif etilgan framework oqim ma’lumotlarini Apache Kafka orqali qabul qiladi, Java Aggregator Service yordamida ularni filtrlab, agregatsiya qiladi va ClickHouse bazasida saqlaydi. Natijalar WebSocket orqali React asosidagi vizual interfeysga uzatiladi.

Tizimning asosiy ustunliklari quyidagilardan iborat:

1. Past kechikish (low latency) — har bir yangilanish real vaqt rejimida foydalanuvchiga yetkaziladi.

2. O‘lchamlanish (scalability) — Kafka va ClickHouse’ning taqsimlangan tabiati katta hajmdagi ma’lumotlar bilan ishlash imkonini beradi.

3. Delta-rendering — faqat o‘zgargan ma’lumotlarni qayta chizish orqali brauzer yuklamasi kamayadi.

4. Moslashuvchanlik (adaptivity) — tizim tarmoq va yuklama sharoitlariga qarab o‘z ish parametrlarini avtomatik o‘zgartiradi.

Shuningdek, framework fintech, IoT, transport monitoringi va kiberxavfsizlik tizimlari uchun real-time tahlilni samarali ko‘rinishda taqdim etish imkonini beradi. Kelgusida, tizimga anomal detection, machine learning asosidagi prediktiv `rivojlantirish rejalashtirilgan.

Natijada, ishlab chiqilgan Event-Driven Visualization Framework oqimli ma’lumotlarni tahlil qilishda reaktiv, ishonchli va vizual jihatdan samarali yechim sifatida e’tirof etilishi mumkin.

Adabiyotlar, References, Литературы:

1. Robert Schulze, Tom Schreiber, Ilya Yatsishin, Ryadh Dahimene, Alexey Milovidov, ClickHouse – Lightning Fast Analytics for Everyone, PVLDB, Vol. 17, No. 12, 2024. VLDB
2. Adrian Göransson, Oskar Wändesjö, Evaluating ClickHouse as a Big Data Processing Solution for IoT-Telemetry, Master’s Thesis, Lund University, 2022. timestored.com
3. Kexin Rong, Peter Bailis, ASAP: Prioritizing Attention via Time Series Smoothing, arXiv preprint, 2017. arXiv
4. Yifan Wu, Remco Chang, Joseph Hellerstein, Arvind Satyanarayan, Eugene Wu, DIEL: Interactive Visualization Beyond the Here and Now, arXiv preprint, 2019. arXiv
5. Piyush Yadav, Dhaval Salwala, Edward Curry, Knowledge Graph Driven Approach to Represent Video Streams for Spatiotemporal Event Pattern Matching in Complex Event Processing, arXiv preprint, 2020. arXiv
6. “Streaming Analytics: Intro, Tools & Use Cases”, Confluent — veb maqola. Confluent
7. “What is real-time analytics? | ClickHouse Engineering Resources” — veb manba. ClickHouse

8. “Planning Your Data Architecture for Streaming Analytics”, DataLere — veb maqola. Datalere
9. “Best practices for monitoring event-driven architectures”, Datadog blog. Datadog
10. “How Stream Processing Makes Your Event-Driven Architecture Even Better”. DEV Community