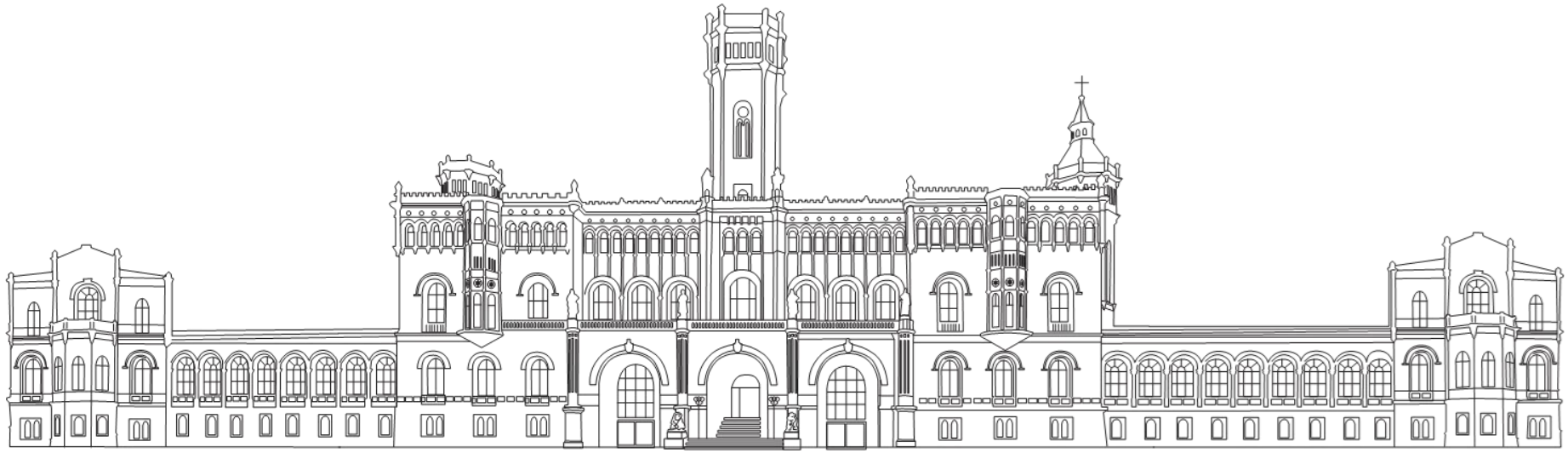
Wie Messdaten und ihre Metadaten standardisiert erfasst und aufbewahrt werden können - Ein Praxisbeispiel



Erfahrungsaustausch FDM

Stefan Warnken

26. April 2024

Übersicht

- **Welche Problemstellungen führten zur Vereinheitlichung der Messdatenformate beim ISD?**
- **Mittel- bis langfristige Ziele**
- **Beispielprojekt LUMO**
- **Auswahlkriterien der Datenformate**
  - **Apache Parquet**
  - **Matlab**
- **Metadaten**
- **Ausblick: Live-Streaming**
- **Komfort als Schlüssel zur Vereinheitlichung**

Ausgangspunkt & Problemstellungen

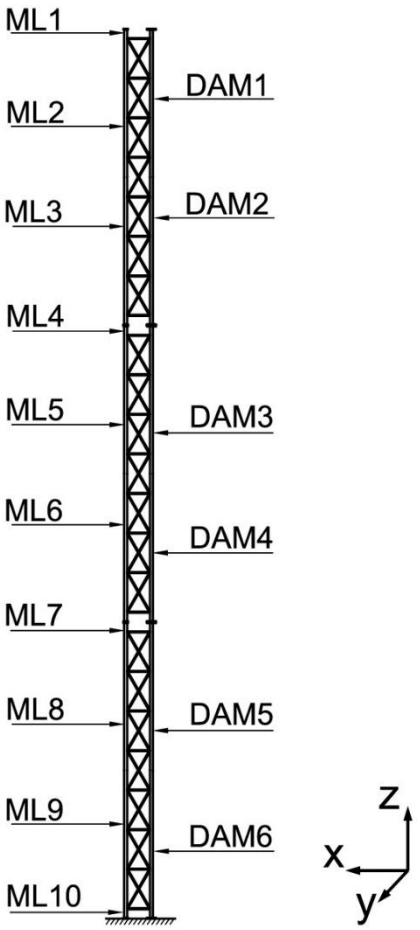
- Der Forschungsdatenbestand des Instituts beträgt aktuell mehr als 100 Terabyte. Eine kosteneffiziente Aufbewahrung ist somit nötig.
- Abteilung Composites / Materialforschung: Die Prüfverfahren sind teilweise standardisiert, benötigten aber ggf. zusätzliche Dokumentation (Zeichnungen, Bilder)
- Abteilung Schwingungen: Messungen im Feld: Windenergieanlagen, Baustellen, Brücken, Strukturüberwachung
- Es wird mit unterschiedlicher Messsoftware gearbeitet
- Es wird mit unterschiedlicher Auswertungs- und Simulationssoftware gearbeitet
- Die Forschungsdaten sollen ggf. externen Forschungspartner:innen zur Verfügung gestellt werden

→ Wie lässt sich möglichst schnell beurteilen, ob eine vorherige Messkampagne als Vergleich für aktuelle Messungen herangezogen werden kann? Wie kann ich die Daten mit meiner Auswertungssoftware nutzen?

Ziele

- Reduzierung des benötigten Speicherplatzes
- Möglichst offene Formate für langfristige Les- und Nutzbarkeit
- Das Format sollte alle Informationen vollumfänglich sichern können. Beispielsweise die Genauigkeit der Werte, mehrdimensionale Arrays etc.
- Metadaten sollen in der Messdatendatei hinterlegt werden, damit diese nicht von den eigentlichen Messdaten getrennt werden
- Metadaten sollen sowohl für Menschen als auch für Maschinen lesbar sein, um ggf. später Suchfunktionen für Messdaten erstellen zu können.
- Die Konvertierung in das Aufbewahrungsformat soll automatisierbar sein.

Beispiel Strukturüberwachung LUMO



ML	ML1	ML2	ML3	ML4	ML5	ML6	ML7	ML8	ML9	ML10
Channel names	accel01x	…02x	…03x	…04x	…05x	…06x	…07x	…08x	…09x	strain01
										strain02
	accel01y	…02y	…03y	…04y	…05y	…06y	…07y	…08y	…09y	strain03
										temp01
in m)	8.95	8.00	7.00	5.95	5.00	4.00	2.95	2.00	1.00	0.15

https://doi.org/10.1002/stc.3077
https://data.uni-hannover.de/dataset/lumo

Auswahl Datenformat(e)

- Das gesuchte Datenformat sollte von Matlab unterstützt werden, da fast alle am Institut mit dieser Software arbeiten
- Es sollte aber auch von Python unterstützt werden, das zunehmend populär wird und insbesondere bei ML / AI-Anwendungen verwendet wird.
- Idealerweise soll das Format auch auf Object Storage wie S3 abgelegt werden können und sich direkt von dort öffnen lassen.
- Es soll die Daten möglichst effizient speichern und für sehr große Datenmengen geeignet sein.
- Metadaten sollen unkompliziert gespeichert werden können.

In die engere Auswahl kamen damit:

- HDF5 bzw. NetCDF4 (vgl. https://de.wikipedia.org/wiki/Hierarchical_Data_Format und https://de.wikipedia.org/wiki/NetCDF) wie beim Forschungspark Windenergie https://forschungspark-windenergie.de/
- CSV in Zip-Dateien
- Apache Parquet (https://en.wikipedia.org/wiki/Apache_Parquet)
- (Matlab) <= v7.2 oder >= v7.3

Apache Parquet

- OpenSource Format unter Apache 2 Lizenz
- Spaltenbasiert inkl. Komprimierung. Im Vergleich ist der Speicherplatzbedarf am Geringsten
- Auf Zeitreihen optimiert. Zeitstempel / Datetime können als solche gespeichert werden.
- Wird von Matlab für den Datenaustausch mit Python empfohlen.
- Metadaten werden grundsätzlich unterstützt, aber es besteht noch Vereinheitlichungsbedarf und Matlab unterstützt bisher keine Metadaten in Parquet
- Variablentypen und Genauigkeit bleiben erhalten
- Kommt aus dem Bigdata-Bereich (Hadoop) und ist auf sehr umfangreiche Daten ausgelegt
- Zugriff auf Objectstorage ist möglich

Einschränkungen / Probleme

- Flache Tabellen. Keine mehrdimensionalen Arrays möglich
- Metadaten können mit Python problemlos genutzt werden, jedoch nicht mit Matlab

Matlab .mat

Das hauseigene Matlab-Format ist nicht so einheitlich, wie erwartet: Die Formate bis Version mat7.2 lassen sich in Python lesen und schreiben. Das aktuelle Matlab 7.3 Format (seit Matlab 2006b) entspricht HDF5, hat aber proprietäre Erweiterungen, beispielsweise für Timestamps.

Vorteile:
- intuitive Nutzung in Matlab
- Fast so effiziente Speicherung wie Parquet
- Komplexer Aufbau möglich wie mehrdimensionale Arrays
- Metadaten lassen sich gut integrieren.
- Die am Institut entwickelte Software kann ohne Änderungen weiterverwendet werden.

Nachteile:
- Proprietäres Timestamp-Format, das sich von Python aus bis auf weiteres nicht lesen lässt.
- Ggf. zunehmende Abhängigkeit von Matlab
- Schulungsaufwand, da nicht automatisch das aktuelle Format verwendet wird.

Anforderungen Metadaten

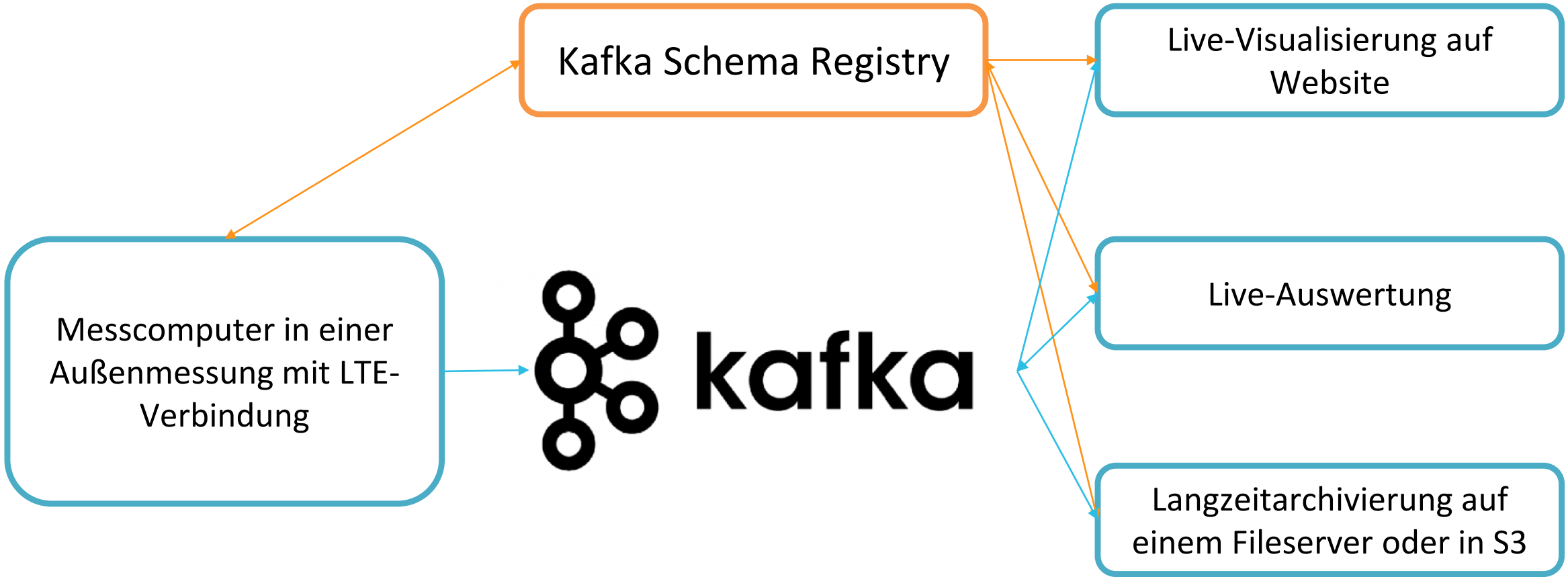
- Metadaten sollen eine schnelle Einordnung der Messung ermöglichen – z.B. für Vergleichsstudien
- Sie sollen sich sowohl sich sowohl aus .mat-Dateien als auch aus .parquet-Dateien in ein einheitliches Format umwandeln lassen, um dieses zum Aufbau einer Datenbank nutzen zu können.
- Eine Schema-Evolution soll möglich sein.
- So viele Metadaten wie möglich sollen automatisch von den Messgeräten übernommen werden
- Alle weitere Daten sollen komfortabel editierbar sein.

→ Wir haben uns für JSON als Format für die Metadaten entschieden und speichern sie auf je unterschiedliche Weise in Matlab und in Python.

Beispiel Metadaten

Name der Variablen	Typ	Zwingend erforderlich	Erklärung
schema_version	utf8 str	ja	“0.2“ – Dies wäre die Startversion. Wenn wir nachträglich die Metadaten ändern, werden wir darauf achten, dass alle neueren Versionen die alten Versionen lesen können bzw. jederzeit ein Update möglich ist.
fs	float	Ja	SampleRate in Hz
fs_by_device	float	Nein	Vom Messaufnehmer zurückgemeldete tatsächliche Rate
start_time	ISO 8601 str	Ja	Startzeit der Messung
time_interpolated	Bool (ja oder nein)	nein	Ist die Zeit aus dem Beginn der Messung und der Frequenz errechnet oder ein tatsächlicher Zeitstempel?
time_zone	ASCII-Str	Ja	UTC, immer UTC!
title	Utf-8 str	Nein	Name der Messung. Z.b. "LUMO - Leibniz Universtity Test Structure for Monitoring"
source	Utf-8 str	Nein	Beispiel: Messung im Rahmen des Projektes LUMO Phase 1
device	Utf-8 str	Nein	Hersteller und Modell des Messaufnehmers
software	Utf-8 str	Nein	Software inkl. Version, mit der aufgezeichnet wurde.
license	UTF-8 str	Nein	Zur Auswahl. Zum Beispiel "Creative Commons Attribution 3.0"
copyright	UTF8-str	Nein	"Leibniz Universit\u00e4t Hannover, Institut f\u00fcr Statik und Dynamik"
authors	List of UTF-8 str	Nein	Wer hat die Messung durchgeführt. Namen oder Kürzel.
test_id	Utf-8 str	Nein	Um zum Beispiel bestimmte Schädigungen den Messdaten zuordnen zu können
description	Utf-8 str	Nein	Versuchsbeschreibung. Unstrukturierte Informationen
condition_code	UTF-8 str	Nein	Sensor ausgefallen etc

Ausblick: Apache Kafka + AVRO – Datenströme in Echtzeit



Komfort als Schlüssel zur Vereinheitlichung

- Es gibt bisher keine expliziten Forschungsdaten-Expert:innen am Institut bzw. für die Fakultät
- Personelle Kapazitäten für die Erarbeitung eines langfristigen Forschungsdatenkonzepts sind nicht vorhanden.
- Die Einhaltung der vom Institut beschlossenen Kriterien wird bisher nicht überprüft.
- Ausgangspunkt für die Entwicklung waren die gebündelten Anstrengungen mehrerer wissenschaftlicher Mitarbeiter:innen, Messtechniker und IT-Systemadministratoren.
- Zentrale Hebel für die Umsetzung sind damit also für aktuelle und zukünftige Wissenschaftler:innen:
    - ✓ Eine fertige Anleitung für die Ablage meiner Forschungsdaten ist bereits vorhanden
    - ✓ Scripte für die automatische Konvertierung der Rohdaten vom Messgerät sind vorhanden
    - ✓ Die weiteren Metadaten lassen sich bequem über eine Eingabemaske hinzufügen und überprüfen
    - ✓ Die über Jahre am Institut entwickelten Simulationsroutinen lassen sich ohne großen Aufwand auf meine Messdaten anwenden
    - ✓ Es gibt einen erprobten Workflow, um die Daten mit AI / ML-Anwendungen auf dem Cluster der LUH zu analysieren
    - ✓ Es gibt aber auch Freiräume zur Abweichung bzw. zur Weiterentwicklung für neue Anforderungen

Dankeschön!

Für die Unterstützung, Anregungen, vorherige Vereinheitlichungen des Matlab-Formates, Spielräume zum Experimentieren und viel viel Messdaten-Knowhow möchte ich mich besonders bedanken bei:

Clemens Jonscher
Tanja Grießmann
Jan Heinemeyer
Leon Liesecke
Christian Claußen
Marcus Lüking
Britt Kahrger
Benedikt Hofmeister
Michael Treiber
Stefan Wernitz
Nikolai Penner

Vielen Dank für Ihre Aufmerksamkeit!