Data Stream Management Systems and Query Languages

Advanced School on Data Exchange, Integration, and Streams (DEIS'10) Dagstuhl

Sandra Geisler

Information Systems - Informatik 5

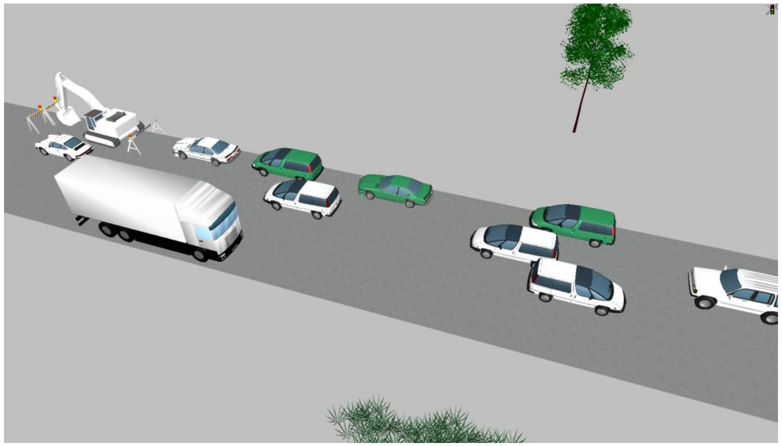
RWTH Aachen University

09.11.2010

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

New Applications – New Requirements



Traffic Applications

- Rapid emission of messages, e.g., hazard warnings
- Derive traffic information from processed data
- Integration of data from multiple mobile and static sources

Health monitoring

- Sensors produce data at high rates
- Integration with further information, e.g., EHR
- Real-time processing to analyze health information and predict events

Other applications:

- Stock analysis
- Production monitoring
- User behaviour (click analysis)
- Position monitoring (soldiers, devices,..)

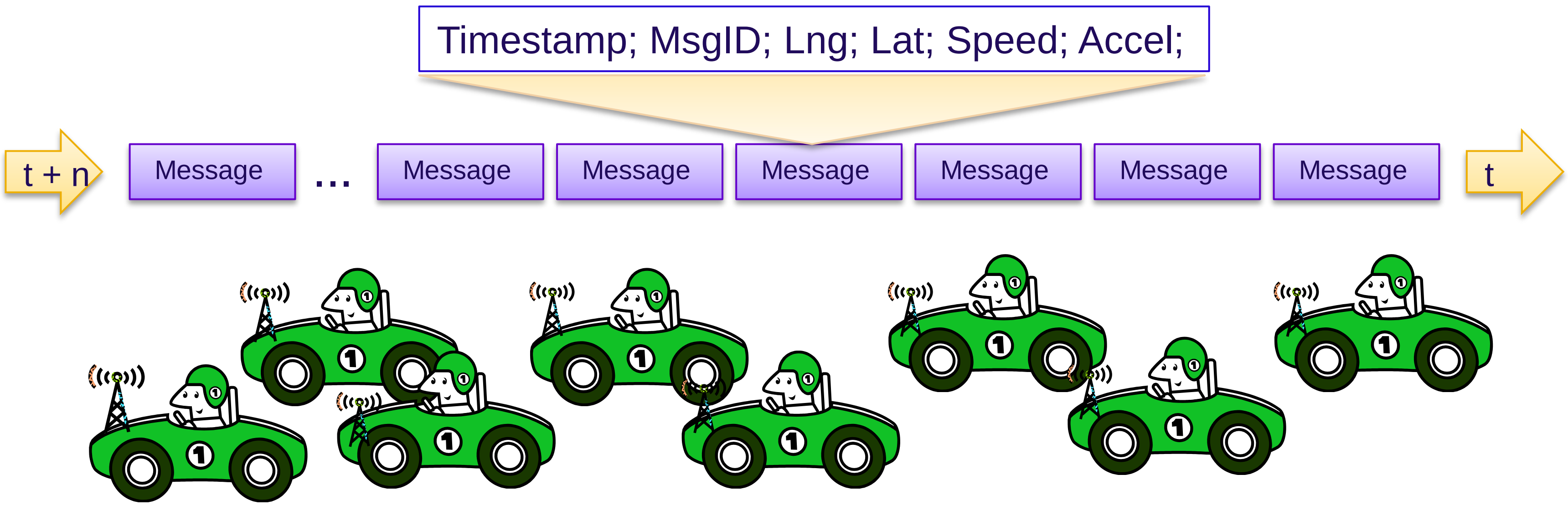
Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Running Example – Car2X Communication

Two kinds of messages:

1. Based on events vehicles produce a message describing the event
2. Vehicles send probe data periodically



Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Comparison of Applications

Traditional Applications	Streaming Applications
Irregular transactions, batch processing	Continuous flow of data
Possibly very large, but finite data set	Unbounded stream
Frequent analysis, multiple passes	Continuous analysis, one pass
More tolerant time requirements, predictable	Data is produced at high rates, real-time requirements, bursty
Time may be unimportant, neglected, all information may be important	Notion of time is important, recent information more important
Passive behaviour (pull)	Active behaviour (push), trigger-oriented, monitoring
Data assumed to be complete up to that point in time	Asynchronous and incomplete data arrival, inaccuracies
Permanent storage required	Not all information must/can be stored permanently → "volatile"

→ What does that mean for a data management system?

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Agenda

1. Introduction
2. **Data Stream Management Systems**
3. Query Languages
4. Query Plans & Operators
5. Quality Aspects in DSMS
6. Our work

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Requirements for a DSMS

- Allow continuous queries, but also ad-hoc queries, views
- Handle unbounded streams while dealing with limited resources
- Delivery of incremental results and processing of subsets
- Fulfilment of real-time requirements for processing and response
- Scalability in number of queries and data rates
- Support for fault tolerance: missing, out-of-order, delayed data
- Active system behaviour → push, trigger
- Predictable and repeatable results → fault tolerance and recovery [Stonebraker et al. 2005]
- High-availability [Stonebraker et al. 2005]
- Update of data after processing [Abadi et al. 2005]
- Dynamic query modification [Abadi et al. 2005]
- Shared processing of data by multiple queries, adaptivity to addition and removal of queries [Chandrasekaran et al. 2003]
- Provide support for signal processing [Girod et al. 2008], objects, lists

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

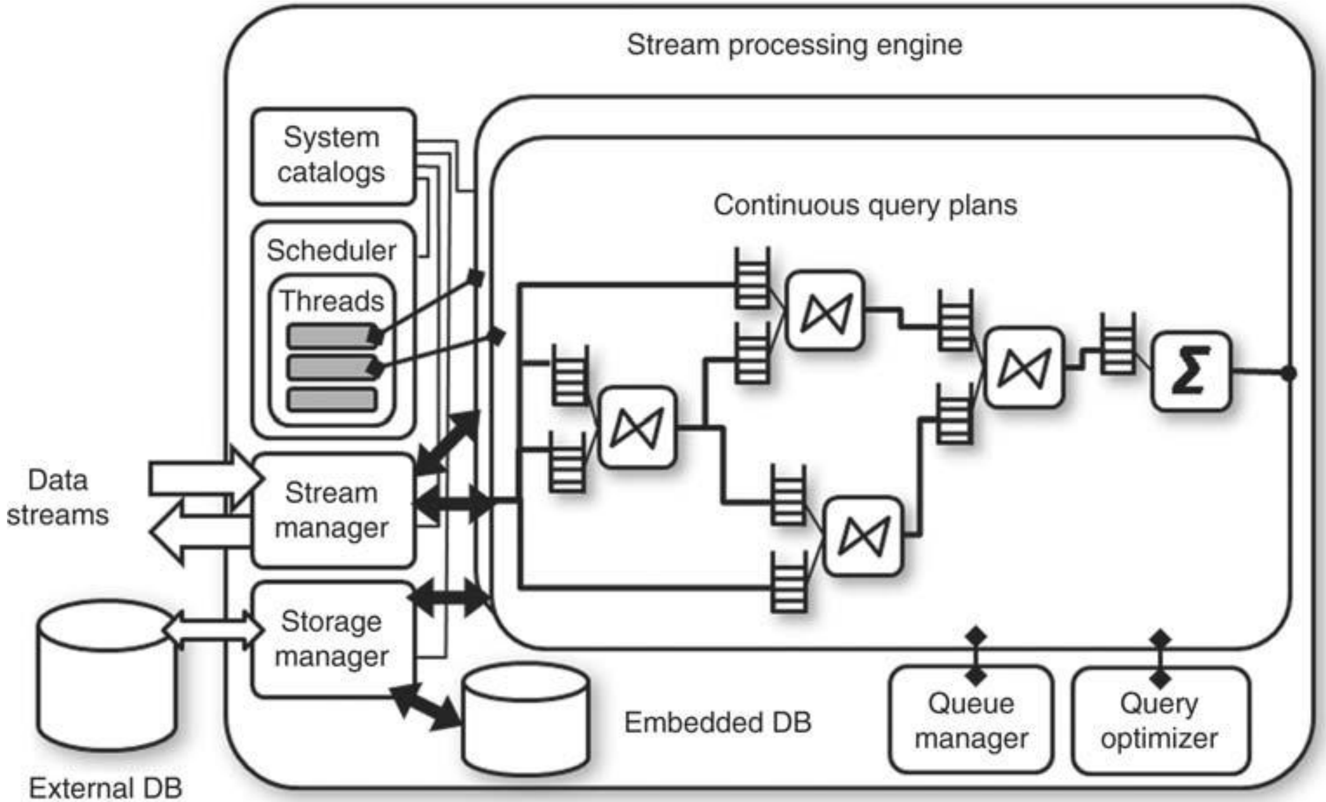
Flaws in Common DBMS Processing Streams

- Human-active DBMS-passive model vs. DBMS-active human-passive model [Abadi et al. 2003]
- Turns common DBMS idea bottom-up → data retrieval triggers queries in contrast to queries trigger data retrieval [Chandrasekaran et al. 2003]
- Relational algebra assumes finite sets → blocking operators do not suit for streams (wait for results, no time-out, no approximate query answering)
- Process-after-store mechanism: triggers can be used, but do not scale [Abadi et al. 2003] → high latency and overhead for handling streaming data
- Cannot deal with out-of-order data [Stonebraker et al. 2005]
- Predictable results → order of storage and processing of data has to be controlled externally [Stonebraker et al. 2005]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

General Structure of an SPE



[Ahmad and Çetintemel, 2009]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Overview of DSMSs – Research Projects

Project	Research Group	Runtime	Description
Tapestry	Xerox Parc (D. Terry, D. Goldberg et al.)	1992 ?	uses a commercial append-only database, cont. querying by SPs
TelegraphCQ http://telegraph.cs.berkeley.edu (Fjords, PSoup.)	UC Berkeley (Hellerstein, Franklin)	2000 - 2007	reuses components from DBMS PostgreSQL, dataflows composed of set of operators (e.g., Eddy, Join) connected by Fjords, Language: SQL, scripts
STREAM http://infolab.stanford.edu/stream/	Stanford University (A. Arasu, J. Widom, B. Babcock, S. Babu et al.)	2000-2006	Probably the most famous one, comprehensible abstract semantics description; Language: CQL
Aurora/Borealis http://www.cs.brown.edu/research/borealis	Brown Univ., Brandeis Univ., MIT (Abadi, Cherniack, Madden, Zdonik, Stonebraker et al.)	2003-2008	Distributed system, uses notions of arrows, boxes and connection points for operator networks ; Commercial: StreamBase; Language SQuAl
PIPES http://dbs.mathematik.uni-marburg.de/Home/Research/Projects/PIPES	Universität Marburg (Seeger, Krämer et al.)	2003-2007	Commercial: RTM Analyzer Language: PIPES, define logical and physical query algebra on multi-sets, use algebraic optimizations
System S/ SPC/ SPADE/ http://domino.research.ibm.com/comm/research_projects.nsf/pages/esps.index.html	IBM T.J. Watson Research	2006-2008	Distributed System, notion of operator network, Commercial: InfoSphere; Language: SPADE
StreamMill http://magna.cs.ucla.edu/stream-mill	UCLA (H. Takkhar, C. Zaniolo)	Ongoing	Inductive DSMS → mining implementable with SQL and UDAs, support for XML data; language: ESL
Global Sensor Networks http://sourceforge.net/apps/trac/gsn/	EPF Lausanne, Digital Enterprise Research Insitute (DERI) (Salehi, Aberer et al.)	Ongoing	Wraps existing rel. DBMS with stream functionality; language: common SQL

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Overview DSMS – Commercial Products

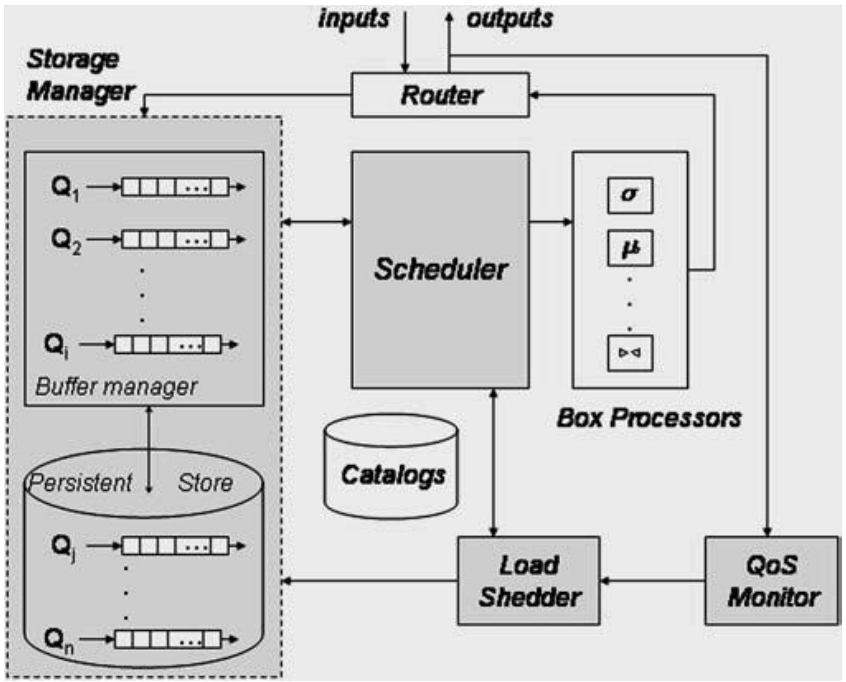
System	Company	Based on	Description
InfoSphere Streams http://www-01.ibm.com/software/data/infosphere/streams/	IBM	System S/ /SPADE/ SPC	Stand-alone product, only supports Linux?, queries over structured and unstructured data sources Language: SPADE
Oracle Streams http://www.oracle.com/technetwork/database/features/data-integration/default-159085.html	Oracle	--	Integrated in Oracle 11g; Language: CQL
StreamInsight http://www.microsoft.com/sqlserver/2008/en/us/r2-complex-event.aspx	Microsoft	---	Integrated in MS SQL Server 2008 Release 2; Language: .NET, LINQ
StreamBase http://www.streambase.com	StreamBase	Aurora/Borealis	Stand-alone products (Server, Studio, Adapters..); Language: StreamSQL
TruSQL Engine http://www.truviso.com	Truviso	TelegraphCQ?	Language: StreaQL
Esper (Open Source) http://esper.codehaus.org/	EsperTech	---	Available in .NET and Java, Stand-alone product; Language: EPL

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Example – The Aurora System

- **Router:** forwards elements to storage manager or outputs
- **Storage Manager:**
  - Maintains operator queues & manages buffer
  - For each queue, disk storage blocks are used (circular buffer)
  - Keeps blocks of high priority queues in main memory
- **Scheduler:**
  - picks the next operator to be executed
  - Shares table with SM with priority, perc. of operator queues in main memory, flag if box is running
  - Priority is based on QoS statistics
  - Train scheduling and superbox scheduling: minimize box calls and I/O operations by building “tuple trains”
- **Box processors:** execute the operators (multi-threading)
- **QoS Monitor:** monitors system performance and activates load shedder
- **Load Shedder:** based on introspection tuples are dropped using QoS information
- **Catalog:** meta information about network, inputs, outputs, statistics etc.



[Abadi et al. 2003]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

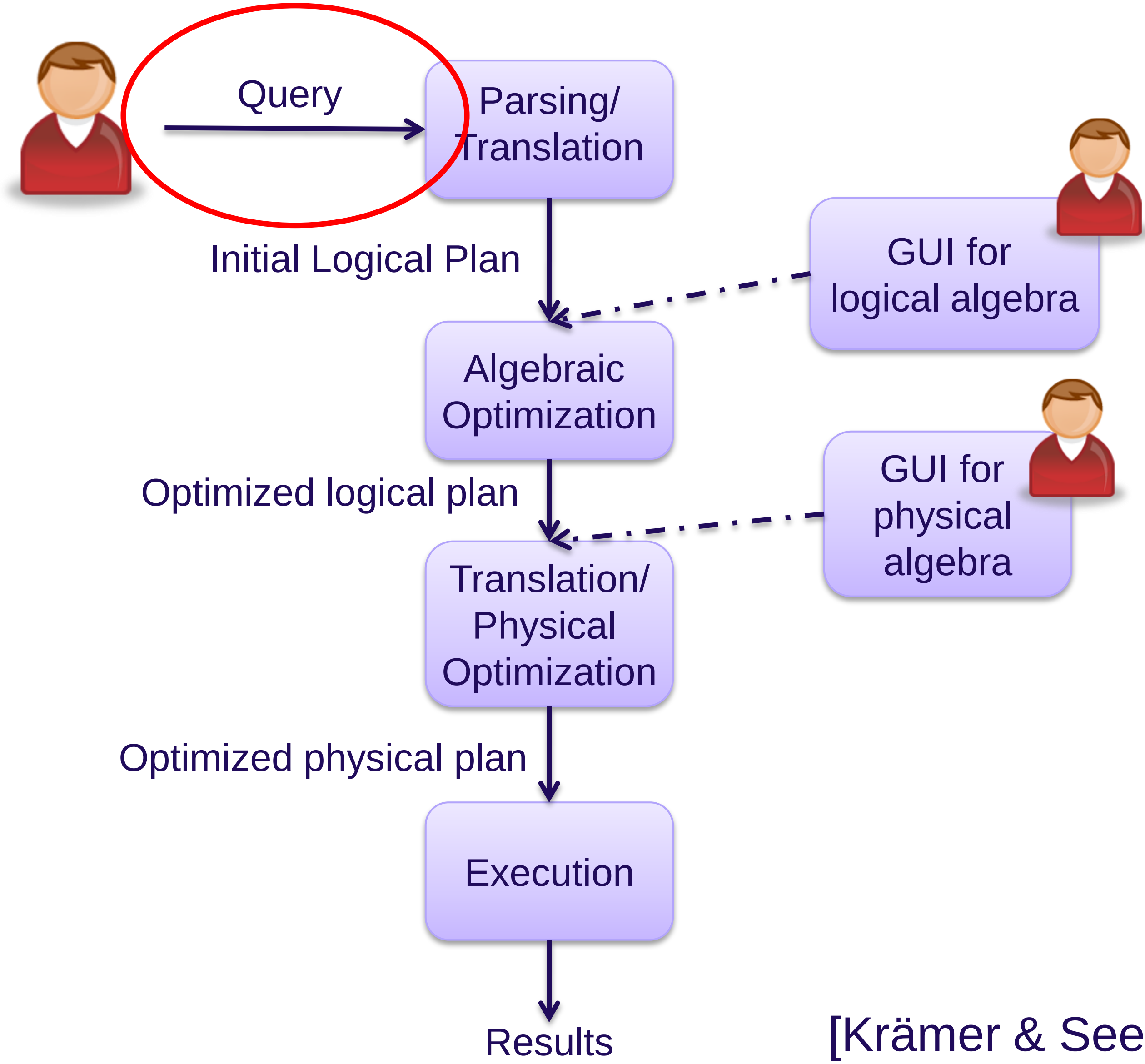
Agenda

1. Introduction
2. Data Stream Management Systems
3. **Query Languages**
4. Query Plans & Operators
5. Quality Aspects in DSMS
6. Our work

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Query Processing Overview



[Krämer & Seeger 2009]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Requirements for Query Languages in DSMS

- Windowing: which kinds of windows are supported?
- Correlation: combine streams and static relations in a query
- Provide all standard SQL operations → approved set of query operators
- User-defined operations/functions
- Language closure: operators get streams as input and output streams → no conversion into finite relations in between
- Pattern matching: identify subsequences of tuples
- Expressiveness: must be expressive enough for targeted apps → which operations can be formulated?
- Well-understood formal semantics, e.g., enables optimization

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Query Formulation

- Extensions of the SQL Standard, e.g.,
  - CQL: STREAM [Arasu et al. 2006], Oracle Streams
  - PIPES [Krämer et al. 2009]
  - ESL [Thakkar et al. 2008]: StreamMill

```
SELECT Istream Count(*) FROM
  C2XMgs[Range 1 Minute Slide 10s]
WHERE
  Speed > 30.0
```

- Assembling of operators, e.g.,
  - Aurora/Borealis (SQuAl)

↓ Filter (Speed > 30.0) ↓ Aggregate(CNT, Assuming O, Size 1 minute, Advance 10 second) ↓

  - System S/ InfoSphere (SPADE)

C2X_Source → Functor → Aggregate → TCP_Sink

- XPath-based languages, e.g., [Peng and Chawathe 2003]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Timestamps

- Monotonic time domain T: ordered, infinite set of discrete time instants $\tau \in T$ [Patroumpas and Sellis 2006] → multi-set semantics
- Explicit or external timestamp (application time):
  - Tuples enter system with a predefined timestamp field from the source
  - Disadvantage: elements may not arrive in order
- Implicit or internal timestamp (system time):
  - Timestamp is defined by the system, add. timestamp field
  - Preserve timestamps → enables to measure output delay (Aurora)
- Logical clock:
  - Consecutive integer with distinct values
  - On receipt (global order) or by each operator's input queue
- Latent timestamps (StreamMill):
  - Only created when required, other operators use order of input queue
- Operator timestamps, e.g., for a join → which timestamp should be used?

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Timestamps – Example ESL

```
CREATE STREAM C2XMgs(
        ts timestamp, msgID char(10), lng real,
        lat real, speed real, accel real)
ORDER BY ts;
SOURCE 'port5678';
```

Explicit Timestamp

```
CREATE STREAM C2XMgs(
        ts timestamp, msgID char(10), lng real,
        lat real, speed real, accel real,
       current_time timestamp)
ORDER BY current_time;
SOURCE 'port5678';
```

Implicit Timestamp

```
CREATE STREAM C2XMgs(
        ts timestamp, msgID char(10), lng real,
        lat real, speed real, accel real)
SOURCE 'port5678';
```

Latent Timestamp

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Semantics – Data Model (Stream Elements)

- In general:
  (s,τ) , tuple s with schema $(A_1,...,A_n)$, timestamp τ

- Simple data types (e.g., STREAM, Aurora, GSN,..)
  - Borealis:
    - With key $(k_1,...k_n,A_1,..,A_m)$, used to identify tuples for revision
    - Adds revision flag: **+, -,** ←, also QoS information can be included
  - CESAR (event processing algebra [Demers et al. 2005]):
    - Event-based: $(A_1,...,A_m, \tau_0,\tau_1)$ → denotes start and end of an event
- Objects (PIPES, System S, Xstream [Girod et al. 2008]):
  - Finite sequence of objects and a timestamp
  - Composite type of a tuple → in relational case the schema [Krämer and Seeger 2009]
  - Can use functions and predicates for arbitrary types

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Ordering in Streams

- Temporal ordering as a many-to-one mapping $f_O : D_S \rightarrow T$ with properties [Patroumpas and Sellis 2005]
  - Existence: $\forall\, s \in S, \exists\, \tau \in T$, such that $f_O(s) = \tau$
  - Monotonicity: $\forall s_1, s_2 \in S$, if $s_1.A_\tau \leq s_2.A_\tau \rightarrow f_O(s_1) \leq f_O(s_2)$
- In general:
  operators assume non-decreasing order of arriving elements, e.g., in STREAM: time advances from τ-1 to τ when all inputs of τ-1 have been processed
- But this is not valid, especially for explicit timestamps
  - Data from sources may be in the right order due to communication problems, delays, asynchronism
  - Ordering, arrival in time not guaranteed
    $\rightarrow$ poses problems when windows are used (are the right tuples included?)

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Handling of Unordered Streams

- Relax assumption about ordering (Aurora):
  - Parameter specification to relax assumptions about local ordering (slack parameter k)

  Ordering Constraints [Arasu et al. 2004]:
  - Ordered arrival constraint (windows) → at least k+1 tuples with A value ≥ s.A after s
  - Clustered arrival constraint (aggregates) → at most k+1 further tuples after s without value v
  - Referential integrity constraint (joins) → delay between a tuple in S_1 and tuple in S_2 at most k

- Dictate ordering
  - Heartbeats & Input Manager (STREAM):
    - sends message with timestamp τ_i which indicates, that τ_i has ended
      → no further elements with timestamp τ_i will arrive
    - Implicit timestamps → elements are ordered anyways, DSMS sends heartbeat
    - Explicit timestamps → sources have to generate the heartbeat or DSMS has to deduce these from "environment parameters", such as time delay between sources
  - Dropping tuples (e.g., GSN)
  - Partition into additional out-of-order stream (StreamMill)
    → handling is left to the user

- Correct stream order locally
  - Ordering operators, such as BSort (Aurora)

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Semantics – Data Model (Streams and Relations)

- Stream
  - In general:
    - Append-only (possibly infinite) sequence of tuples with uniform schema evolving in time
  - STREAM:
    - Base stream and derived stream, unbounded multi-set of elements → duplicates
  - PIPES (logical and physical algebra & query plans):
    - **Raw streams** (s, τ): sequence of elements from input sources
    - **Logical streams** (s, τ, n): order-agnostic representation of multi-set of elements → show validity of tuples at time-instant level

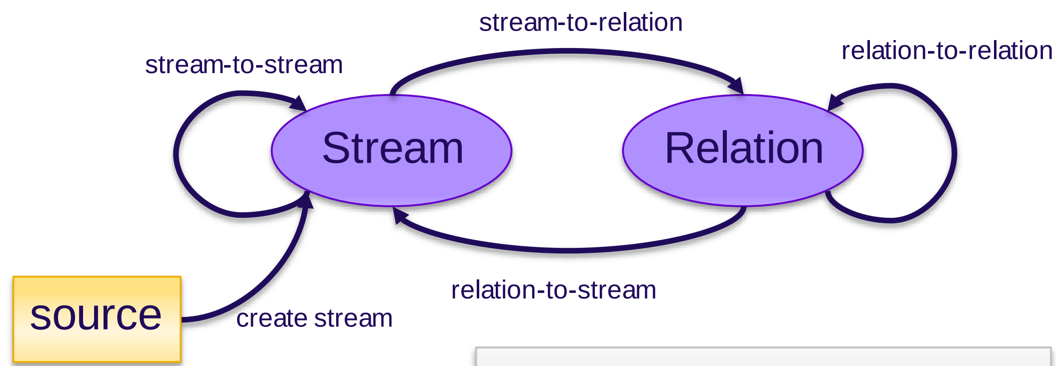
      $\times(S_1, S_2) := \{(s_1 \circ s_2, \tau, n_1 \cdot n_2) \mid (s_1, \tau, n_1) \in S_1 \wedge (s_2, \tau, n_2) \in S_2\}$
    - **Physical streams** (s,v): v validity interval, processed in physical operators
  - Denotational View [Maier et al. 2005]:
    - Gives several different representation possibilities described by reconstitution functions, e.g., set(t), bag(t)
- Relation
  - STREAM
    - Mapping from each time instant in T to a finite, but unbounded multi-set of tuples with schema R → notion of time
    - Set of tuples that may vary over time → instantaneous relation

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Semantics - Operators

- Create Stream
- Stream-to-Relation
- Relation-to-Relation
- Relation-to-Stream
- Stream-to-Stream



Adapted from [Arasu et al.2004]

- Language closure:
  - S2S operators (Borealis, System S, ..) → closed under streams
    - Allows nesting of queries
    - Allows for better algebraic optimization
  - No real Stream-to-Stream (Istream, Rstream, Dstream)

- Correlation, e.g., in STREAM, StreamMill, Aurora:
  - Variants of joins: with or without windows

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Stream-to-Relation - Windows

- Tackle problem of unbounded streams → retrieve finite portion of the stream (temporary relation)
- General definition [Patroumpas and Sellis 2005]:
  A window is the set of all elements of a stream for which a conjunctive window condition E holds at a certain time instant (also window state for this time instant)
- Windowing attribute: determines the ordering → mostly timestamps
- Definition of Windows:
  - Implicit definition: integrated in other operators (e.g., Aurora)

    ```
    Aggregate(CNT, Assuming O, Size 1 minute, Advance 10 second)
    ```

  - Explicit definition: operator on its own, e.g., in STREAM
    → may violate language closure

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Windows – Measurement Unit and Edge Shift

- Measurement Unit:
  - One bound must be specified to define size
  - Logical units:
    - Time-based windows
    - Value-based windows: need increasing sequence of values for discriminating attribute → have to know when no more values lie in this interval
  - Physical units:
    - Count-based or tuple-based windows
    - Partitioned windows: separates the stream into substreams depending on grouping attributes → window is the union of the windowed substreams
- Edge shift
  - Fixed-bound(s) windows: at least one bound is fixed, e.g., fixing lower bound and shifting upper bound → **landmark windows**
  - Fixed-band windows: fixed upper **and** lower bounds → keep state
  - Variable-bounds windows: both bounds are flexible, size is fixed → **sliding windows**

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Windows – Progression Step

- Progression step:
  - Window progresses up on arrival of new tuples or time advancement
  - Unit step vs. Hops: number of tuple or time instants at a time

  - Tumbling:
    - windows is filled until its boundaries are reached, no overlapping
    - If it is full → operator evaluates the content
  - Sliding:
    - Window moves forward on tuples or time advancement, overlapping possible
    - Non-monotonic → while window moves, new results are produced and old ones expire (no accumulative results)
    - Option: Use of negative tuples to cancel expired results
  - Punctuation-based:
    - Punctuations are flags set into the stream
    - Operators accumulate elements until a punctuation is reached and then evaluate the window

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Windows - Examples

```
SELECT Istream Count(*) FROM
  C2XMgs[Range 1 Minute Slide 10s]
WHERE
  Speed > 30.0
```

Sliding Window (CQL)

time

```
SELECT Istream Count(*) FROM
  C2XMgs[Range 1000 Slide 1000]
WHERE
  Speed > 30.0
```

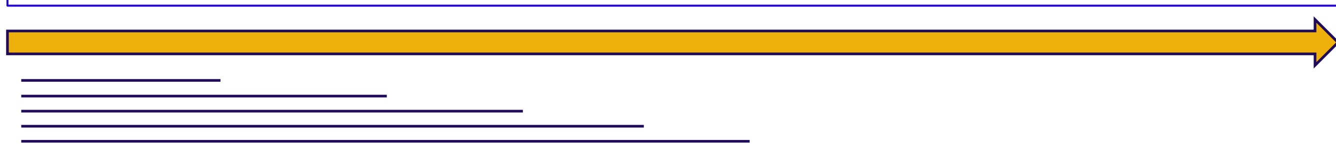
Tumbling Window (CQL)

time

```
SELECT Count(*) FROM
  C2XMgs <LANDMARK RESET AFTER 600 ROWS ADVANCE 20 ROWS>
WHERE
  Speed > 30.0
```

Landmark Window (TruSQL)

time

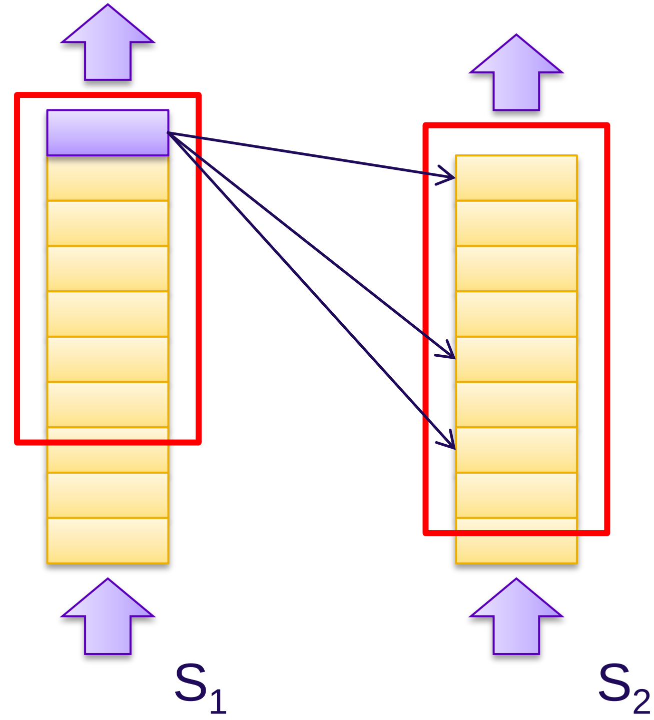


Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Stream-to-Relation – Windowed Operators

- Projection, Selection: not necessary, but often required for applications
- Deduplication: only returns the most recent tuple of its kind
- Windowed Join, Sliding Window Join:
  - Between two windows, but extendable to multi-way join
  - When a new tuple arrives in one of the windows it is matched against tuples of the other window
  - Commutative & associative, distributive over selection and projection
  - Eager and lazy variants [Golab and Öszu 2003]
- Aggregates:
  - Grouping of tuples in window according to attributes in group list
  - Application of aggregate function
- Set operations
  - Windowed union & intersection: not distributive over selection



Adapted from [Patroumpas and Sellis 2005]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Relation-to-Stream

- Creates an unbounded stream S from finite relation R
- Concatenate tuples by creation timestamps as operator output
  → too many duplicates (accumulative results)
- Better: just consider the differences between two time steps
- Explicit use of specific operators (STREAM):
  - **Istream** (insert stream): whenever a new tuple is added to R between τ-1 and τ, it is also added to S
    → only new tuples with timestamp τ are output
  - **Rstream** (relation stream): outputs all tuples of relation R at time τ
  - **Dstream** (delete stream): Outputs all tuples which have been deleted from R between τ-1 and τ → only deleted tuples with timestamp τ are output

```
SELECT Dstream(MsgID) FROM C2XMgs[Range 20 Seconds]
```

- Implicitly integrated into other operators
  - Istream mostly used (e.g., TelegraphCQ, Aurora, StreamBase)

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Stream-to-Stream – Example Aurora (Stateless)

- **Filter** $(P_1,...P_m)(S)$: defines one or more filter predicates on an input stream. If a tuple matches → route it to the corresponding P_i output, m+1 outputs (one for else), similar to rel. SELECT
- **Map**$(B_1=F_1,...,B_m=F_m)(S)$: constructs new stream elements for an output by defining functions over the input tuples (similar to projection)
- **Union**$(S_1,...,S_n)$: streams with common schema are merged into one stream

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Stream-to-Stream – Example Aurora (Stateful)

- Have to assume some ordering to stay in finite bounds → **Order**: O (On A, Slack n, GroupBy $B_1,..,B_m$)
- **BSort**(Assuming O)(S): Bubble sort on the stream over the data on attribute A
- **Join**(P, Size s, Left Assuming O_1, Right Assuming O_2)(S_1,S_2): P being a join predicate, s = Size of the window, O_1 and O_2 are orderings on S_1, S_2 respectively.
- **Resample**(F, Size s, Left Assuming O_1, Right Assuming O_2) (S_1,S_2): similar to semijoin, asymmetric, F= window/aggregate interpolation function over S_2
- **Aggregate** (F, Assuming O, Size s, Advance i,[Timeout z])(S): F = window/aggregate function (e.g., AVG), s = Size of the window, i=sliding step, timeout to prevent blocking when waiting for elements

Aggregate(CNT, Assuming O, Size 1 minute, Advance 10 second)

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

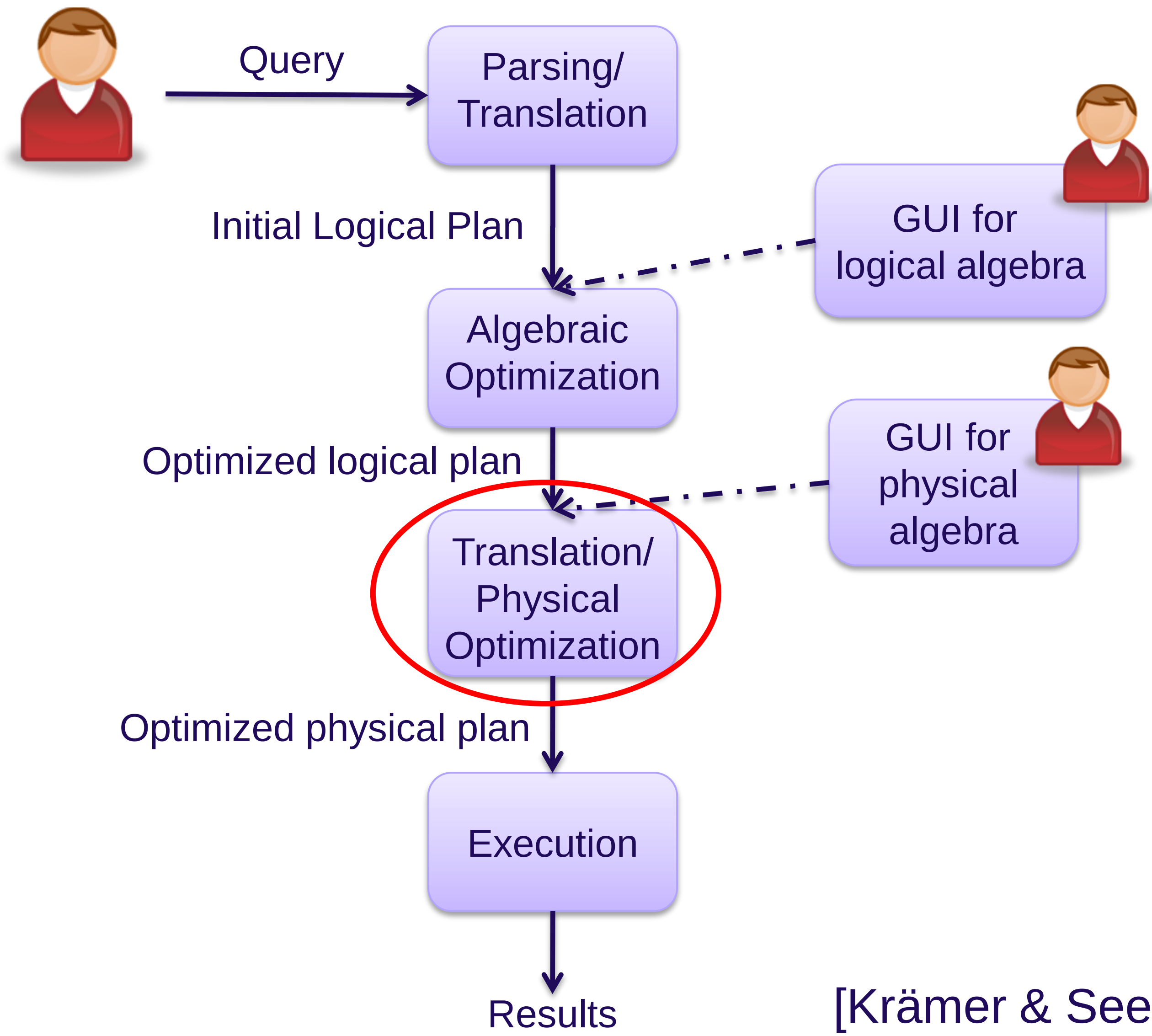
Agenda

1. Introduction
2. Data Stream Management Systems
3. Query Languages
4. **Query Plans & Operators**
5. Quality Aspects in DSMS
6. Our work

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Query Processing Overview

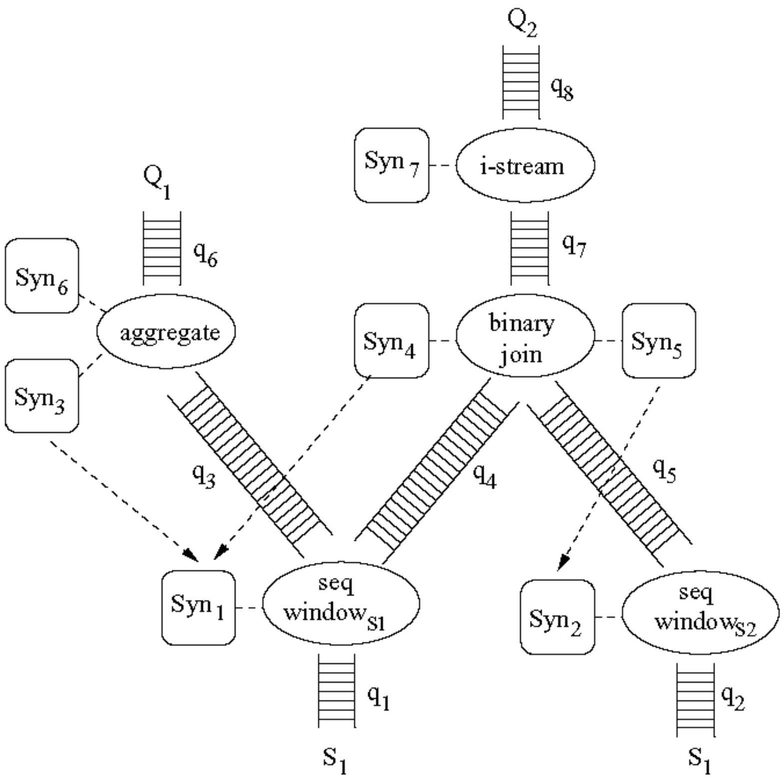


[Krämer & Seeger 2009]

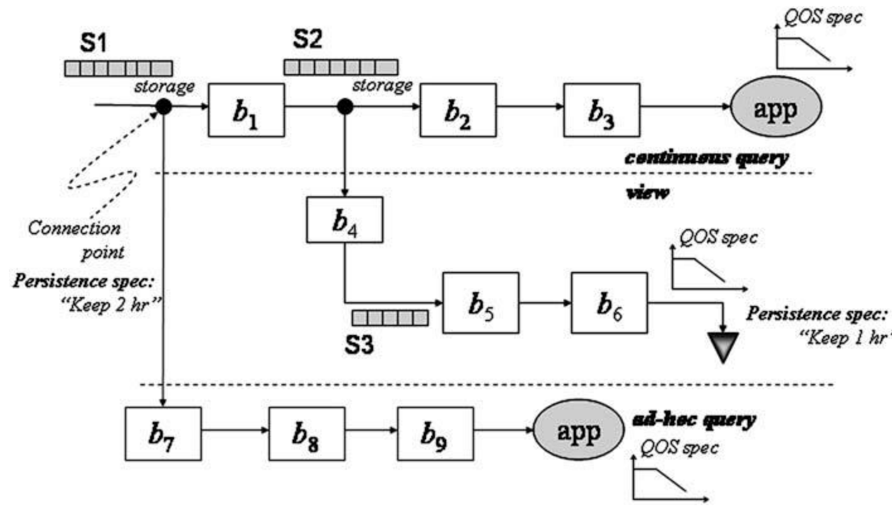
Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Physical Query Plan - Examples



[Arasu et al., 2006]



[Abadi et al. 2003]

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Supporting structures

- Queues
  - Connect outputs of producing operators with inputs of consuming operators
  - Buffer the elements for the consuming op.
  - Ordering: elements can be placed in a specific order in the queue
- Synopses
  - Stores a state for an operator in a specific data structure
  - Examples:
    - SHJ store hash tables for each stream
    - Window state
    - Summary for approximate query answering → different techniques, e.g., using wavelets, histograms, sketching...
  - Synopsis sharing with stores and stubs (STREAM)
    - Some operators may need similar or identical results
    - Store: keeps the union of intermediate results
    - Same interface as synopses → reports status to store

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Blocking Operators

- Unable to produce a tuple without knowing the entire input

- Operators: sorting, count, min, max, avg, join

- Resolutions:
  - Punctuations:
    - Mark in the stream when an operator should evaluate
    - After a punctuation no tuples with matching data will come
    - Representation, e.g., in Niagara → data using the schema of the stream filled with a series of pattern, e.g. restrict timestamp field indicates no more tuples matching an interval of dates will come
    - Disadvantage: sources have to produce these punctuations
  - Non-blocking counter parts → Example: Joins

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

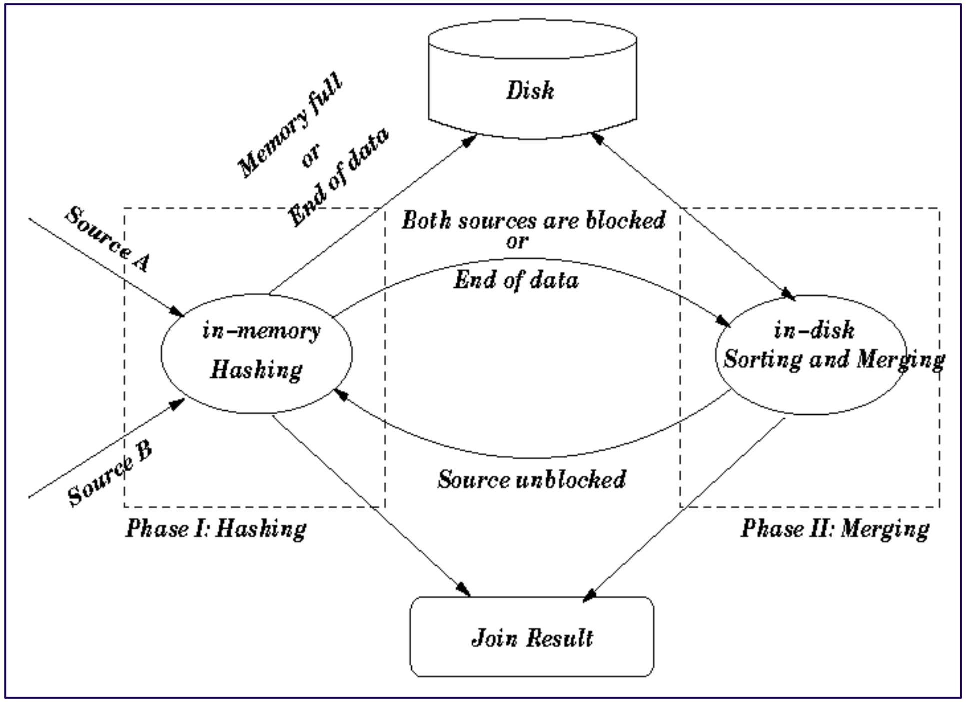
Join Implementations

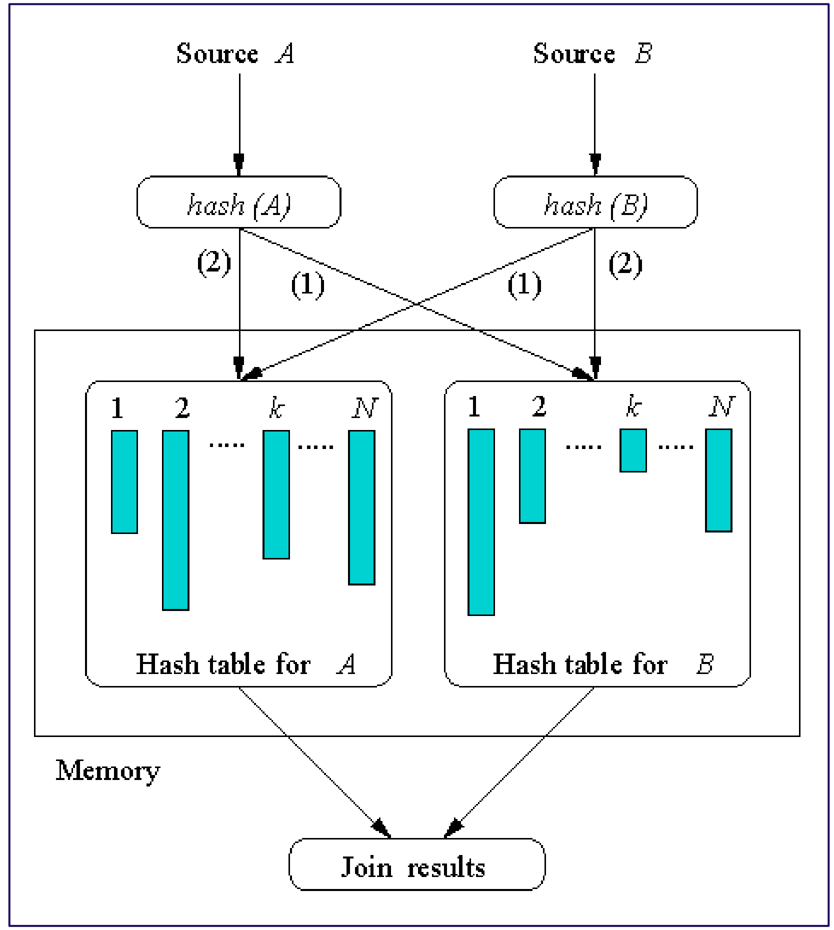
- Nested Loop Join (sliding window join) [Kang et al. 2003]
- Non-blocking Symmetric Hash Join:
  - Two hash tables A, B, both in memory, a hash function h
  - If a new tuple t_1 for stream A arrives, calculate $h(t_1)$ for A and probe it with values for $h(t_1)$ in hash table B, store tuple in the hash table at $h(t_1)$
  - Disadvantage: Only equi-join possible
  - Use trees or lists → can be used for Theta-Joins
- XJoin [Urhan and Franklin 2000]:
  - Similar to SHJ
  - if memory exceeded thresholds outsource biggest bucket
  - if one or both sources are stalled (no tuple arrives) → perform join with outsourced data
  - no interruption, all results are produced
- Ripple join [Haas and Hellerstein 1999]
  - Retrieve randomly one tuple from each stream at each sampling step → are joined with each other and previously seen tuples
  - Square: sampling rate of both is equal, rectangular: one stream is sampled more often than the other
- Adaptive solution [Kang et al. 2003]:
  - Depending on predicate, stream rate etc. the operator is dynamically chosen

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Example – Hash-Merge-Join [Mokbel et al. 2004]





Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Agenda

1. Introduction
2. Data Stream Management Systems
3. Query Languages
4. Query Plans & Operators
5. **Quality Aspects in DSMS**
6. Our work

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

System Perspective (QoS):

- Aurora [Abadi et al. 2003]
  - Calculates QoS values for response times, tuple drops and values produced
  - Users defines two-dimensional QoS graphs for each output and each quality dimension to describe QoS → tolerable QoS boundaries
  - Example: importance of (numerical) values can be described by a function
  - Uses QoS for adaptation of scheduling priorities
    - State-based: rates utility of a box output, scheduler picks the output with highest utility, whereby utility means, how much it will harm QoS if its execution is deferred
    - Feedback-based: if latency in QoS of an output is high, priority is increased, otherwise decreased

- Borealis [Abadi et al. 2005]
  - QoS is predictable at any point in the query, not only outputs
  - Extends messages with QoS information (Vector of Metrics) which contains content-related (e.g., tuple importance) and performance-related metrics (e.g., dropped tuples up to now)
  - Also: parameterizable Score Function, which can calculate from a VM the current impact of a message on QoS

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Data Perspective [Klein et al. 2009]

- Divide the stream into non-overlapping, jumping data quality windows for each attribute
- Window contains the values for the attribute, timestamp and a set of attributes, which contain values for quality dimensions
- Dimensions: accuracy, confidence, completeness, data volume, timeliness
- Distinguish operator classes: data-modifying (e.g., filtering, Join), data-generating (e.g, Interpolation), data-reducing (e.g., Projection, Sampling), data-merging(e.g., Aggregate
- Define quality operator analogs to operators
- Data quality operators implement a function which calculates a new quality value for elements resulting from the operator
- Implemented adaptive window size algorithms based on interestingness → finer granularity of windows at high peaks, threshold excess, fluctuations

Timestamp	...	210	220	230	240	250	260	270	280	290	300	310	320	330	340	350	360	370	380	390	400	...
Lifetime	...	300	298	295	292	292	292	292	283	274	265	255	252	250	242	233	206	195	190	187	184	...
Accuracy	...					3.0					3.3					2.78					2.86	...
Completeness	...					0.9					0.8					0.9					1	...

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Agenda

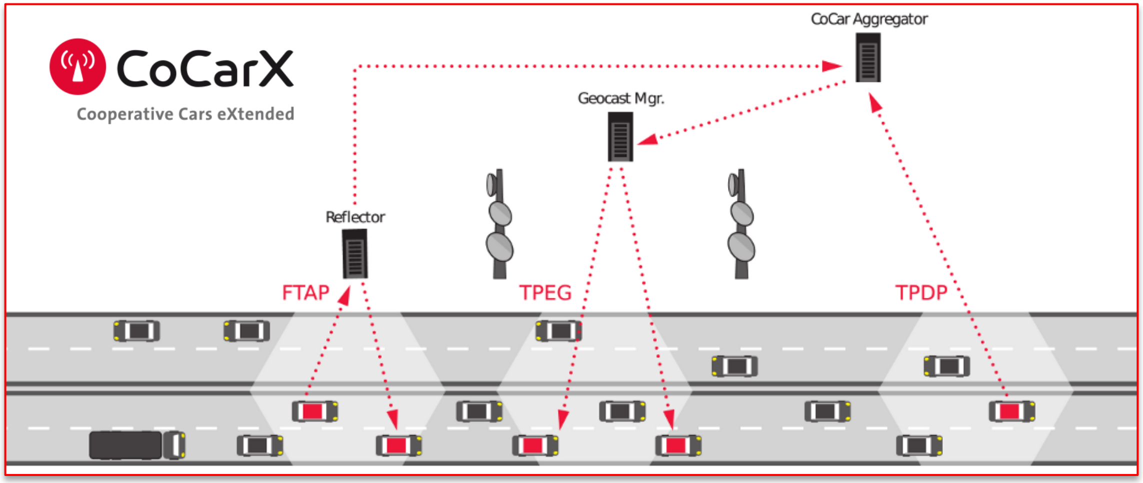
1. Introduction
2. Data Stream Management Systems
3. Query Languages
4. Query Plans & Operators
5. Quality Aspects in DSMS
6. **Our work**

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

The Cooperative Cars Project

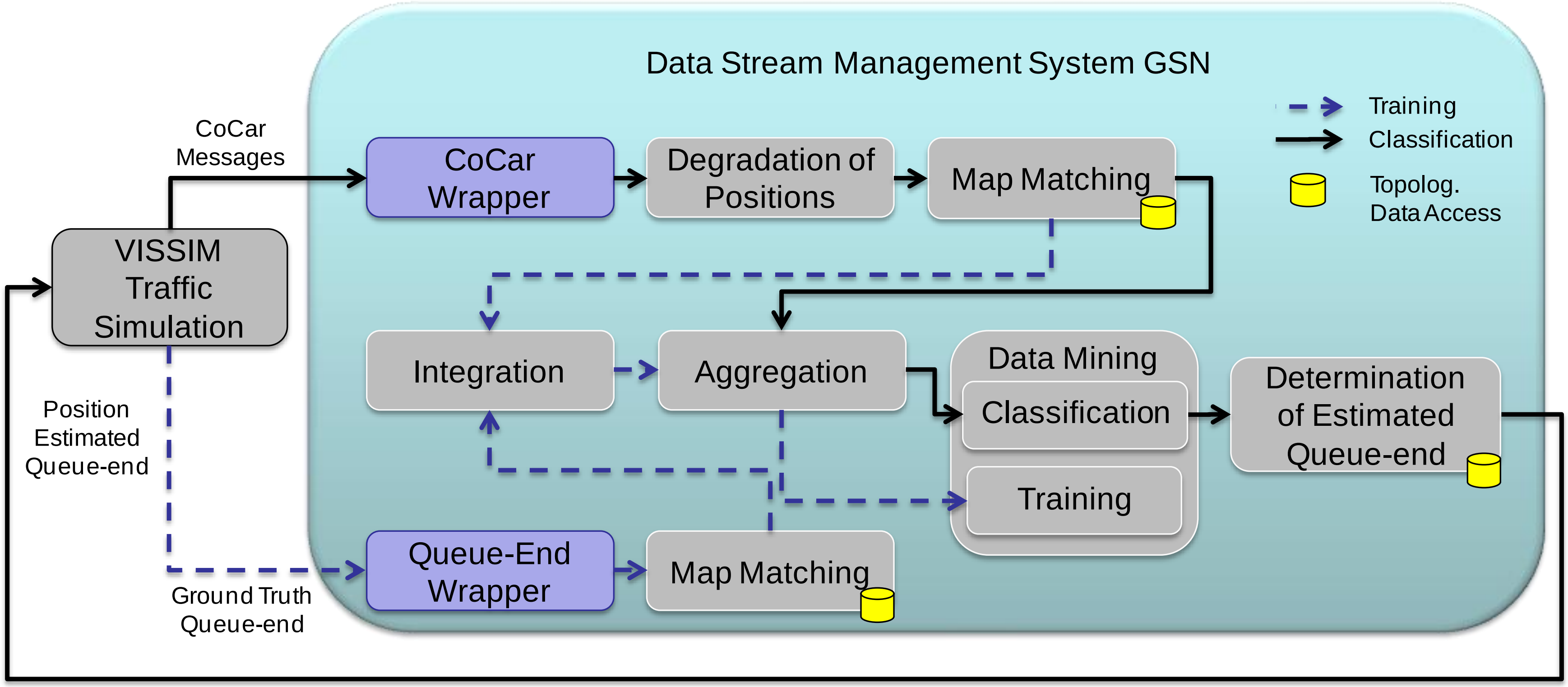
- Development of a Car2X communication infrastructure and according applications based on cellular networks (emphasizing 3G and 3G+)
- Application: send hazard warning messages over cellular network infrastructure, e.g., a vehicle braking very hard
- Poses challenges for mobile communication: latency, data privacy, reliability
- Poses challenges for data management & applications
  - High data rates → scalability, performance
  - Integration of multiple data sources
  - Information accuracy (e.g., Floating Phone Data)
  - Data stream mining to derive new information from events



Sandra Geisler
Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Queue-end Detection Scenario

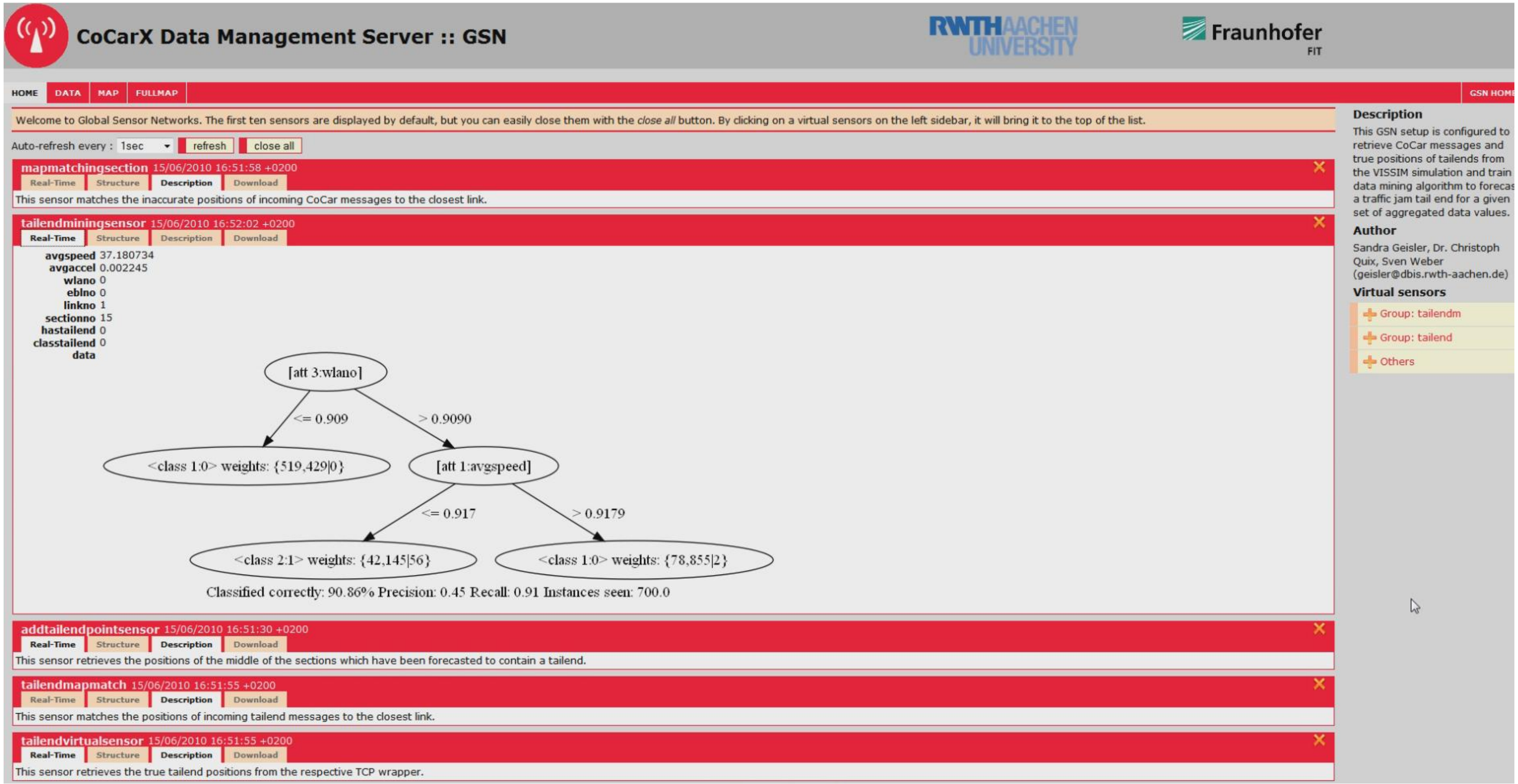
- Idea: Separate each road into sections and determine if it contains a queue-end → binary classification task
- Use CoCar messages as data sources only
- Use data stream mining → test which algorithm suits the task best and which parameters influence mining results



[Geisler et al. 2010]

Sandra Geisler
Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

Realization



References (1)

[Abadi et al. 2005] Abadi, D. J.; Ahmad, Y.; Balazinska, M.; Çetintemel, U.; Cherniack, M.; Hwang, J.-H.; Lindner, W.; Maskey, A.; Rasin, A.; Ryvkina, E.; Tatbul, N.; Xing, Y. & Zdonik, S. B. The Design of the Borealis Stream Processing Engine *Proc. 2nd Biennal Conference on Innovative Data Systems Research (CIDR),* **2005**, 277-289

[Abadi et al. 2003] Abadi, D. J.; Carney, D.; Çetintemel, U.; Cherniack, M.; Convey, C.; Lee, S.; Stonebraker, M.; Tatbul, N. & Zdonik, S. B. Aurora: a new model and architecture for data stream management.*VLDB Journal,* **2003***, 12*, 120-139

[Ahmad & Centintemel 2009] Ahmad, Y. & Cetintemel, U. Liu, L. & Özsu, M. T. *(ed.)* Data Stream Management Architectures and Prototypes. *Encyclopedia of Database Systems, Springer,* **2009**, 639-643

[Amini et al. 2006] Amini, L.; Andrade, H.; Bhagwan, R.; Eskesen, F.; King, R.; Selo, P.; Park, Y. & Venkatramani, C. SPC: a distributed, scalable platform for data mining. *DMSSP '06: Proceedings of the 4th international workshop on Data mining standards, services and platforms, ACM,* **2006**, 27-37

[Arasu et al. 2004] Arasu, A.; Babcock, B.; Babu, S.; Cieslewicz, J.; Datar, M.; Ito, K.; Motwani, R.; Srivastava, U. & Widom, J. STREAM: The Stanford Data Stream Management System. *Stanford InfoLab,* **2004**

[Arasu et al. 2006] Arasu, A.; Babu, S. & Widom, J. The CQL continuous query language: semantic foundations and query execution. *VLDB Journal,* **2006***, 15*, 121-142

[Biem et al. 2010] Biem, A.; Bouillet, E.; Feng, H.; Ranganathan, A.; Riabov, A.; Verscheure, O.; Koutsopoulos, H. & Moran, C. IBM InfoSphere Streams for Scalable, Real-Time, Intelligent Transportation Services. *Proc. of SIGMOD'10,* **2010**

[Babcock et al. 2002] Babcock, B.; Babu, S.; Datar, M.; Motwani, R. & Widom, J. Models and Issues in Data Stream Systems. *PODS 2002,* **2002**

[Chandrasekaran et al. 2003] Chandrasekaran, S.; Cooper, O.; Deshpande, A.; Franklin, M. J.; Hellerstein, J. M.; Hong, W.; Krishnamurthy, S.; Madden, S.; Raman, V.; Reiss, F. & Shah, M. A. TelegraphCQ: Continuous Dataflow Processing for an Uncertain World. *Proc. 1st Biennal Conference on Innovative Data Systems Research (CIDR),* **2003**

[Cherniack et al. 2009] Cherniack, M. & Zdonik, S. Liu, L. & Özsu, M. T. *(ed.)* Stream-Oriented Query Languages and Architectures. *Encyclopedia of Database Systems, Springer,* **2009**, 2848-2854

[Demers et al. 2005] Demers, A.; Gehrke, J.; Hong, M.; Riedewald, M. & White, W. A General Algebra and Implementation for Monitoring Event Streams.. http://hdl.handle.net/1813/5697 *Cornell University,* **2005**

[Gedik et al. 2008] Gedik, B.; Andrade, H.; Wu, K.-L.; Yu, P. S. & Doo, M. SPADE: the system s declarative stream processing engine. *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, ACM,* **2008**, 1123-1134

[Geisler et al. 2010] Geisler, S.; Quix, C. & Schiffer, S. Ali, M.; Hoel, E. & Shahabi, C. (ed.) A Data Stream-based Evaluation Framework for Traffic Information Systems Proc. 1st ACM SIGSPATIAL International Workshop on GeoStreaming, 2010

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen

References (2)

Girod et al. 2008] Girod, L.; Mei, Y.; Newton, R.; Rost, S.; Thiagarajan, A.; Balakrishnan, H. & Madden, S. XStream: a Signal-Oriented Data Stream Management System. *ICDE*, **2008**, 1180 -1189

[Golab and Özsu 2003] Golab, L. & Özsu, M. T.Issues in Stream Management SIGMOD Record, 2003, 32, 5-14

[Kang et al. 2003] Kang, J.; Naughton, J. & Viglas, S. Evaluating window joins over unbounded streams *Data Engineering, 2003. Proceedings. 19th International Conference on,* **2003**, 341 - 352

[Klein et al. 2009] Klein, A. & Lehner, W. Representing Data Quality in Sensor Data Streaming Environments *ACM Journal of Data and Information Quality,* **2009**, *1*, 1-28

[Krämer & Seeger 2009] Krämer, J. & Seeger, B.,Semantics and Implementation of Continous Sliding Window Queries over Data Streams. *ACM Trans. on Database Systems,* **2009**, *34*, 1-49

[Maier 2005] Maier, D.; Li, J.; Tucker, P.; Tufte, K. & Papadimos, V. Semantics of Data Streams and Operators.*ICDT 2005, Springer,* **2005**, 37-52

[Mokbel et al. 2004] Mokbel, M. F.; Lu, M. & Aref, W. G. Hash-Merge Join: A Non-blocking Join Algorithm for Producing Fast and Early Join Results *ICDE,* **2004**

[Patroumpas & Sellis 2005] Patroumpas, K. & Sellis, T. K. Window Specification over Data Streams *Current Trends in Database Technology - EDBT 2006 Workshops,* **2006**, 445-464

[Peng and Chawathe 2003] Peng, F. & Chawathe., S. S. XSQ: A Streaming XPath Engine *Technical Report CS-TR-4493 (UMIACS-TR-2003-62)., Computer Science Department, University of Maryland,* **2003**

[Stonebraker et al. 2005] Stonebraker, M.; Çetintemel, U. & Zdonik, S. B. The 8 requirements of real-time stream processing. *SIGMOD Record,* **2005**, *34*, 42-47

[Terry et al. 1992] Terry, D. B.; Goldberg, D.; Nichols, D. A. & Oki, B. M. Stonebraker, M. *(ed.)* Continuous Queries over Append-Only Databases. *Proc. ACM SIGMOD International Conference on Management of Data, ACM Press,* **1992**, 321-330

[Thakkar et al. 2008] Thakkar, H.; Mozafari, B. & Zaniolo., C. Designing an Inductive Data Stream Management System. the Stream Mill Experience *The Second International Workshop on Scalable Stream Processing Systems,* **2008**

[Urhan and Franklin 2000] Urhan, T. & Franklin, M. J.XJoin: A Reactively-Scheduled Pipelined Join Operator *Bulletin of the IEEE Computer Society Technical Committe on Data Engineering,* **2000**, *23*, 27-33

[Viglas 2005] Viglas, S. Chaudhry, N. A.; Shaw, K. & Abdelguerfi, M. *(ed.)* Query Execution and Optimization.*Stream Data Management, Springer,* **2005**, 15-32

[Zdonik et al. 2004] Zdonik, S.; Sibley, P.; Rasin, A.; Sweetser, V.; Montgomery, P.; Turner, J.; Wicks, J.; Zgolinski, A.; Snyder, D.; Humphrey, M. & Williamson, C. Streaming for Dummies, http://list.cs.brown.edu/courses/csci2270/archives/2004/papers/paper.pdf, **2004**

Sandra Geisler

Prof. Dr. M. Jarke
Lehrstuhl Informatik 5
(Informationssysteme)
RWTH Aachen