Fachhochschule Nordwestschweiz
Hochschule für Architektur, Bau und Geomatik

Bachelor-Thesis 2020

# IoT Datenpipeline für die TimescaleDB

**Autor:** **Tobias Hofmann**

**Examinator:** **Prof. Martin Christen**

**Experte:** **Bernhard Draeyer**

# IoT Datenpipeline für die TimescaleDB

**Die Anzahl der Sensoren und Sensortypen, die auf Baustellen verwendet werden, ist in den letzten Jahren stark angewachsen. In Zusammenarbeit mit der in-Terra GmbH wird eine neue Datenpipeline entwickelt, um Sensordaten abzuholen, allenfalls Umrechnungs- und Korrekturformeln anzubringen und die Daten in der Zeitreihendatenbank TimescaleDB zu speichern. Dabei wird Apache Kafka verwendet, eine Plattform für die verteilte Verarbeitung von Datenströmen, um eine horizontal skalierbare Datenpipeline zu entwicklen.**

**Schlagworte:** IoT, Python, MQTT, Apache Kafka, PostgreSQL, TimescaleDB, Sensor

## 1. Ausgangslage

Terryx ist eine umfassende digitale Lösung für das operative Baustellenmanagement der in-Terra GmbH. Ein Bestandteil sind diverse Sensoren, die Messdaten über LoRaWAN an *The Things Network* (TTN) senden. Von dort aus werden die Messdaten über MQTT eingelesen und in einer NoSQL-Datenbank gespeichert. Die Messdaten werden auch, wenn nötig, umgerechnet.

## 2. Aufgabenstellung

In dieser Arbeit wird unter Verwendung von Apache Kafka eine neue Datenpipeline entwickelt. Dafür müssen die Schnittstellen, um die Daten von MQTT nach Kafka weiterzuleiten und die Schnittstellen zur TimescaleDB konfiguriert sowie ein Datenmodell für die TimescaleDB entwickelt werden. Weiter wird mit dem Python-Modul Faust, einer Lösung für die Datenstromprozessierung, eine Applikation zur Anwendung von Umrechnungs- und Korrekturformeln entwickelt.

## 3. Kernkomponenten

**Apache Kafka** ist eine Plattform für die verteilte Verarbeitung von Datenströmen. Grundsätzlich ist Kafka ein *Publish/Subscribe Messaging* System. Dies bedeutet, dass keine direkten Verbindungen zwischen den Klienten, die Daten veröffentlichen, und denen, die Daten empfangen besteht. Stattdessen werden alle Daten kategorisiert und über einen zentralen Nachrichtenbroker verteilt. Die Kategorisierung löst Kafka, indem die Daten auf ein *topic* veröffentlicht werden. Kafka verfügt über mehrere Programmierschnittstellen (APIs), mittels denen mit einer populären Programmiersprache wie Python Daten eingefügt, prozessiert und empfangen werden können.

**TimescaleDB** ist eine für die Speicherung von Zeitreihendaten optimierte Open-Source-Datenbank. Sie wird als Erweiterung von PostgreSQL installiert und es kann über die gleichen Zugriffsmethoden, über SQL auf die Daten zugegriffen werden. Zusätzlich zur effizienten Speicherung von Zeitreihendaten bietet TimescaleDB typische Zeitreihendaten-bank-Funktionen wie Online-Aggregate über ein beliebiges Zeitinervall.

# 4. Datenpipeline

Eine Nachricht mit den Messdaten eines Sensors durchläuft folgende Schritte (vgl. Abb. 1):

1. Weiterleitung vom MQTT-Broker auf ein Kafka-*topic* (*MQTT Source Connector*)
2. Die Python-Applikation empfängt die Nachricht und extrahiert alle benötigten Infomationen wie den Sensoridentifikator, der Messzeitpunkt und die Messwerte. (*Streams Processor API*)
3. Ist eine Umrechnungs- oder Korrekturformel für den Messzeitpunkt und den Sensor vorhanden, wird die Formel auf die Messdaten angewendet. (Python-Modul *Faust*)
4. Eine neue, umstrukturierte Nachricht mit dem umgerechneten Messwert wird generiert und auf ein Kafka-*topic* veröffentlicht. (*Streams Processor API*)
5. Die umstrukturierte Nachricht wird in der TimecaleDB gespeichert. Dabei müssen die Felder der Nachricht den Spalten in der Tabelle entsprechen. (*JDBC SinkConnector*)

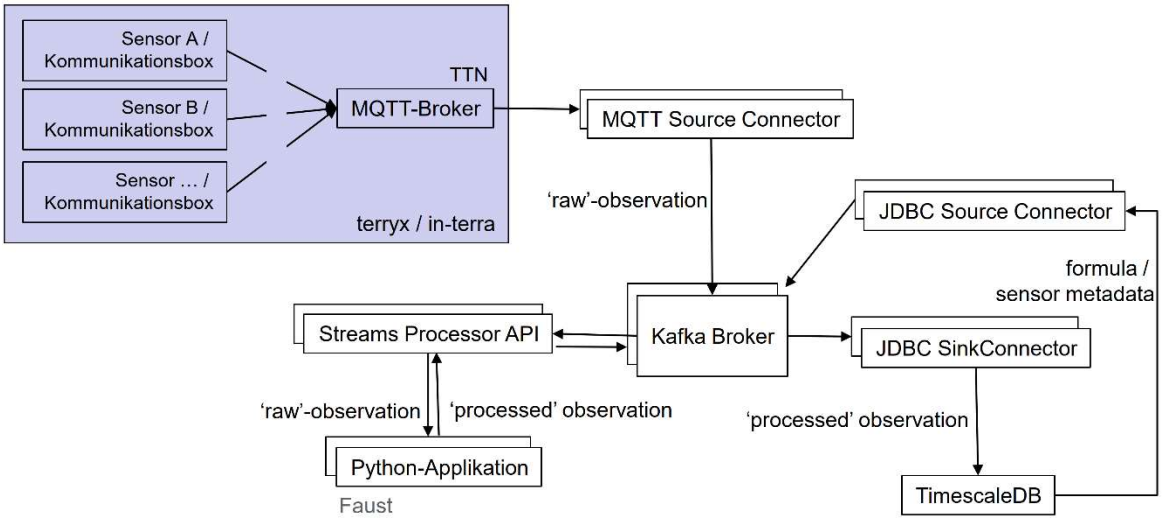


Abb. 1 Übersicht der Datenpipeline

# 5. Fazit

Apache Kafka bietet umfangreiche Konfigurationsmöglichkeiten. Auf der anderen Seite ist die Konfiguration von Kafka und den Schnittstellen entsprechend komplex und erfordern einen hohen Zeitaufwand. Während den Tests der Pipeline auf dem lokalen Computer benötigte die Python-Applikation mit Abstand die grösste Rechenleistung. Am wenigsten Rechenaufwand benötigte die TimescaleDB mit einer CPU-Auslastung von unter 5 %. Dabei konnten etwa 50'000 Nachrichten pro Minute beziehungsweise etwa 800 Nachrichten pro Sekunde verarbeitet werden. Diese bereits beachtliche Verarbeitungskapazität kann durch die Kafka-eigene horizontale Skalierung je nach Bedarf gesteigert werden.

# Kontakt


| | | |
|---|---|---|
| Autor: | Tobias Hofmann | hofmann.tobias@live.com |
| Examinator: | Prof. Martin Christen | martin.christen@fhnw.ch |
| Experte: | Bernhard Draeyer | info@in-terry.ch |