

Federated Stream Processing Support for Real-Time Business Intelligence Applications

Irina Botan, Younggoo Cho, Roozbeh Derakhshan, Nihal Dindar,
Laura Haas, Kihong Kim, **Nesime Tatbul**



Introduction

- Business Intelligence (BI) enables better decision-making for businesses.
- In **operational BI**, real-time response to business events is critical, which requires:
 - reducing latency
 - providing rich contextual information
- We propose **MaxStream federated stream processing system** as a platform to meet these needs.

Talk Outline

- Example Use Cases & Motivation
- MaxStream System
 - Architecture
 - Usage
 - Feasibility
- Conclusions & Open Challenges

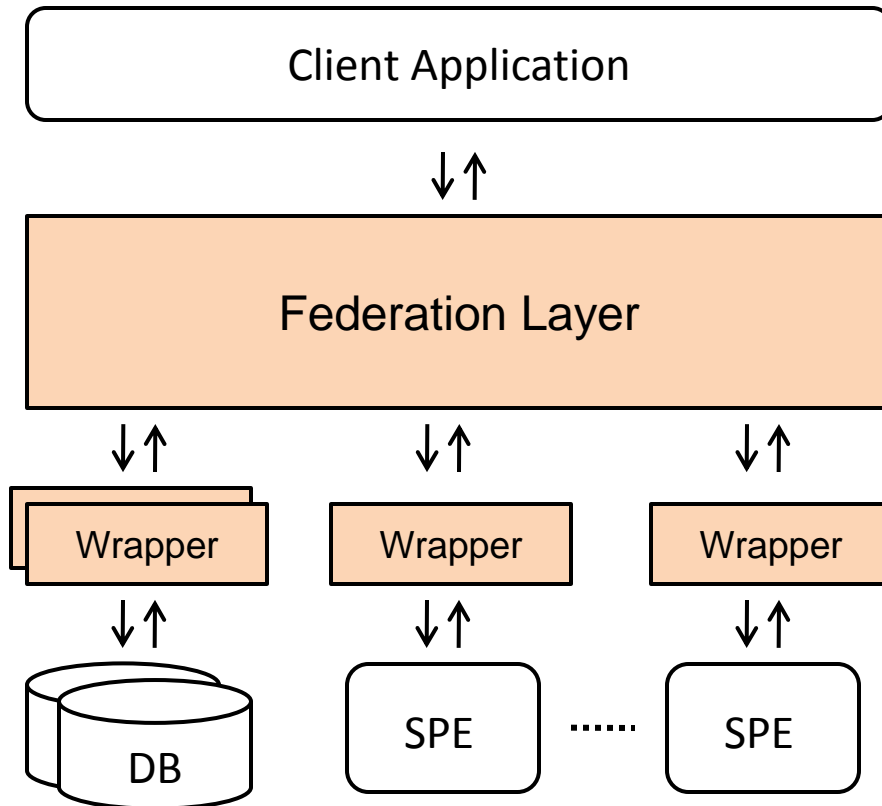
Example Use Cases

- Supply-Chain Optimization
 - Call Center Management
 - Quality Management in Manufacturing
 - SLA Monitoring and Maintenance
 - Global Shipment & Delivery Monitoring
 - Fraud Detection in Financial Companies
 - Real-time Marketing
 - ...
- Different levels of latency and data persistence requirements

e.g., Call Center Management

- Multiple centers across the globe
- Every incoming call is captured with arrival time, service start and end times
- Main BI tasks:
 - Run statistics on wait time, service duration, etc. for different regions
 - Generate reports, analyzing problems and proposing strategic improvements

MaxStream Architecture: From 30,000 ft



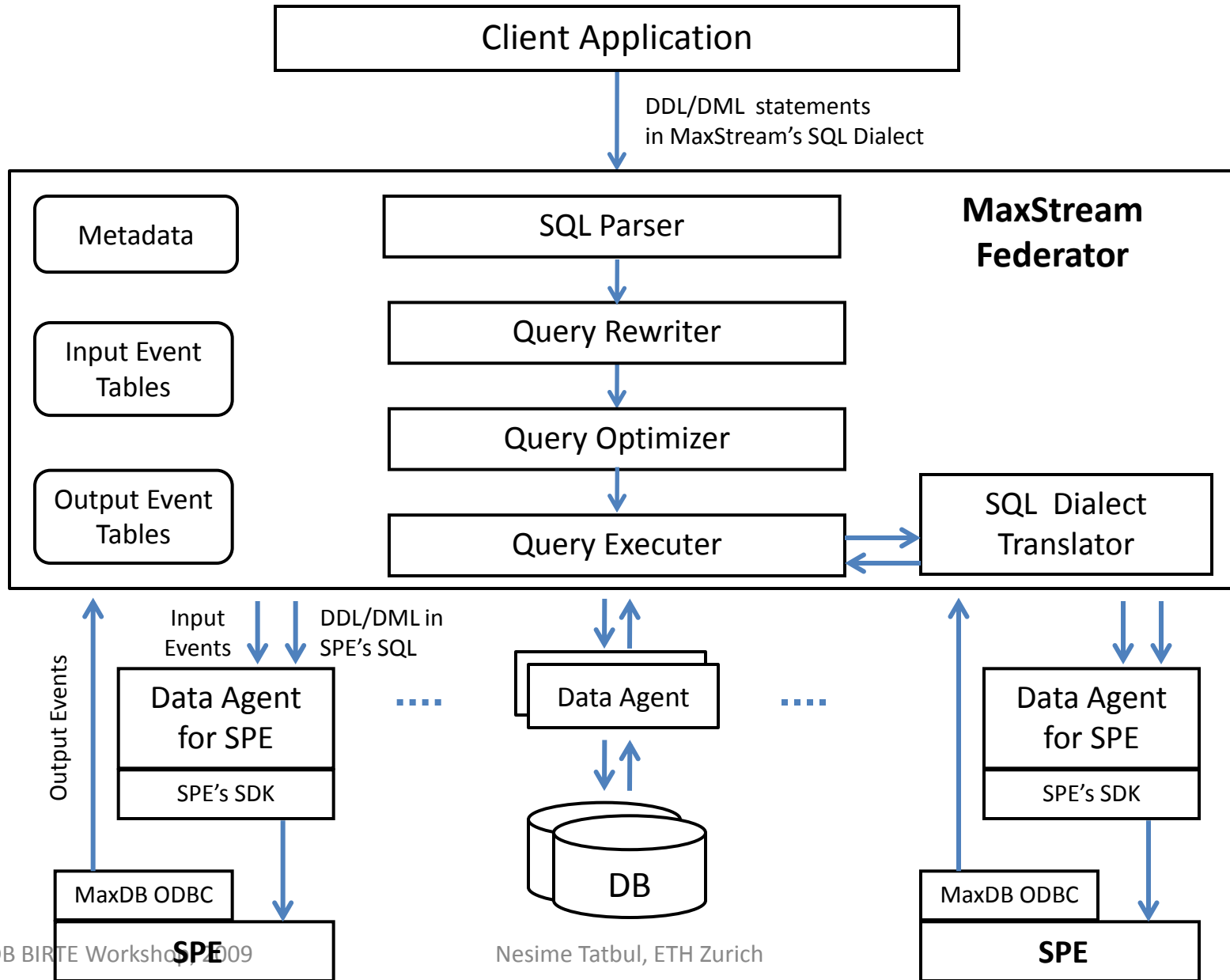
- Key ideas:

- Uniform query language and API
- Relational database infrastructure as the basis for the federation layer (in our case: SAP MaxDB and SAP MaxDB Federator)
- “Just enough” streaming capability inside the federation layer

Putting MaxStream into Context

- vs. Federated Databases
 - Less focus on data locality, more focus on functional heterogeneity
- vs. Stream Processing Engines (SPEs)
 - Unlike distributed SPEs, there may be heterogeneity
 - Unlike stream-relational SPEs, MaxStream federator is not a full-fledged SPE
- vs. Business Intelligence Software
 - Tighter integration between (possibly heterogeneous) SPEs and databases

MaxStream Architecture: A Closer Look



MaxStream Architecture

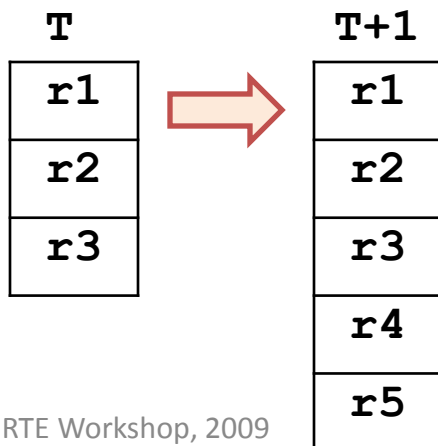
Two Key Building Blocks

- Streaming Inputs through MaxStream
 - **ISTREAM** Operator for Persistent input events
 - Tuple Queues for Transient input events
- Streaming Outputs through MaxStream
 - **Monitoring Select** over Event Tables
 - Persistent Event Tables for Persistent output events
 - In-Memory Event Tables for Transient output events

Streaming Persistent Input Events

- The ISTREAM (“Insert STREAM”) Operator
 - Relation-to-Stream operator first proposed by Widom et al. [STREAM Project], that streams new tuples being inserted into a given relation.
 - Example:

```
INSERT INTO STREAM CallStream
  SELECT OpCode, ArrivalTime, StartTime, EndTime
  FROM ISTREAM(CallTable);
```



ISTREAM(CallTable) at T+1 returns:

$\langle r4, T+1 \rangle, \langle r5, T+1 \rangle$

Streaming Output Events

- Opposite of streaming input events, but...
 - Unlike the SPE interface, the client application interface is not push-based.
- Alternative solutions:
 - Each client monitors its own alerts on a given table.
 - cumbersome and error-prone
 - A monitoring program does so for all registered clients using periodic select queries (i.e., polling) or triggers.
 - Not event-driven, inefficient, not scalable

Streaming Output Events

- Our Solution: Monitoring Select
 - Select operation blocks until there is at least one row to return.
 - For continuous monitoring, the client program re-issues Monitoring Select in a loop.
 - Monitoring Select operates on “Event Tables”.
- Example: Detect calls with unusually long waiting times.

```
SELECT *  
  FROM /*+ EVENT */ CallAnalysis  
 WHERE AvgWait > 10;
```

Hybrid Queries in MaxStream

- Hybrid queries are continuous queries that join Streams with Tables
 - Similar to joining Fact tables with Dimension tables in data warehouses
- One can conveniently use hybrid queries in MaxStream in two ways:
 - To enrich the input stream before it is passed to the SPE
 - To enrich the output stream after it is received from the SPE

Hybrid Queries: Call Center Example

```
CREATE TABLE CallTable (Opcode, ArrivalTime, StartTime, EndTime);
```

```
INSERT INTO STREAM CallStream
SELECT o.RegionNm AS Region, c.StartTime-c.ArrivalTime AS WaitTime,
       c.EndTime-c.StartTime AS Duration
FROM   ISTREAM(CallTable) c, OperatorsbyRegion o
WHERE  c.Opcode = o.Operator;
```

Enriching
the input in
MaxStream:

```
INSERT INTO TABLE CallAnalysis
SELECT Region, COUNT(*) AS Cnt, AVG(WaitTime) AS AvgWait,
       AVG(Duration) AS CallLength
FROM   CallStream
GROUP BY Region
KEEP 1 HOUR;
```

Continuous
Query
in SPE:

```
SELECT a.Region, a.AvgWait, a.AvgDuration, r.NOps, r.Training
FROM   /* +Event */ CallAnalysis a, Regions r
WHERE  AvgWait > 10
       AND a.Region = r.RegName;
```

Enriching
the output in
MaxStream:

Initial Feasibility Study

- Goal: to show
 - if MaxStream is useful in supporting real-time BI applications
 - whether MaxStream's performance overhead is acceptable
- Setup: SAP Sales and Distribution Benchmark
 - Persistent events, Throughput critical
 - Original benchmark: No streaming
 - We add streaming and compare the following two setups:
 - SD vs. SD with MaxStream/ISTREAM + SPE "X"
 - SD vs. SD with MaxStream/Monitoring-Select

SAP Sales and Distribution (SD) Benchmark

- It is a business benchmark that models a sell-from-stock scenario that consists of 6 transactions, each with 1-4 dialog steps and around 10 seconds of think-time for each.
 - Example transactions: Create customer order document, Create order delivery document, Create invoice, etc.
- Measure: throughput in the number of processed dialog steps per minute (SAPs).

Use of MaxStream in SAP SD Benchmark

MaxStream/ISTREAM + SPE "X"

- Stream incoming orders.
- Forward sales orders to SPE "X" via MaxStream in order to continuously compute the daily sum of sales orders for each product and region.

```
INSERT INTO STREAM SalesOrderStream  
  SELECT  A.MANDT, A.VBELN, A.NETWR,  
          B.POSNR, B.MATNR, B.ZMENG  
FROM    ISTREAM(VBAK) A, VBAP B  
WHERE   A.MANDT = B.MANDT  
        AND A.VBELN = B.VBELN;
```

MaxStream/Monitoring-Select

- Monitor big sales.
- Continuously monitor big sales orders (i.e., with amount > 95) by storing purchase orders in an event table and running Monitoring Select over it.

```
SELECT  A.MANDT, A.VBELN, B.KWMENG  
FROM    /*+ EVENT */ VBAK A, VBAP B  
WHERE   A.NETWR > 95  
        AND A.MANDT = B.MANDT  
        AND A.VBELN = B.VBELN;
```

MaxStream SAP SD Benchmark Performance

	SD	SD with ISTREAM	SD with Monitoring-Select
# of SD Users	16,000	16,000	16,000
Throughput (SAPs)	95,910	95,910	95,846
Dialog Response Time (msec)	13	13	13
DB Server CPU Utilization (%)	49.8%	50.6%	50.1%

- SD with streaming features achieves similar performance as the standard one.

Conclusions

- Real-time BI requires new platforms which offer
 - low latencies of stream processing
 - support for analytics of data warehouses
 - flexible, dynamic access to data of data federation engines
- MaxStream stream federation engine provides
 - access to heterogeneous SPEs and DBs
 - flexible persistence and data federation capabilities
- MaxStream is low-overhead and useful in various operational BI scenarios.

Open Challenges

- Unified continuous query execution model and semantics
- Cost- and Capability-based query optimization and dispatching over multiple SPEs
- Transactional aspects of federated stream processing
- Distributed operation aspects (e.g., load balancing, high availability)

Thanks!

- You 😊
- MaxStream team
- Chan Young Kwon (SAP Labs, Korea)
- ETH Zurich Enterprise Computing Center (ECC)



- More information:

<http://www.systems.ethz.ch/research/projects/maxstream/>