Processing Time Series Data

@SCALE in the CLOUDs using Confluent Platform

Dr. Mirko Kämpf
SA @Confluent
2020

Goals:

This deck will show how the OpenTSx demo can be used with Confluent cloud.

Overview

- Definitions: to define the scope
- Today's Challenges
- Goals of OpenTSx
- Planned Deliverables
- Summary

Definitions: (1)

TS:

- **unit of data** (like a record) with well defined properties (sampling interval, unit of measurement)
- enables usage of **repeatable algorithms** with variable algorithm-parameters (e.g., for auto-ML)

TSA: Time Series Analysis

- any kind of analysis in which timely ordered series of time stamped data points are used
- univariate: data point is a scalar value
- multivariate: data point is a vector, and all dimensions are used
    - *the vector elements are either from multiple time series (pairwise analysis) or multiple observed properties*
    - *if only one dimension out of a vector is used for calculation, it is still univariate TSA*

TSDB: Time Series Database

- can be a key-value store or an RDBMS or an Excel-Sheet
- keeps data points and provides timely ordered data points (called time series)

Definitions: (2) - Prefered Representations

Panel Data: https://en.wikipedia.org/wiki/Panel_analysis

- well adopted format for representing time dependent data to analysts

Example [edit]

balanced panel:

person	year	income	age	sex
1	2001	1300	27	1
1	2002	1600	28	1
1	2003	2000	29	1
2	2001	2000	38	2
2	2002	2300	39	2
2	2003	2400	40	2

unbalanced panel:

person	year	income	age	sex
1	2001	1600	23	1
1	2002	1500	24	1
2	2001	1900	41	2
2	2002	2000	42	2
2	2003	2100	43	2
3	2002	3300	34	1

TS-VA: Time Series Vector Analogy

- equidistant observations are given as a list of double values
- each double value is the measured value at time t = t_0 + i * dt (where i is the index in the list, t_0 the time of first observation)
- Tuple with 4 MD elements and a TS-Vector (**TSV**):

```
(id, [context,] t_0, dt, (o1, o2, o3, … , on))
```

OPTIONAL

Definitions: (3)

Time Series Bucket:

- group of time series which is used as input for TSA
- like a table in an RDBMS
- result of ETL >> to bring content in shape >> data model
- result of EXPLORATION >> search the right sequence >> **episode search**
    - typically, we find only tag-based search
    - a real property based search is hard to find

Consistent Data Model

for time series data across *all data components* ***in CP***

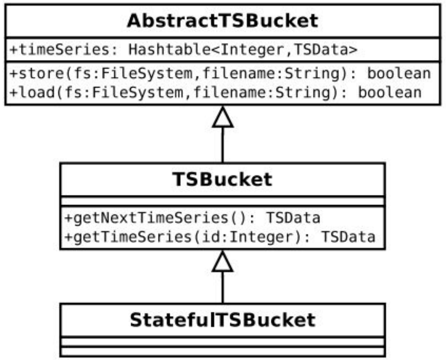


Fig. 3. Class diagram for the core TSBucket implementation. An abstract class implements all necessary functionality to handle time series data by a standalone application or within the a MapReduce job. The BucketCreator is used to create an TSBucket, which is stored in a binary data file. The SequenceFileInputFormat, which is part of the Hadoop distribution, passes the data record by record (a record is time series in this context) to the mapper of a MapReduce program. This procedure is optimized for highly parallel processing.

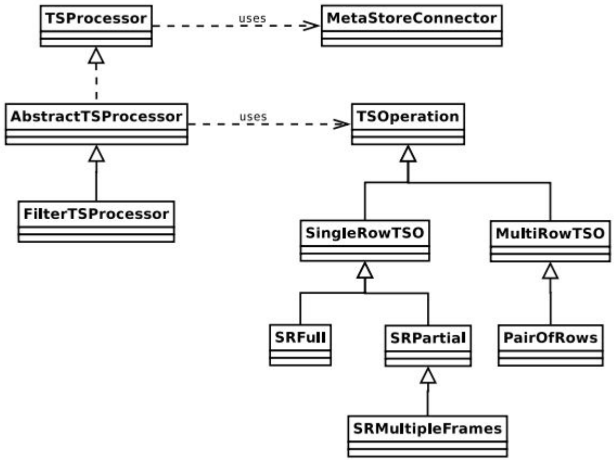
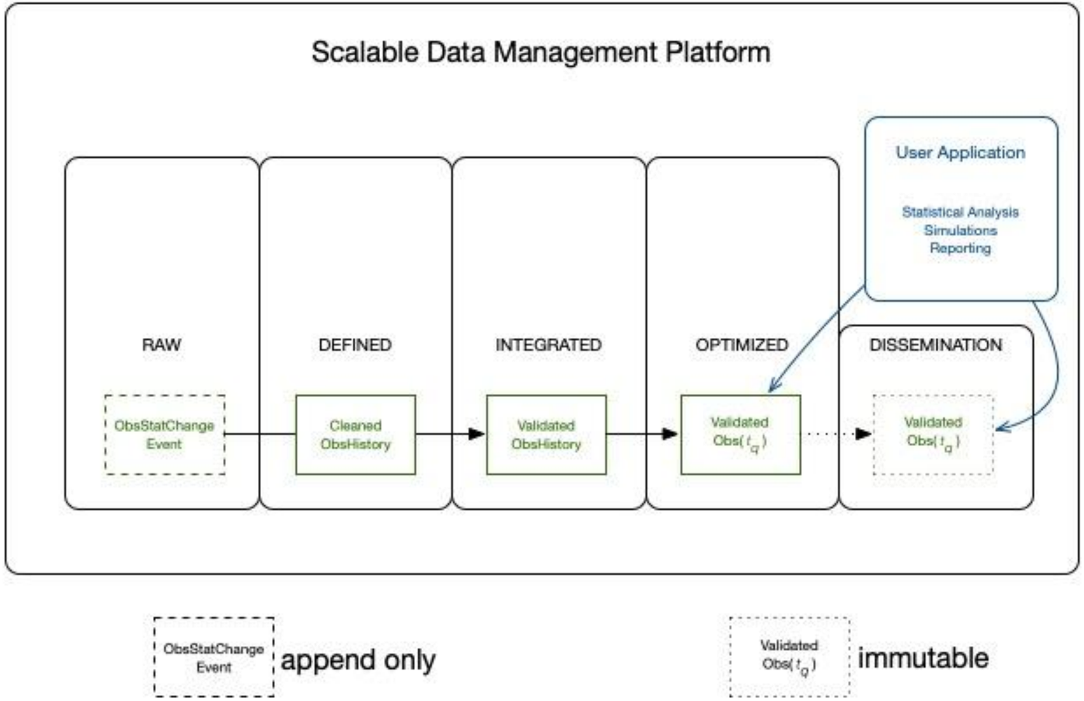


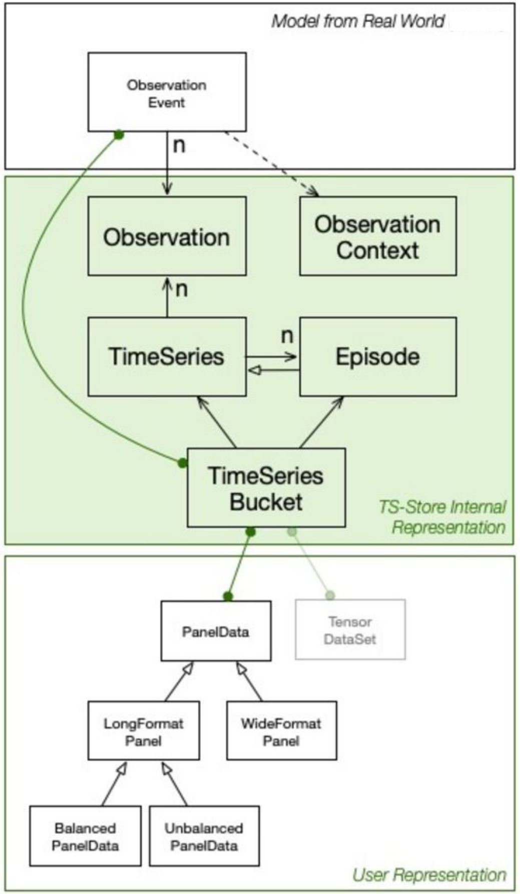
Fig. 4. Class diagram for the TSProcessor and the TSTool implementation. The AbstractTSProcessor implements a connection to an external metadata store. The FilterTSProcessor is a kind of a map-side join implementation for a standalone application and all analysis functionality is implemented within the TSOperation classes.

https://www.ijcaonline.org/archives/volume74/number17/12974-0233 (2013)

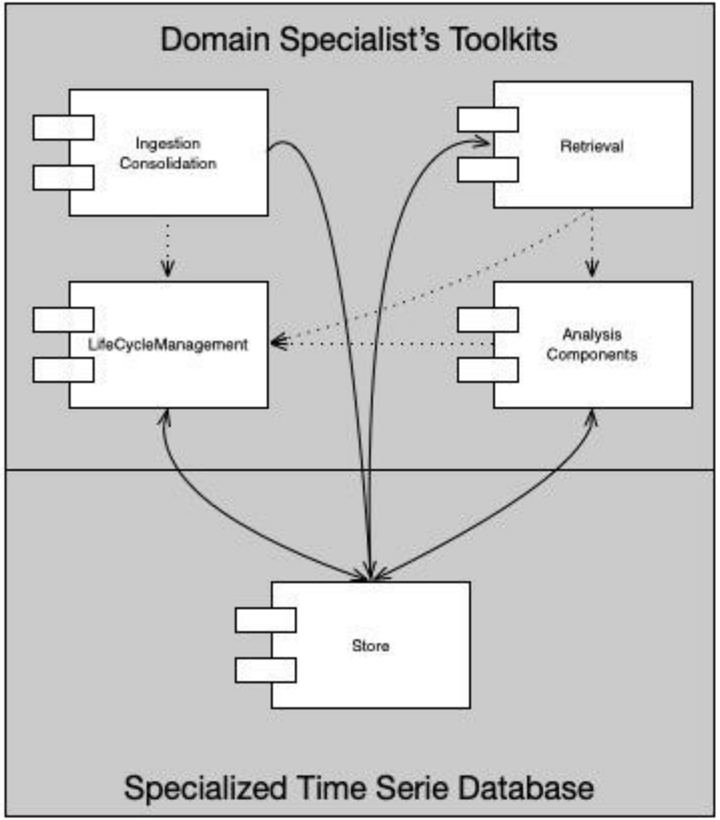
From Observations to Time Series Objects … to Panel Data

Event and episode transformation are a typical workload for our streaming processing platform.





Two (of today's) Challenges:



How is Time Series (TS) Processing done today?

TSA is a crucial aspect of **any kind** of *modern technology*… no doubts about that!

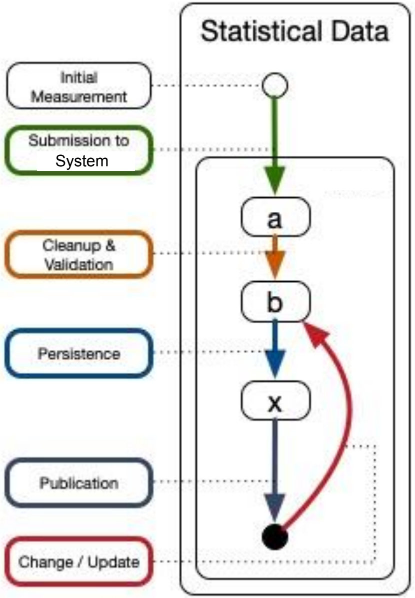
Event data is crucial for an digital business.
But: **TS data** is **not yet part** of many platform offering.

Customers still treat TS data as a special case using tools **outside our platform**.

Observations from the field:

1. typical TSDBs force you to **use a particular schema** to ingest data
   - VIOLATION of the "SCHEMA on read principle"
   - we could rather stage all raw data before it is stored in a **special purpose system**
2. **search** for similar episodes and pattern search are **rare**
   - data exploration is a challenge for time series data
3. which time is needed?
   - measurement time
   - delivery time
   - processing time ….
     - typically, the TSDB data model can only handle ONE TIME
4. data set life-cycle and governance
   - changing an observed value is usually possible, but then we lose the previous information, if versioning is not available
   - this is not acceptable in case of financial institutions => all changes need to be recorded
   - this topic is also relevant in technical contexts, but often ignored

Life Cycle of an Observation



Each observation of any particular metric is modeled as a state machine.

The *ObsState* dataset is a collection of state machines in various states. Filtering provides a separation of *Observations* by state.

The latest state is in the scope of the representation of the *ObsState* dataset.

The state at time *t* can be derived from the state change event data set.

The *ObsStateChangeHistory* is an immutable log of all actions related to any *Observation*.

How to address those challenges?

Those, who believe in ML/AI and data driven decisions must handle
TS data and TS analysis as first class citizen of their IT systems.

1. easy data integration across domains
   >> manage consistency
2. comprehensive algorithm collections
   >> shorter time to market
3. highly optimized access paths
   >> efficient and economical dissemination of data

OpenTSx aims on a consistent integration of required components (data model, functions, parametrised streaming apps) in Confluent platform.

Goals of OpenTSx ?

Open Time Series toolbox:

- set of reusable robust artefacts
  on top of Apache Kafka

- reliable and repeatable data processes,
  with full governance, ready for domain experts

- deep integration into domain specific special devices

- enrichment of the domain expert's experience
  with state of the art ML/AI on streaming data in the cloud

Next Deliverables:

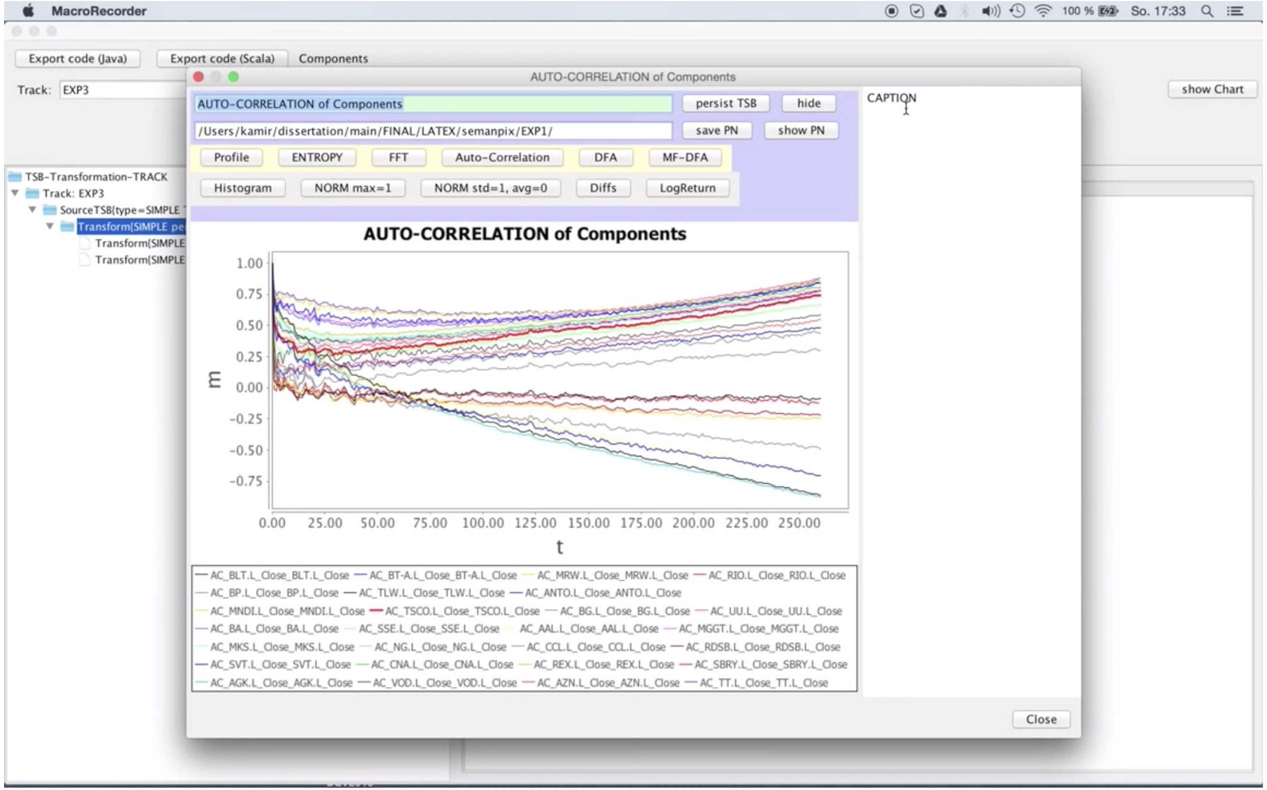
Components

Integration into **Visual Experiment Editor**

Deep integration

of ***edge devices***:

SDK, tutorials, demo, visual-workflow recording



https://www.youtube.com/watch?v=BQ8NHHZ-SuA (2015)

Access to Episodes via Consumer and Search

- Show access paths to time series data ...
- Episode search via special purpose systems
  - Elastic Search,
  - Cassandra

Extensible **Algorithm Library**

Rich **collection of integrated algorithms** from various libraries and frameworks.

- use a generic plugin mechanism and wrappers to integrate available implementations

- 1-st class support is planned for:
    - Deeplearning4J
    - TensorFlow / Keras
    - PANDAS
    - SCIKIT LEARN

Integration of Deeplearning4J

Show a demo component ...

Process Templates and **Code Generation**

Reusable workflow templates allow:

- faster experiment design and implementation
- faster migration from legacy tools/special purpose systems which don’t scale

```
TSBucket tsbComponents = tsbCollection.processBucket( "CACHE", null );
TSBucket tsbShuffled = tsbComponents.processBucket( "SHUFFLEYVALUES(1)", null );
TSBucket tsbDFA_2Shuffled = tsbShuffled.processBucket( "DFA_OF_", null );
TSBucket tsbAC_Components = tsbComponents.processBucket( "AC_OF_", null );
TSBucket tsbFFT_Components = tsbComponents.processBucket( "FFT_OF_", null );
TSBucket tsbDFA_2Components = tsbComponents.processBucket( "DFA_OF_", null );
TSBucket tsbPDF_Components = tsbComponents.processBucket( "PDF_OF_", null );
TSBucket tsbLogRETURN_Components = tsbComponents.processBucket( "LOGRETURN_OF_", null );
TSBucket tsbStandardized_Components = tsbComponents.processBucket( "STANDARDIZED_OF_", null );
TSBucket tsbH_Components = tsbComponents.processBucket( "H_OF_", null );
```

- this flow can be processed on a local workstation, or on a local Confluent platform, and in cloud based containerised applications

Create Kafka Streams Topology from Experiment

- using KSQL UDFs we can indirectly create a topology
- Java code generation can be used for direct topology creation

Summary

OpenTSx delivers:

Unified concept for data representation
for time series analysis across *all data components* ***in Confluent Platform.***

Deep integration of edge devices:
SDK, quick start bundles, tutorials, demos.

Rich collection of integrated algorithms from various libraries and frameworks.

Reusable workflows for faster migration from legacy tools.

THE CALL:

- Confluent can lead customers and partners to the future of time series processing in the cloud.

- This has to go **far beyond** using *time series databases*, and it is **much more than** just *sensor data collection*.

- We should open the right customer-budgets and **bring more TSA workloads into our platform**.

Read more:

The Time Series Analysis Toolbox in the cloud

https://docs.google.com/document/d/1mRHB_G_wJS3U2ZbemiBdwA9lsv_Q5Kq25amqo-5Im3Q/edit#heading=h.8swi2s6oxv7t