УДК 004.75

Архитектурные подходы к обработке потоковых данных в облачных средах: систематизация, эволюция паттернов и перспективы интеграции с интеллектуальными системами

Темерсултанова П.В. студентка группы ЗИСТ-25м, 1 курс
ФГБОУ ВО «Грозненский государственный нефтяной технический университет имени академика М.Д. Миллионщикова»
г. Грозный

Аннотация

В статье рассматриваются архитектурные подходы к обработке потоковых данных в облачных средах, а также определяется вектор развития современных систем. Рассматривается эволюция архитектурных паттернов — от классических Lambda и Kappa до гибридных решений и событийно-ориентированных архитектур (EDA - Event-Driven Architecture), интегрирующих возможности машинного обучения. Выявлено направление развития к унификации пакетной и потоковой обработки в архитектуре Lakehouse. Проведен анализ преимуществ применения оконных функций и управления состоянием для обеспечения точности вычислений. Сравниваются платформы Apache Kafka, Apache Flink в реализации масштабируемых конвейеров данных. Главные выводы свидетельствуют о том, что современные архитектуры эволюционируют в сторону интеллектуальной оркестрации, контейнеризации и микросервисной декомпозиции. Практическая значимость работы состоит в предоставлении основы для выбора архитектурных решений в зависимости от требований бизнес-задач и характеристик облачной инфраструктуры.

Ключевые слова: потоковая обработка данных, архитектура Lambda, событийно-ориентированная архитектура (EDA), облачные вычисления, Apache Kafka, Apache Flink, масштабируемость, обработка в реальном времени, Lakehouse, интернет вещей (IoT - Internet of Things).

Введение

В современную цифровую эпоху наблюдается экспоненциальный рост объемов данных, генерируемых устройствами интернета вещей, промышленными сенсорами, банковскими транзакциями и социальными сетями. Эта информация поступает непрерывно и с высокой скоростью, образуя потоки данных, ценность которых максимальна именно в момент их возникновения. Традиционные методы пакетной обработки, предполагающие накопление и последующий анализ данных, не успевают за требованиями времени, уступая место системам, способным обрабатывать информацию в реальном времени [1]. Потребность в немедленном выявлении аномалий, прогнозировании состояний и

оперативном реагировании на события обусловила стремительное развитие архитектурных подходов к обработке потоковых данных, особенно в гибкой и масштабируемой среде облачных вычислений [2].

Вопросы организации потоковой обработки получили широкое освещение. Фундаментальные различия между пакетной и потоковой семантикой, а также связанные с этим сложности управления состоянием и обработки поздно поступающих данных, подробно рассматриваются в технологических руководствах [3]. Карпович М.Н. акцентирует внимание на особенностях проектирования микросервисно-событийных архитектур (EDA) для высоконагруженных систем, что перекликается с современными подходами к построению гибких и устойчивых конвейеров данных [4]. В свою очередь, Шибанов С.В. с соавторами подчеркивает важность управления метаданными в сервисах потоковой обработки, предлагая метамодели для описания событий и активных правил [5]. Проблемы управления данными и обеспечения их качества в облачных средах, включая потоковые сценарии, поднимаются в работах, размещенных на платформе CyberLeninka [6, 7]. Отдельное направление связано с применением потоковой обработки в специализированных областях, таких как выявление аномалий в реальном времени [8] или обеспечение качества больших языковых моделей (LLM - Large Language Model) [9].

Несмотря на обилие публикаций, посвященных отдельным инструментам и задачам, существует определенный пробел в комплексных исследованиях, рассматривающих эволюцию архитектурных подходов в целом. Часто вне фокуса остаются вопросы взаимосвязи классических паттернов (Lambda, Kappa) с новыми трендами, такими как унификация пакетной и потоковой обработки в озерах данных Lakehouse [10] или интеграция с системами искусственного интеллекта и машинного обучения для предиктивной оптимизации самих потоковых процессов [11].

Актуальность данного исследования обусловлена необходимостью преодоления этого пробела и формирования целостного, систематизированного представления о современных архитектурных подходах к обработке потоковых данных в облаке.

Целью данной статьи является систематизация и сравнительный анализ эволюции архитектурных подходов к потоковой обработке данных в облачных средах и определение ключевых тенденций их развития для обеспечения масштабируемости, отказоустойчивости и эффективности.

Основная часть

1. Методология исследования

Исследование носит теоретико-аналитический характер и базируется на общепризнанных методах научного познания, адаптированных для анализа сложных распределенных систем. Основу методологии составили: сбор и первичная систематизация данных, сравнительный архитектурный анализ и синтез выявленных тенденций.

Был проведен сбор и критический анализ открытых источников информации. Критериями отбора материалов выступали их релевантность теме, авторитетность источника и актуальность. Для классификации архитектурных подходов и платформ применялся метод таксономии, позволивший выделить ключевые классификационные признаки: семантика обработки, архитектурный паттерн, способ управления состоянием и гарантии доставки сообщений. Для сравнительной характеристики платформ (Apache Kafka, Apache Flink, Apache Spark Structured Streaming) использовался метод сопоставительного анализа по таким критериям, как задержка обработки, модели вычислений, механизмы обеспечения отказоустойчивости и возможности интеграции с облачной инфраструктурой [2]. Это позволило выявить сильные и слабые стороны каждого подхода в контексте различных классов задач. Для синтеза полученных данных и формирования целостной картины эволюции архитектур, использовался метод абстрагирования для перехода от описания конкретных технологий к обобщенным архитектурным паттернам. На основе анализа трендов были сформулированы прогнозы дальнейшего развития исследуемой области.

2. Результаты исследования

Рассматривая эволюцию архитектурных паттернов, следует отметить, что все многообразие существующих решений может быть классифицировано на основе фундаментальных параметров: требований к задержке обработки и сложности реализации. К числу классических паттернов относится Lambda-архитектура, которая сочетает параллельные пакетный и потоковый конвейеры для обеспечения одновременно точности и оперативности обработки. Данный подход сохраняет актуальность для систем, где критически важна корректность пересчета исторических данных, например, в финансовой сфере или биллинговых системах [14]. Однако существенным недостатком Lambda-архитектуры остается высокая сложность разработки и эксплуатации, обусловленная необходимостью поддержки двух различных кодовых баз. Альтернативой выступает Kappa-архитектура, которая упрощает эту модель, используя единый потоковый движок для всех задач, включая пересчеты. Это достигается за счет воспроизведения потока событий из долговременного хранилища, что делает архитектуру Kappa предпочтительной для сценариев, где допустима небольшая задержка при пересчете и важна операционная простота [15].

Среди современных эволюционных форм отчетливо прослеживается тренд к унификации пакетной и потоковой обработки в рамках единой платформы, реализованный в архитектуре Lakehouse, например, Delta Lake, Apache Iceberg [10]. Использование метаданных и транзакционных журналов позволяет применять потоковые фреймворки, такие как Structured Streaming в Spark, для инкрементальной обработки данных, хранящихся в озерах данных. Это фактически стирает грань между двумя исторически сложившимися парадигмами обработки информации.

Важным аспектом исследования выступает анализ роли платформенных решений, поскольку выбор конкретной платформы потоковой обработки критически зависит от требований решаемой задачи. Apache Kafka зарекомендовала себя как высоконадежная, масштабируемая шина данных, обеспечивающая гарантированную доставку и долговременное хранение потока. Ее роль как центрального нервного узла в событийно-ориентированных архитектурах является общепризнанной [13]. В свою очередь, Apache Flink выделяется как лидер в области истинной потоковой обработки с минимальной задержкой и богатыми возможностями для управления состоянием и обработки событийного времени [16]. Благодаря этим характеристикам Flink идеально подходит для задач, требующих сложной аналитики в реальном времени, например, для выявления мошеннических транзакций. Библиотека структурированной потоковой передачи в Apache Spark, использующая микропакетную модель, предлагает компромисс между задержкой и пропускной способностью, а также обеспечивает бесшовную интеграцию с экосистемой Spark для пакетной обработки и машинного обучения, что делает его удобным выбором для создания унифицированных платформ данных [17].

Проведенное исследование позволило выявить несколько макротрендов, определяющих будущее архитектур потоковой обработки. Первый и наиболее значимый тренд связан с интеграцией потоковой обработки и искусственного интеллекта. Потоковая обработка становится технологической основой для систем, работающих с большими языковыми моделями (LLM). Она применяется для динамического обогащения контекста запроса в реальном времени и для непрерывного мониторинга качества ответов модели, включая отслеживание галлюцинаций, релевантности и использования токенов [9]. Реализация таких сценариев требует тесной интеграции потоковых платформ с векторными базами данных и фреймворками MLOps.

Второй тренд связан с эволюцией событийно-ориентированных архитектур (EDA), которые переходят от простой асинхронной коммуникации к более интеллектуальным формам взаимодействия. Применение потоковых платформ в сочетании с контейнеризацией (Docker, Kubernetes) позволяет создавать высокодецентрализованные, самоуправляемые системы, где микросервисы реагируют на события независимо друг от друга [4]. Такой подход повышает отказоустойчивость и упрощает масштабирование сложных распределенных приложений.

Третий тренд отражает усложнение управления метаданными и состоянием. Современные системы обработки потоков требуют развитых механизмов для работы с состоянием вычислений. Это включает не только встроенные механизмы чекпоинтов (checkpointing) в Apache Flink, но и внешние системы для управления метаданными о потоках, схемах данных и происхождении данных [5]. Развитие этих механизмов критически важно для обеспечения согласованности и возможности аудита в сложных композитных архитектурах.

Четвертый тренд обусловлен необходимостью оптимизации ресурсов в облачной среде. Использование облачных сред диктует потребность в более гибком и экономичном управлении ресурсами. Наблюдается тенденция к применению алгоритмов машинного обучения для предиктивного автоскалинга (автоматического масштабирования) потоковых кластеров на основе анализа метрик нагрузки, что позволяет минимизировать расходы при сохранении требуемой производительности [11].

Таким образом, совокупность рассмотренных результатов свидетельствует о переходе от статических, жестко заданных архитектур к динамическим, самонастраивающимся системам, способным адаптироваться к изменяющимся условиям нагрузки и требованиям бизнеса.

Заключение

Проведенное исследование, посвященное анализу архитектурных подходов к обработке потоковых данных в облачных средах, позволяет сформулировать следующие основные выводы и результаты.

Во-первых, в работе систематизирована эволюция архитектурных паттернов обработки потоковых данных. Выявлено последовательное движение от сложных и многослойных решений, представленных Lambda-архитектурой, к более унифицированным и технологичным моделям. Kappa-архитектура и, в особенности, emerging -архитектура Lakehouse демонстрируют отчетливую тенденцию к консолидации пакетной и потоковой обработки на единой платформе. Такая консолидация позволяет существенно упростить эксплуатацию систем данных и снизить совокупную стоимость владения.

Во-вторых, в ходе исследования определена ключевая роль специализированных платформ потоковой обработки, которые выступают фундаментальными строительными блоками современных data-платформ. Проведенный анализ показал четкое разделение ниш между основными платформами: Apache Kafka функционирует как надежная шина данных и основа событийно-ориентированных архитектур; Apache Flink является отраслевым стандартом для низкозатратной обработки с сохранением состояния; Apache Spark Structured Streaming предлагает универсальное решение для унифицированной аналитики. Выбор конкретной платформы должен определяться требованиями к допустимым задержкам, сложности вычислений и необходимостью интеграции с существующей экосистемой.

В-третьих, выявлены ключевые тенденции, определяющие дальнейшее развитие рассматриваемой предметной области. Наиболее значимой из них является глубокая интеграция потоковой обработки с системами искусственного интеллекта, что проявляется как в использовании потоков для оперативного обогащения контекста больших языковых моделей, так и в применении методов машинного обучения для предиктивной оптимизации самих потоковых инфраструктур в облаке. Одновременно наблюдается усложнение механизмов

управления метаданными и состоянием, необходимое для обеспечения надежности и согласованности в распределенных событийно-ориентированных архитектурах.

Теоретическая значимость работы заключается в уточнении классификации современных архитектурных подходов и выявлении закономерностей их эволюции под влиянием новых технологий, включая искусственный интеллект и концепцию Lakehouse. Практическая значимость состоит в создании целостной картины, которая может служить методологической основой для выбора технологического стека и проектирования масштабируемых, отказоустойчивых и экономически эффективных систем обработки данных в реальном времени.

Прогнозируя дальнейшее развитие исследуемой области, следует ожидать усиления роли интеллектуальной оркестрации потоков и окончательного стирания границ не только между пакетной и потоковой обработкой, но и между транзакционными и аналитическими системами. На основе полученных результатов можно рекомендовать при проектировании новых систем данных изначально рассматривать возможность использования унифицированных платформ (Lakehouse) и событийно-ориентированных подходов (EDA) как наиболее гибких и перспективных. Перспективным направлением для будущих исследований является разработка формализованных методов оценки совокупной стоимости владения для различных архитектурных паттернов в публичных и гибридных облаках, а также создание эталонных реализаций (reference architectures) для типовых отраслевых задач в таких сферах, как финансы, телекоммуникации и промышленный интернет вещей.

Список литературных источников

1. Потоковая передача данных: основные концепции и возможности [электронный ресурс] // Хабр. URL: https://habr.com/ru/companies/ruvds/articles/500028/ (дата обращения: 11.02.2026).
2. Сюй Явэнь, Цзинь Чжи Использование ТЕХНОЛОГИИ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ В КОМПЬЮТЕРНОМ АНАЛИЗЕ БОЛЬШИХ ДАННЫХ // Международный журнал гуманитарных и естественных наук. 2024. №9-5 (96). URL: https://cyberleninka.ru/article/n/ispolzovanie-tehnologii-oblachnyh-vychisleniy-v-kompyuternom-analize-bolshih-dannyh (дата обращения: 13.02.2026).
3. Намир Хашим Касим, Наталия Боднар, Хайдер Махмуд Салман, Салама Идрис Мустафа, Фахер Рахим Проблемы управления данными и решения в облачных средах // РЭНСИТ. 2024. №1. URL: https://cyberleninka.ru/article/n/problemy-upravleniya-dannymi-i-resheniya-v-oblachnyh-sredah (дата обращения: 13.02.2026).
4. Карпович, М. Н. Особенности проектирования микросервисно-событийных архитектур для высоконагруженных распределенных систем обработки информаций / М. Н. Карпович // Труды БГТУ. Серия 3, Физико-

математические науки и информатика. — 2023. — № 1 (266). — С. 89-95. — DOI: 10.52065/2520-6141-2023-266-1-15.

5. Шибанов, С. В. Архитектура метаданных сервиса потоковой обработки событий и исполнения активных правил / С. В. Шибанов, А. С. Гусаров, Я. С. Шлепнев // Вестник Пензенского государственного университета. — 2023. — № 3. — С. 95-103.
6. Пальмов Сергей Вадимович, Кабирова Дайана Фаридовна, Добролюбова Ксения Сергеевна Облачные технологии для обработки больших данных: преимущества и ограничения // Индустриальная экономика. 2025. №7. URL: https://cyberleninka.ru/article/n/oblachnye-tehnologii-dlya-obrabotki-bolshih-dannyh-preimuschestva-i-ogranicheniya. (дата обращения: 14.02.2026).
7. Утегенов Нұрдәулет Бауржанұлы ИНТЕРНЕТ ВЕЩЕЙ (IOT) И ИНФОРМАЦИОННЫЕ СИСТЕМЫ // Universum: технические науки. 2023. №7-1 (112). URL: https://cyberleninka.ru/article/n/internet-veschey-iot-i-informatsionnye-sistemy (дата обращения: 15.02.2026).
8. Савицкий Д. Е., Дунаев М. Е., Зайцев К. С. ВЫЯВЛЕНИЕ АНОМАЛИЙ ПРИ ОБРАБОТКЕ ПОТОКОВЫХ ДАННЫХ В РЕАЛЬНОМ ВРЕМЕНИ // International Journal of Open Information Technologies. 2022. №6. URL: https://cyberleninka.ru/article/n/vyyavlenie-anomaliy-pri-obrabotke-potokovyh-dannyh-v-realnom-vremeni (дата обращения: 16.02.2026).
9. Потоковая обработка данных и EDA-архитектура для LLM-систем [электронный ресурс] // Big Data School. URL: https://bigdataschool.ru/blog/news/machine-learning/eda-and-stream-processing-for-llm/ (дата обращения: 16.02.2026).
10. Унификация пакетной и потоковой обработки в Delta-архитектуре [электронный ресурс] // Big Data School. URL: https://bigdataschool.ru/blog/batch-and-streaming-processing-in-lakehouse/ (дата обращения: 16.02.2026).
11. Lakkireddy, S. HyScaleFlow: An ML-Driven DAG-Based Orchestration Framework for Real-Time Stream Processing in Hybrid Cloud Environments / S. Lakkireddy // Informatica: An International Journal of Computing and Informatics. — 2025. — Vol. 49, No. 9. — DOI: 10.31449/inf.v49i9.9498.
12. Hazelcast. Stream Processing Reference Guide [электронный ресурс] // Hazelcast. URL: https://hazelcast.com/resources/reference-guide-stream-processing/ (дата обращения: 18.02.2026).
13. Архитектурные паттерны потоков данных [электронный ресурс] // Datafinder. URL: https://datafinder.ru/products/arhitekturnye-patterny-potokov-dannyh (дата обращения: 18.02.2026).
14. Лямбда-архитектура: обработка больших данных [электронный ресурс] // Хабр. URL: https://habr.com/ru/companies/otus/articles/766672/ (дата обращения: 18.02.2026).

15. Kreps, J. Questioning the Lambda Architecture / J. Kreps // O'Reilly Radar. URL: https://www.oreilly.com/radar/questioning-the-lambda-architecture/ (дата обращения: 19.02.2026).
16. Carbone, P. Apache Flink™: Stream and Batch Processing in a Single Engine / P. Carbone et al. // IEEE Data Engineering Bulletin. — 2015. — Vol. 38, No. 4. — P. 28-38.
17. Zaharia, M. Apache Spark: A Unified Engine for Big Data Processing / M. Zaharia et al. // Communications of the ACM. — 2016. — Vol. 59, No. 11. — P. 56-65. — DOI: 10.1145/2934664.