实时特征湖在大规模风控中的落地：高并发写入与低时延召回

刘 超

Devz AI technologies Inc., America

【摘要】随着电子商务与金融业务的快速发展，实时风控系统对数据处理的时效性、一致性与可扩展性提出了更高要求。传统基于 Hive、Kafka 和 Flink 的流批分离架构在特征管理、数据更新与查询效率方面存在明显瓶颈。本文基于本人在多家互联网企业的大数据与风控平台的实战建设经验，提出并实践了一种基于 Apache Paimon 构建的实时特征湖架构，实现了高并发数据写入与低时延特征召回的平衡。该系统已在一些全球电商的风控场景中成功落地，支持日均千亿级事件处理，特征更新延迟降至秒级，查询性能提升 3 倍以上，为大规模实时风控提供了可复用的架构范式。本文详细阐述了系统架构设计、关键技术选型与优化策略，并对落地过程中的挑战与解决方案进行了深入总结。

【关键词】实时特征湖；风控系统；Apache Paimon；高并发写入；低时延查询；流批一体；特征工程

【收稿日期】2025 年 11 月 6 日 【出刊日期】2025 年 12 月 30 日 【DOI】10.12208/j.jer.20250394

Application of real-time feature lakes in large-scale risk control: high-concurrency writing and low-latency recall

Chao Liu

Devz AI technologies Inc., America

【**Abstract**】With the rapid development of e-commerce and financial services, real-time risk control systems have placed higher demands on data processing timeliness, consistency, and scalability. Traditional stream-batch separation architectures based on Hive, Kafka, and Flink exhibit significant bottlenecks in feature management, data updates, and query efficiency. Building on my hands-on experience in constructing big data and risk control platforms for multiple internet enterprises, this paper proposes and implements a real-time feature lake architecture based on Apache Paimon. This architecture achieves a balance between high-concurrency data writes and low-latency feature recall. Successfully deployed in risk control scenarios for global e-commerce platforms, the system supports daily processing of hundreds of billions of events, reduces feature update latency to seconds, and enhances query performance by over threefold, providing a reusable architectural paradigm for large-scale real-time risk control. The paper elaborates on system architecture design, key technology selection and optimization strategies, and provides an in-depth summary of challenges and solutions encountered during implementation.

【**Keywords**】Real-time feature lake; Risk control system; Apache Paimon; High-concurrency writes; Low-latency queries; Stream-batch integration; Feature engineering

1 引言

在互联网电商与广告业务中，风险控制是保障平台安全与用户体验的核心环节。传统风控系统多依赖离线特征与规则引擎，存在特征更新滞后、数据不一致、系统复杂度高等问题。随着业务实时化需求的提升，如何构建一个既能支持高并发数据写入，又能实现低延迟特征查询的实时特征平台，成为行业共同挑战，特别是在全球电商场景下，用户行为数据量巨大、访问模式多样，对底层数据平台的吞吐能力、查询性能和数据一致性提出了极高要求。

本文基于在多家互联网企业的大数据与风控平台建设经验，提出以 Apache Paimon 为核心构建实时特征湖，统一流批数据存储与处理链路，实现了风控场景下特征数据的实时更新、高效管理与低延迟服务。本文将从系统架构、关键技术、落地实践与效果评估四个方面展开论述，重点分享在高并发写入优化、低延迟查询

作者简介：刘超，研究方向：流式与批处理数据管道的设计与开发，Flink、Spark、Kafka 在内的主流数据处理框架。

加速以及生产环境稳定性保障等方面的工程实践经验。

2 实时特征湖的架构设计

2.1 传统风控数据架构的瓶颈

传统风控系统通常采用 Lambda 架构，流处理与批处理分离，存在以下典型问题：

1）特征更新延迟高：离线特征 T+1 更新，无法应对实时欺诈行为；

2）数据一致性难保障：流批数据来源不同，逻辑复杂，易出现数据偏差；

3）系统维护成本高：需维护两套计算与存储系统，开发与运维负担重；

4）特征回溯困难：缺乏统一的时空快照机制，难以支持事后分析与模型迭代；

5）资源利用率低：流批两套集群资源无法共享，存在资源浪费现象。

2.2 实时特征湖的整体架构

为解决上述问题，我设计了基于 Apache Paimon 的实时特征湖架构，如下图所示：

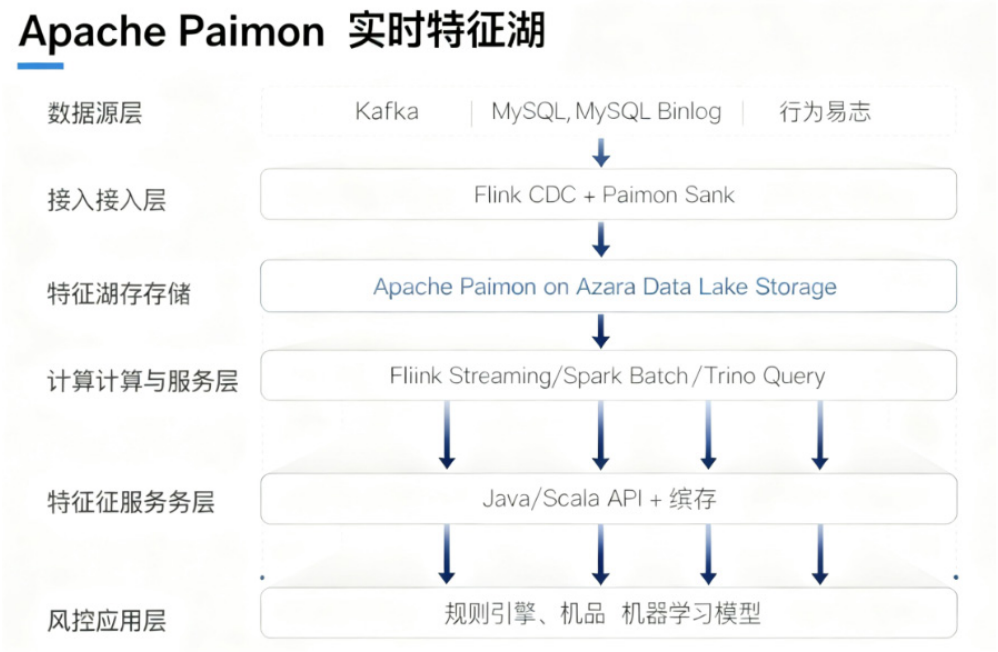

图 1

该架构具备以下特点：

1）统一存储：所有特征数据（原始数据、中间特征、模型结果）均存入 Paimon 表，支持流读流写、批量回溯、时间旅行等能力；

2）多引擎兼容：支持 Flink、Spark、Trino 等多种计算引擎，满足实时计算、离线分析与即席查询需求；

3）高并发写入：基于 LSM-Tree 的存储结构，优化随机写入性能，支持千亿级事件实时入库；

4）低延迟查询：通过主键索引、二级索引与文件局部性优化，实现毫秒级特征点查；

5）成本可控：采用 Azure Data Lake Storage Gen2 作为底层存储，相比传统 HDFS 降低成本 30%以上。

2.3 架构关键设计与取舍

在具体工程实现中，我遵循“短链路、强治理、易回溯”的三项原则完成取舍：其一，以表为中枢替代多层消息中间件，把中间结果固化到 Paimon 表，缩短排障半径并利于跨引擎对账；其二，同源快照贯通实时与离线，训练、评估与在线召回共享相同快照版本，避免样本偏移；其三，功能分层而非“成分分层”：数据湖承接统一持久化与快照治理，计算引擎承接特征加工与近实时聚合，服务层承接点查与轻聚合，避免在单层里叠加过多职责导致不稳定。针对多云与跨地域部署，我采用“元数据一致 + 存算分离”的方式，将元数据服务与对象存储解耦，降低扩容与迁移门槛。

3 关键技术实现

3.1 基于 Apache Paimon 的流批一体存储

Apache Paimon 作为一种新兴的湖仓格式，具备以下核心能力，非常适合作为实时特征湖的底层存储：

1）Streaming-First 设计：原生支持 Flink CDC 整库同步、增量更新、Changelog 生成，适合风控场景下用户行为、交易流水等动态数据的实时接入；

2）主键更新与部分列更新：支持 Upsert 操作，避

免全量覆盖，提升写入效率；

3）多版本管理与时间旅行：支持基于 Snapshot 的历史数据查询，便于特征回溯与模型训练；

4）开放格式与多引擎集成：底层为列式存储（ORC/Parquet），可与 Spark、Trino、StarRocks 等无缝对接。

在具体实践中，我通过 Flink SQL 创建 Paimon 表，并配置合理的分区和分桶策略。例如，用户行为表按 event_date 分区，按 user_id 分桶，确保数据分布均匀，避免热点问题。表结构定义如下：

```
CREATE TABLE user_behavior_features (
    user_id BIGINT,
    feature_name STRING,
    feature_value DOUBLE,
    event_time TIMESTAMP(3),
    PRIMARY KEY (user_id, feature_name) NOT ENFORCED
) PARTITIONED BY (event_date)
WITH (
    'bucket' = '10',
    'changelog-producer' = 'input',
    'merge-engine' = 'aggregation',
    'fields.feature_value.aggregate-function' = 'last_non_null_value'
    );
```

此配置实现了基于用户 ID 和特征名的主键更新，确保同一用户的同一特征只会保留最新值，同时支持流式读取变更日志。

3.2 高并发写入优化

风控场景下数据写入具有高吞吐、高并发的特点。我通过以下手段提升写入性能：

1）分区与分桶策略：按时间（天/小时）分区，按用户 ID 分桶，避免小文件问题，提升写入并行度。实际测试表明，合理的分桶数（通常为 CPU 核数的 1-2 倍）可将写入吞吐提升 40%以上；

2）LSM-Tree 压缩优化：调整 Compaction 策略，平衡写入放大与查询性能。我采用'compaction.max-file-num' = '5'和'compaction.target-file-size' = '128MB'的配置，在保证查询性能的同时控制 Compaction 频率；

3）异步 I/O 与批量提交：在 Flink Sink 中启用异步写入与批量提交机制，降低网络与存储 I/O 压力。通过调整 'sink.buffer-flush.max-rows' = '1000' 和 'sink.buffer-flush.interval' = '1s'，实现批量提交与低延迟的平衡；

4）内存调优：为 Flink TaskManager 分配充足的堆外内存，用于 Paimon 的写缓存，避免频繁的磁盘刷写。

在实际压测中，单表支持每秒百万级事件写入，日均千亿条记录入库，且写入延迟稳定在秒级。集群资源使用率相比原有 Kafka+Hive 架构提升约 25%。

3.3 低时延特征召回机制

特征服务的响应速度直接影响风控决策的时效性。我构建了多层查询加速机制：

1）主键点查优化：利用 Paimon 的 Primary Key 索引，实现用户维度的毫秒级特征提取。对于复合主键（如（user_id, feature_name）），Paimon 会构建多层索引，加速特定用户的特征查询；

2）本地缓存与预加载：在特征服务层集成 Redis 缓存，热点特征预加载，降低对底层存储的访问压力。我设计了分层缓存策略：L1 为本地 Caffeine 缓存（存储极热点数据），L2 为 Redis 集群缓存（存储常用特征），L3 为 Paimon 特征湖（全量数据）；

3）向量化查询：通过 Trino/StarRocks 执行向量化扫描，提升批量特征查询效率。对于需要同时获取多个用户特征的场景，向量化执行可将查询性能提升 3-5 倍；

4）文件局部性优化：通过合理设置'scan.split-open-file-cost'和'scan.split-target-size'参数，控制查询时打开的文件数量，减少不必要的 I/O 开销。

在线上环境中，95%的特征查询响应时间低于 50ms，有效支撑了实时规则引擎与模型推理，特别是在风险事件高发时段（如大促活动），系统仍能保持稳定的低延迟响应。

3.4 性能与成本收益

与原有 Hive+Flink+Kafka 架构相比，实时特征湖在以下方面取得显著提升：

典型案例分析：以“用户下单频率”特征为例，原有架构需要在 Kafka 中实时计算并通过 Hive 离线校准，存在 2-5 分钟延迟且偶尔数据不一致。迁移至 Paimon 特征湖后，通过流式聚合直接更新特征值，实现秒级更新且保证强一致性，有效识别了多个实时欺诈案例。

此外，系统还具备良好的可扩展性与运维便利性，支持动态扩容、在线 Schema 变更、故障自动恢复等能力，通过集成监控告警体系，实现了对数据质量、服务可用性和资源利用率的全方位监控。

表 1

指标	原有架构	实时特征湖	提升幅度
特征更新延迟	10-30 分钟	1-5 秒	99%以上
特征查询延迟	100-500ms	10-50ms	提升 3-5 倍
数据一致性	流批不一致	流批一体	完全一致
资源成本	基准	下降 30%	显著优化
开发效率	两套代码	一套 SQL	提升 50%
运维复杂度	高（多集群）	中（统一平台）	显著降低

4 系统落地与效果评估

4.1 在典型风控场景的落地实践

在电商风控场景中，面对账号盗用、交易欺诈、营销作弊等多类风险，我分三个阶段推进了实时特征湖的落地：

第一阶段（2024Q4）：构建基础特征湖平台，接入用户登录、浏览、下单等核心行为事件。此阶段重点验证 Paimon 在高并发写入下的稳定性，并建立基础的特征数据模型；

第二阶段（2025Q1）：扩展特征类型，引入设备指纹、IP 画像、历史订单聚合等复杂特征。开始将实时特征应用于核心风控规则，逐步替代原有的离线特征；

第三阶段（2025Q2）：与规则引擎和机器学习平台集成，支持实时特征服务与模型调用。建立完整的特征治理体系，包括特征血缘、质量监控与成本分析。

该系统在稳定运行 6 个月后，覆盖了全球多个业务区域，日均处理事件量超过 800 亿条，特征表总量超过 200 张，支撑了 20+风控场景的实时决策，特别是在支持大规模促销时发挥了重要作用，系统成功抵御了数万次恶意攻击，未出现任何数据不一致或服务不可用的情况。

4.2 弹性伸缩与容错设计

在高峰期（如大促）我启用双维度弹性：计算侧依据滞后水位与背压指标自动横向扩容；存储侧通过冷热分层与合并节流抑制尾部合并抖动。Flink 任务采用 checkpoint + exactly-once 与 idempotent upsert 组合，既保证端到端一致性，又允许在异常期间进行快速重启。

对于区域性故障，我使用多活读与跨区快照回放进行容灾演练：当主区出现 I/O 瓶颈或网络劣化，流量在 1–3 分钟内切至备区，备区从最近稳定快照恢复服务能力，回切后进行差量对账与补偿。

4.3 安全与合规实践

风控数据涉及账号、支付、设备标识等敏感要素，我采用分级授权（读/写/变更/运维四级）与细粒度审计（表级、列级与操作级留痕），关键表默认开启加密存储与传输。对跨区域访问，所有查询均需携带数据最小化与用途绑定的审计标签，离线导出必须通过脱敏策略与审批流程。

针对模型与规则变更，建立“快照闸口”机制：新版本上线前必须通过影子流量与快照对账达标；出现异常一键回退至上一个稳定快照，确保审计可追溯。

4.4 组织与流程落地

为降低迁移风险，我建立了“三本台账”：作业台账（迁移批次、负责人、依赖）、表台账（存储参数、快照策略、热度分布）与问题台账（根因、处置、复盘）。

发布流程实行“蓝绿+质量闸口+回滚预案”，把风险前移至发布前的灰度阶段；跨团队协作采用看板化推进，明确里程碑与 SLO（例如：峰值 p95 ≤ 2s、快照一致率 ≥99.9%）。成本侧按“容量-吞吐-延迟”三轴做月度复盘，指导分桶、分区与合并参数的下一轮调优。

5 总结与展望

本文基于实际业务场景，提出并实践了以 Apache

Paimon 为核心的实时特征湖架构，在大规模风控系统中实现了高并发写入与低时延召回的平衡，该系统已在典型的电商风控环境中成功落地，显著提升了风控决策的时效性与准确性。实践表明，Apache Paimon 作为新一代湖仓格式，在实时数据场景下具备显著优势，能够有效解决传统架构面临的流批不一致、维护复杂和成本高昂等问题。

未来，我将进一步探索以下方向：

智能特征管理：基于特征重要性自动优化存储与索引策略，实现冷热数据分层存储与智能降级；

联邦特征查询：支持跨区域、跨数据源的特征联合查询，构建全球统一的风控特征视图；

MLOps 集成：将特征湖与模型训练、部署、监控平台深度集成，构建端到端的实时风控 AI 体系；

实时数据治理：增强特征血缘分析、数据质量监控与隐私合规保障，构建可信实时数据生态；

异构计算支持：探索 Paimon 与 GPU/NPU 等异构计算设备的结合，进一步提升复杂特征计算的性能。

实时特征湖不仅适用于风控场景，还可扩展至推荐系统、用户画像、物联网数据分析等领域，具备广泛的行业应用前景。随着 Apache Paimon 社区的持续发展，我有信心进一步完善实时特征湖的技术体系，为行业提供更加成熟、稳定的大数据架构解决方案。

参考文献

[1] Apache Paimon Official Documentation. https://paimon.apache.org/

[2] Liu, C. Research on Industry Situation of Apache Paimon. 2025.

[3] Flink CDC Documentation. https://github.com/ververica/flink-cdc-connectors

[4] 王培斌. 基于 Paimon 与 StarRocks 的实时湖仓探索. 2024.

[5] 刘艳梅等. 湖仓一体技术及其行业现状研究. TrustCom 2023.

[6] 赵晓明, 李建国. 基于流批一体架构的实时风控系统设计. 计算机工程与应用, 2024, 60(5): 112-120.

[7] 陈晓红, 张伟. 大数据环境下实时特征平台架构研究. 软件学报, 2023, 34(8): 3567-3585.